IJCLA ISSN 0976-0962

International Journal of Computational Linguistics and Applications

Vol. 3 No. 1 Jan-Jun 2012

Guest Editor
Grigori Sidorov

Editor-in-Chief
Alexander Gelbukh

ISSN 0976-0962

International Journal of Computational Linguistics and Applications

Vol. 3 No. 1 Jan-Jun 2012

International Journal of Computational Linguistics and Applications – IJCLA (started in 2010) is a peer-reviewed international journal published twice a year, in June and December. It publishes original research papers related to computational linguistics, natural language processing, human language technologies and their applications.

Indexing: Cabell's Directory of Publishing Opportunities.

Editor-in-Chief: Alexander Gelbukh

Subscription:	India:	Rs. 1999
	Rest of the world:	US$ 150

Payments can be made by Cheques/Bank Drafts/International Money Orders drawn in the name of BAHRI PUBLICATIONS, NEW DELHI and sent to:

BAHRI PUBLICATIONS
1749A/5, 1st Floor, Gobindpuri Extension,
P. O. Box 4453, Kalkaji, New Delhi 110019
Telephones: 011-65810766, (0) 9811204673, (0) 9212794543
E-mail: bahrius@vsnl.com; bahripublications@yahoo.com
Website: http://www.bahripublications.com

Printed & Published by Deepinder Singh Bahri, for and on behalf of **BAHRI PUBLICATIONS**, New Delhi.

International Journal of Computational Linguistics and Applications

Vol. 3 **No. 1** **Jan-Jun 2012**

CONTENTS

Editorial

This issue of IJCLA presents a selection of papers on two featured topics: sentiment analysis and information retrieval and, in addition, a paper on translation process research.

Nowadays we witness—and actively participate in—an explosion of interest to sentiment analysis, opinion mining, analysis of emotions and subjectivity in text, etc.: the vibrant human side of natural language processing technology, as opposed to the traditional study of cold and impersonalized grammar and statistics of language. We begin to discover for ourselves the huge body of real-life language, language of the "steet" of Internet, language of the blogs, tweets, and social networks, language that several years ago we would not even consider "proper" language—it's twitting, not speaking! ;-)) Apparently it is this kind of language that, due to a huge number of native "speakers" (native twitters?) in this "thumb generation", conveys enormous commercially important information—thus the explosion of interest to it from commercial companies.

To catch up with this trend, I selected for this issue of the journal a number of papers on this hot topic.

A. Wawer (Poland) shows how to automatically construct a dictionary of words expressing emotions, or sentiment, by analyzing a large corpus. As many other works on learning ontologies or dictionaries from corpus, this work employs automatically expanding patterns. In this particular case, the patterns make extensive use of morphology—a feature that is present in a huge number of languages but is not yet properly addressed because of its lack in English.

A. Bakliwal *et al.* (India) continue the topic of automatic construction of dictionaries of sentiment and subjectivity-related expressions. In this case, they rely on existing dictionary, WordNet, to achieve this goal. They target Hidni, a language that is native language for more than ten percent of the entire humankind but has not yet receive nearly as much attention from the computational linguistics community as it deserves. Their work (as well as the previous paper) can also be seen as an example of how such lexica can be constructed for languages other than English.

N. N. Bora (India) suggests an algorithm for extracting the majority opinion from a large number of tweets. This task has a great number of applications. Probably one of the most interesting and important application is for a company, political party or personality, artist, etc., to know what people really think of their products, their proposals, their performances, etc.—and not what smooth-tongued helpers tell them people supposedly think.

N. Konstantinova *et al.* (UK) continues this idea with a study of which features are characteristic of indication of the customers' opinions on commercial products. Such research is indispensable for building recommender systems that could help us, as customers, users, and buyers, to choose the best products and services and to avoid those that look attractive but prove to be flawed or even fraudulent.

M. Neunerdt *et al.* (Germany) close this section with a discussion of a small piece—part of speech tagging—of a much larger problem: to achieve all those goals described above, the researchers will have to develop powerful methods of analyzing and understanding this language of "native twitters" with the same quality as we used to understand the almost extinct nowadays "correct" language. The authors conclude that we are still in the beginning of this long but fascinating journey.

The other topic on which I selected papers for this issue is, in contrast, as old as Internet itself, but is still ever more important: information retrieval, our thread of Ariadne in the exponentially exploding Web, and information extraction, our current substitute for full language understanding.

D. Melo *et al.* (Portugal) propose a novel architecture for question answering systems, based on tight integration of ontologies, natural language processing, and information retrieval techniques. The architecture that they call cooperative question answering not simply answers the user's question when the user happens to ask the right question, but actively helps the user to improve the question if it cannot be successfully answered due to, say, ambiguity. In such cases the system would conduct a dialog with the user in order to achieve mutual understanding on what their information need is.

H. Imran and **A. Sharan** (India) discuss a more traditional technique a aimed to the same goal: to help the user to better formulate the query; namely, automatic query expansion: a technique that allows the computer to rewrite the user's query so that the result would better satisfy their information need. While traditionally this is done by propagating the relevance judgments via term co-occurrence, the authors suggest going beyond co-occurrences by using more advanced term similarity measures. They call their technique lexical cohesion based query expansion and claim that this idea has a great potential for improvement of information retrieval systems.

J. Makhlouta and **F. Zaraket** (Lebanon) address an important particular topic in information extraction (that is, almost language understanding): parsing and understanding temporal expressions—expressions referring to time: intervals of time, points in the past or future, age, etc. While this topic has recently enjoyed certain attention, the authors tackle it the context of a morphologically-rich language; what is more, Arabic—a language with unique and amazing Semitic intraflective morphology.

R. Winnemöller (Germany) addresses another, and much wider, topic in information extraction: named entity recognition, which is a task of detecting that several words (such as *United States of America* or *Santa Claus*) form a complex name of a single entity (in this case, a country or a person). He shows that his theory of sub-symbolic semantic Text Sense Representations (whatever it means—see Section 2 of his paper if you wonder) can be successfully applied to this task, too, as it has been already applied to many other natural language processing tasks.

Finally, I decided to include in the issue a paper on a rather unusual for this community topic: how the human behavior details while solving a linguistic task can be physically recorded and studied, which can potentially shed light to the underlying cognitive mechanisms.

M. Carl (Denmark) presents Translog II, a new version of a physical device and corresponding software capable of recording and analyzing the movement of human eyes when the subject translate text from one language to another. This very valuable information, totally unavailable to the researcher in the traditional text analysis setting, carries important details on which words cause delays and difficulties, how garden path situations are dealt with by the subject, etc.—which may ultimately uncover details of how this wonderful mechanism called language, which we the natural language processing community strive so much to understand and model, has been implemented in the first place: in the human brain.

I hope that this issue of the IJCLA journal will be both interesting and useful for wide readership, including both general public and students and professionals of natural language processing. Finally, I would like to thank the Editorial Board of the journal for inviting me to guest-edit this issue—it was a hard work but I enjoyed it a lot.

GUEST EDITOR:

GRIGORI SIDOROV
MEMBER OF MEXICAN ACADEMY OF SCIENCES.
CENTER FOR COMPUTING RESEARCH,
INSTITUTO POLITÉCNICO NACIONAL,
07738, DF, MEXICO
E-MAIL: <SIDOROV@CIC.IPN.MX>

Sentiment Analysis and Opinion Mining

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 11–24
RECEIVED 01/11/11 ACCEPTED 09/12/11 FINAL 16/06/12

Extracting Emotive Patterns for Languages with Rich Morphology

ALEKSANDER WAWER

Institute of Computer Science, Poland

ABSTRACT

This paper describes a method of acquiring emotive patterns for morphosyntactically rich languages. The goal is to maximize the recall of automatically generated sentiment lexicons in a resource lean fashion. The algorithm requires a small corpus with morphosyntactic annotations to acquire candidates for emotive patterns and evalute the vocabulary, and web as corpus for lexical expansion. The approach, which involves rule mining and contrast sets discovery, is demonstrated and evaluated for Polish.

1 INTRODUCTION AND EXISTING WORK

The research on automated sentiment lexicon acquisition typically falls into one of several categories.

One of the popular related paradigms is focused on using extraction patterns to acquire subjective resources [17, 1]. Subjectivity is a wide term which includes not only evaluations, but also opinions, emotions, and speculations. While its recognition is clearly useful in multiple tasks including opinion mining, the goal of this paper is to demonstrate that sentiment resource acquisition can be succesfully applied using resource-lean methods that do not directly depend on subjectivity[1].

Another approach is to extend WordNet lexicon with sentiment [7]. Despite obvious benefits, evaluative experiments [2, 4] reveal rather moderate successes of applications of this resource. Perhaps an even bigger

[1] We prefer to begin with sentiment acquisition and then extend it with subjectivity, which is both more general and resource-demanding.

problem is that it can not be applied to languages without a WordNet or a poorly developed WordNet – which is an issue we try to avoid with our algorithm.

For the reasons outlined above, the method described in this paper expands and continues efforts to automatically obtain lexicons of evaluatively connotatated words described by [8]. In this approach a set of "emotive patterns"[2] is submitted to a search engine to find candidates for evaluatively charged words. In the second step, semantic orientations (polarity and evaluative strengths) of candidate words are computed using pointwise mutual information – $SO - PMI$ measure as described by [16]. This is done by submitting them to a search engine to find their distributions in neighborhoods of paradigm positive and negative words.

The purpose of using emotive patterns is to select likely candidates for evaluative words. While it is theoretically possible to try all lexemes in a language using any sentiment assessment algorithm such as the $SO - PMI$, in practice it is extremely resource intensive and simply not feasible. The purpose of extraction using emotive patterns is to increase the probability of acquiring evaluatively charged words.

It is notable that any method relying on a fixed, predefined set of emotive patterns is constrained to extracing words that appear in a limited number of syntactic configurations. Consequently, acquired lexemes are often limited to certain part of speech types only, as is the case with patterns described in [8] extracting just adjectives and adverbs.

Put generally, the technique proposed in this paper aims at improving the recall of automated sentiment dictionary generation. Another closely related goal is to extract not only adjectives and adverbs, but also other part of speech types.

While the founding research [6, 12] and multiple subsequent sentiment studies such as [9, 18, 10] focused exclusively on adjectives, it is hardly disputable that nominal word forms and nouns can also carry negative connotations. This fact is at least partially confirmed by research on automated acquisition of subjective resources [15] and WordNet's sentiment extensions [7], but none of the methods can be applied in a resource-lean manner to acquire sentiment vocabulary.

This paper is organized as follows: in Section 2 we start by explaining the reasons of using two types of corpora and separation of pattern generation from lexical acquisition. Section 4 describes how we gener-

[2] We follow the original terminology here as proposed in [8], although one could use the notion of "sentiment extraction patterns" interchangeably.

ate initial pattern candidates, then in Section 5 we give an overview of the algorithm and cover in details contrastive sets attribute selection and greedy tag mining. Sections 6 and 7 present results and discussion of future work.

2 Morphosyntactic and Lexical Corpora

We start by hypothesizing that emotive patterns share certain *morphosyntactic properties*, relevant for acquiring new patterns. The goal is to learn properties which lead to discovering the best patterns given quality criteria discussed in section 3. The approach is similar to that in [15], where a bootstrapping process looks for words that appear in the same extraction patterns as the seeds and assumes that they belong to the same semantic class.

For expanding the set of emotive patterns, a corpus with morphosyntactical tags is required: queries to this corpus are based on combinations of morphosyntactic attributes, selected as potentially improving the quality of patterns which extract evaluatively connotated words. The key property of this approach is that size of this morphosyntactically tagged corpus is of secondary importance because it is not used for lexical acquisition in the sense of extracting sentiment candidate words.

The corpus we used was The National Corpus of Polish [3] with 3 millions segments[4] [13] and Poliqarp[5] query formalism. While sufficient to acquire candidates for emotive patterns, it is not extensive enough to acquire evaluative vocabulary by pattern continuations, because emotive patterns occur in the small, morphosyntactic corpus no more than tens of times and typically exactly once. For the same patterns, search engines like Google or Bing return hundreds or even thousands of results, depending on pattern. Then, on one hand it is straightforward that acquisition of candidates for evaluative words has to take advantage of lexical magnitude and proceed along the web as corpus approach. On the other, candidates for emotive patterns could potentially be extracted from smaller corpora with benefits from morphosyntactic information.

[3] `http://www.nkjp.pl`

[4] The notion of a segment roughly corresponds to a word. The distinction was introduced due to non-standard behaviour of certain rare morphological forms.

[5] `http://poliqarp.sourceforge.net`

3 Pattern Productivity

Algorithms presented in the latter parts of this paper may be evaluated in multiple ways. Quantifiable comparisons of the methods demand formulations of emotive pattern productivity metrics. Evaluation criteria for emotive patterns should involve two main factors:

- Number of distinctive lexemes set L returned by a pattern p.
- Aggregated, absolute $SO - PMI$ of all lexemes in L.

Thus, productivity pr of a pattern p is computed according to the following formula:

$$pr_p = \frac{\sum_{i=0}^{n} |SO - PMI(L_n)|}{\parallel L \parallel}$$

It promotes patterns which return many unique lexemes with high evaluative loading, either positive or negative.

4 Pattern Generation

This section discusses in a step by step fashion how we generate emotive pattern candidates, explaining the rationale and design choices.

4.1 *Part of Speech Sequences*

Probably the most simple conceivable approach to generating sequences which may be suspected of being emotive patterns is by focusing on parts of speech types that constitute known emotive patterns and searching the morphosyntactic corpus for sequences of lexemes which consist of selected part of speech types only. Let emotive patterns be created only by lexemes belonging to part of speech types listed in POS_{emot}:

$$POS_{emot} = \left\{ \substack{conj, ppron*, prep, qub, \\ praet, inf, fin, ger} \right\}$$

The above list is too unrestrictive because it admits a large class of nearly meaningless, but very frequent patterns, as for example sequences of particles (qub) and prepositions (prep). In fact, sequences composed of conjunctions, pronouns, prepositions and particles *only* are very unlikely to be emotive patterns because their role is mostly syntactic. Therefore, we introduced the second list of part of speech types POS_{na}: at least one part of speech type from this list is required to appear in any emotive pattern candidate, but also no emotive pattern may consist of lexemes belonging to these part of speech types only.

$$POS_{na} = \{conj, ppron*, prep, qub\}$$

Firstly, the morphosyntactic corpus is queried for sequences of at least two tokens consisting of lexemes that belong to POS_{emot}. Secondly, all sequences which contain only POS_{na} tokens are removed. However, the number of sequences is still too large and they mostly lack emotive pattern characteristics[6]. This is why subsequent filterings are still needed to approach a plausible set of emotive pattern candidates.

4.2 *Frequency Filtering*

The benefit of using word frequencies in pattern discovery was demonstrated recently by [5]. The key idea, applicable also in the context of emotive pattern discovery, is dividing patterns into high frequency syntactic word slots and slots for less frequent, content words which are actually the subject of interest in concept discovery. While the Davidov and Rappoport method ignored morphosyntactic information, we propose to mix both types of data, frequency and morphosyntactic tags, which seems reasonable for Slavic or German languages.

We introduce frequency as three valued rank variable, dividing word rankings at 200 and 5000. Contrary to Davidov and Rappoport, we include also frequency information about the medium frequency range.

4.3 *Sequence Contexts Filtering*

An intuitive method of such filtering is by checking left and right contexts of a sequence (candidate for an emotive pattern) for occurrences of known, highly evaluative words – the value was operationalized as $SO - PMI$ threshold. Number of rules after filtering with a range of tresholds values, for the set of rules $R1$, is illustrated in Figure 1.

The desired treshold value of $SO - PMI$ should be high enough to avoid patterns without highly evaluative words in their continuations.

Because most of sequences of interest appear exactly once in the morphosyntactic corpus, no further statistical elaboration and systematic

[6] It has been determined by random sampling and comparing average quality to known, effective emotive patterns.

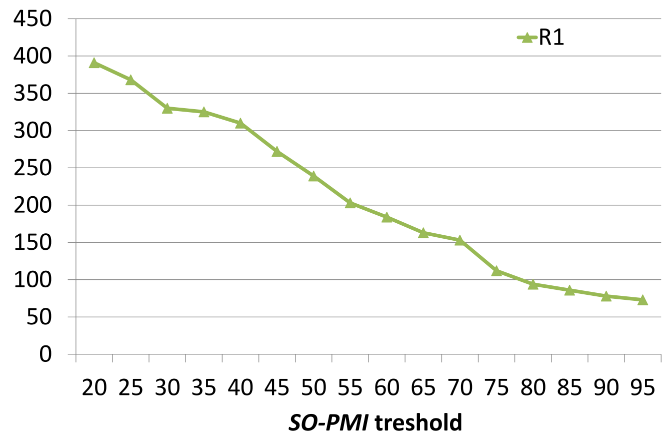


Fig. 1. Number of rules generated for a range tresholds values

analysis of sequence properties is possible, at least not on results obtained from the morphosyntactic corpus.

The proposed method of sequence generation and subsequent multistep filtering is designed to maximize the probability that obtained sequences are indeed emotive patterns. One obvious drawback of imposing presence of known evaluative words around pattern candidates is that it renders impossible detection of emotive patterns surrounded by evaluative words that are not yet recognized. On one hand, it is less of a problem after multiple iterations – once the list of emotive words is expanded. However, it still does not guarantee obtaining all evaluative words, because certain such words may (at least in theory) be obtainable only by a set of emotive patterns which never co-occur with any known evaluative word. The issue is at least partially taken into account by promoting patterns which lead to wide range of emotive words as in section 3.

5 Algorithm Overview

Figure 2 illustrates the processing scheme and core ideas of our approach.

The prerequisite is to prepare an initial seed of emotive patterns. Next steps of the method are iterative and can be summarized as follows:

1. Using any search engine and web as corpus method, issue queries for emotive patterns, get results, extract candidate words and compute their $SO - PMI$. Apply morphosyntactic analysis and disam-

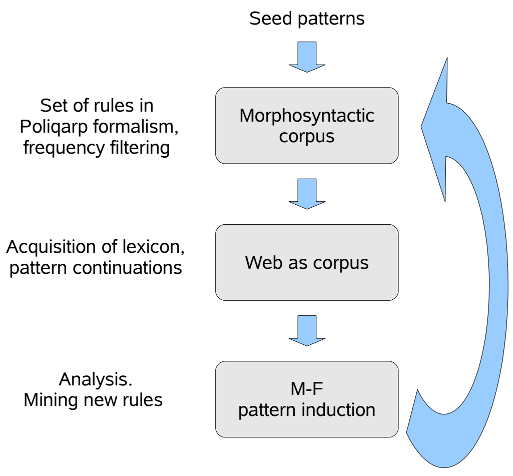

Seed patterns
Set of rules in Poliqarp formalism, frequency filtering
Morphosyntactic corpus
Acquisition of lexicon, pattern continuations
Web as corpus
Analysis. Mining new rules
M-F pattern induction

Fig. 2. Processing phases of the iterative emotive pattern acquisition

biguation so that each recognized token carries a list of associated tags.

2. Create a matrix M such that each row corresponds to one pattern match occurrence with contexts of k tokens each side of the sought candidate for evaluative word, along with morphosyntactic tags for each token and $SO - PMI$. Each emotive pattern is associated to corresponding M rows.
3. Select two sets of rows from M of approximately equal frequency by highest and lowest productivity of patterns that generated them. Call rows generated by high quality patterns M_{Hi-PR} and rows by low quality patterns M_{Low-PR}[7].
4. Using contrast sets mining [3] and association rule discovery as in Algorithm 1, learn a set of morphosyntactic and frequency rules $M-F$, describing the rows in M_{Hi-PR} according to optimization criteria defined by $SO - PMI$ as described below in more detail.
5. Apply $M - F$ rules on morphosyntactically annotated corpus to obtain a new set of emotive patterns.

[7] It seems reasonable to split the rows into groups based on the pattern that generated them and assign those groups into either M_{Hi-PR} or M_{Low-PR} – rather selecting individual rows. This is why frequencies might be different.

5.1 *Seed Patterns*

In the beginning of our experiment, 112 gramatically correct patterns were created by generating cartesian pairs from two sets of words, **A** and **B**. We disclose neither the exact words of **A** and **B** nor the two lists of paradigm positive and negative words, used in $SO - PMI$ computation. This is because both lists are language specific and do not contribute much to this paper. For a reference list, applied successfully in English, see [8].

What is important is that the initial (seed) patterns, submitted to a search engine, resulted in over 11 thousands web pages and 1381 unique lexemes obtained as pattern continuations.

5.2 *M-F Rules*

This section describes rules used to generate lexical patterns from the morphosyntactic corpus.

Rules, in case of Polish, follow tagging scheme defined in [14]. It is notable that this scheme also defines the feature space, which in our case is constituted by 17 variables spanned over 6 word positions (assuming 3 tokens left and 3 right from the extracted sentiment word placeholder). For Slavic and German languages the numbers and consequently tagging schemes will be not far from the one used here:

- Part of speech (in Polish, specific tags defined for over 35 token types).
- Morphosyntactic attributes (example: case, number, person for nouns and adjectives).
- Frequency information (as 3-valued rank).

Not all tag combinations are always present on all positions, therefore the actual feature space is typically more narrow. Rules are defined as sets of type:

$$position_{attribute} = value$$

For example:

$$k - 2_{case} = nom$$

denotes attribute $case$ positioned two tokens left ($k - 2$) from the placeholder for extracted evaluative word, whose value is nom (nominative). Rules define values (tag or frequency ranks) for corresponding attributes and thus every rule potentially selects a number of rows from M – where the values match.

A real example of a rule is presented below:

$$k{-}1_{number}{=}sg, k{+}3_{rank}{=}2, k{-}2_{aspect}{=}imperf$$
$$k{-}3_{rank}{=}1, k{-}3_{gender}{=}n$$

The rule consists of five attributes, two of them are frequency ranks at different positions. Extracted rules were very rarely longer than 5 or 6 attributes. This is caused by the greedy rule generation algorithm described in listing 1.

5.3 *Contrast Sets Attribute Selection*

For attribute (feature) selection, our method relies on contrast sets. In essence, rules that are later applied to the morphosyntactic corpus are formed only from those attributes (morphosyntactics, frequency and position), for which corresponding values (tags) differ meaningfully in their distributions across more and less productive rules.

For each attribute, we compare the corresponding distributions in M_{Hi-PR} and M_{Low-PR} filtering out attributes that do not differ significantly between the two matrices in terms of χ^2. As an example, let us consider attribute $k - 2_{case}$ (morpholosyntactic attribute "case" positioned two tokens to the left from where candidates for evaluative words occurred). Distributions of five cases, which occurred in M_{Hi-PR} and M_{Low-PR}, are as follows:

$$[121, 10, 197, 4, 3], [2, 0, 64, 0, 0]$$

We compute χ^2 and significance (0.0), thus $k - 2_{case}$ is considered in further analysis. For the first (seed) data matrix obtained, the contrastive sets method leaves only 54 attributes out of 102 possible for the initial matrix.

5.4 *Rule Generation*

In typical descriptive rule mining, algorithms typically seek surprising deviations or identify significant relationships using support, confidence, information theory measures or various combinations thereof.[8]

[8] An extensive coverage of various approaches, proposing unified terminology, is presented in [11].

Identification of morphosyntactic pattern configurations relevant for acquiring productive emotive patterns should, in our assesment, follow a different objective: the mechanism of rule induction should maximize absolute $SO - PMI$. This formulation seems the most relevant for our purpose and at the same time the most intuitive.

The rule generation algorithm is given in Algorithm 1. Let A denote the set of attributes. Rule r is then generated as follows:

```
$R_r$ = $\{a_r = t\}$ , $t$ : $\arg\max |SO - PMI|$
**foreach** *iteration* $i$ **do**
    **foreach** *attribute* $a$ *in* $A$ **do**
        **foreach** *tag* $t$ *of* $a$ **do**
            $|SO - PMI|$ for $\{R_i, a = t\}$
        **end**
    **end**
    $a, t$ : $\arg\max \Delta|SO - PMI|$
    $R_{i+1}$ = { $R_i$ , $a = t$ }
**end**
```

Algorithm 1: Rule generation

The algorithm begins by initializing empty rules R, one for every attribute r in A, and finds a tag which – for this specific attribute – maximizes its absolute $SO - PMI$ in M. Then, in iterative fashion, all other attributes and values are tested, and at the end of each iteration, an attribute and its tag value are appended to a rule, which maximizes absolute $SO - PMI$ of newly created rule.

The algorithm is greedy, because every tag and attribute selection is based on the principle of $SO - PMI$ maximization. This guarantees that every part of created rule contributes to absolute $SO - PMI$, but does not ensure finding globally optimal combinations which maximize $SO - PMI$. Such algorithm would have to compare every tag value of an attribute with all tag combinations of all other attributes, which gives complexity of $O(n!)$.

6 Results

Figure 3 presents $SO - PMI$ distributions for newly acquired lexemes – not seen in any of the previous iterations (unique).

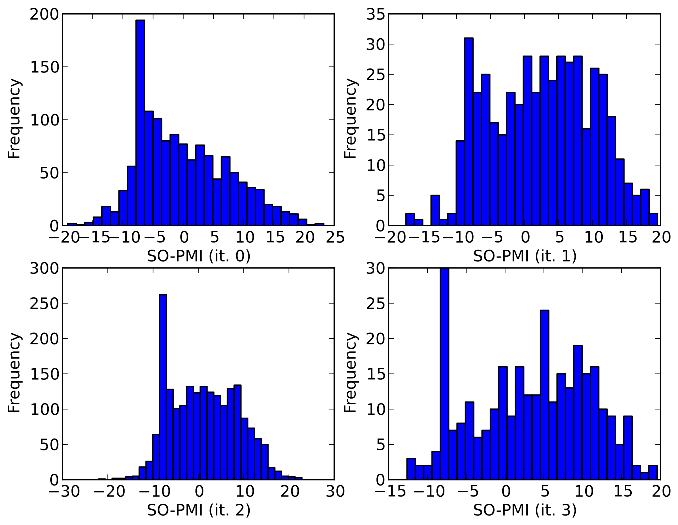


Fig. 3. Histograms of unique SO-PMI words acquired in 4 iterations

The histograms reveal that patterns generated automatically are at least as productive as the human made ones in terms of acquiring highly positive and highly negative words. However, the distribution of new vocabulary in terms of polarity appears to be different in each iteration: the first set of manually crafted patterns (iteration 0) leans towards negative words, iterations 1 and 3 are slightly skewed towards positive, while iteration 2 seems the most balanced, perhaps due to frequency.

Arguably, patterns created automatically have the advantage of generating more balanced distributions. Such results are easy to explain given the randomized pattern generation technique described in Section 4 which does not favour any specific polarity.

Interestingly, negative words tend to be much more densely distributed with a clear peak between -9 and -7. Distributions of positive words are more uniform. In each iteration certain amount of discovered words falls in between -5 and +5; it is likely that these should be considered neutral. The ratio of such words remains similar in each iteration and must be considered not avoidable.

Table 1 presents the results from a different perspective: organized by part of speech[9].

Table 1. Standard Deviation of $SO - PMI$ (std) and percentages of unique vocabulary acquired in each iteration, by part of speech types

	it.0		it.1		it.2		it.3	
POS	std	%	std	%	std	%	std	%
adv	8.04	17.49	6.91	3.97	5.72	3.6	4.68	2.29
pact	7.86	0.99	6.32	1.05	1.69	0.15	0.0	0.33
ger	0.0	0.0	8.27	1.05	7.48	0.64	5.79	0.98
pcon	0.0	0.0	0.0	0.0	0.0	0.05	0.0	0.0
fin	0.0	0.0	0.0	0.21	0.0	0.1	0.0	0.0
subst	6.68	27.06	7.55	66.11	7.71	55.46	7.54	65.69
ppas	0.0	0.33	8.8	0.84	6.03	0.69	1.04	0.65
inf	6.97	15.51	7.45	15.69	7.22	24.59	6.74	16.99
adj	7.69	38.61	7.57	9.62	6.81	13.73	7.18	11.11

The first iteration acquired many adjectives (38%). In subsequent iterations, percents of adjectives among unseen lexemes remained relatively smaller, with much larger relative amounts of nouns (subst). The quality of extracted lexemes, measured by the standard deviation of $SO - PMI$, did not vary a lot between iterations: further iterations were on average as good as the first one, with two exceptions: adverbs, which tended to be more centered around mean, and nouns, which tended to be more spread.

7 Conclusions and Future Work

The contribution of this work affects at least two aspects of sentiment analysis research.

First, it demonstrates how to expand sentiment lexicons beyond the limits of emotive patterns method described by [8]. This is possible thanks to the new iterative technique of vocabulary expansion, based on splitting the task into emotive pattern generation, which may be done by mining small morphosyntactic corpora and frequency information, and lexical acquisition using web as corpus framework.

[9] A description of POS types along with their abbreviations as in the table, can be found here: `http://nkjp.pl/poliqarp/help/ense2.html#x3-30002.1`.

Second, it allows including other part of speech types than just adjectives and adverbs. While this achievement is not distinctive in the context of sentiment lexicons (especially [7] and [15]), it seems one of very few approaches that do not rely on additional resources such as WordNets or specifically annotated corpora.

The method has been implemented and tested in Polish, but it is equally applicable to any language with rich morphology and syntax and comparable resources.

Future work will focus on evaluations against other feasible techniques of emotive pattern acquisition. Certain efforts should also be dedicated to finetuning of various parameters of the algorithm, such as for example $SO - PMI$ thresholds, and examination of net effects of each step of the procedure.

Finally, one natural extension is to re-implement it in other languages, Slavic or German.

ACKNOWLEDGEMENTS The research presented in this paper was supported by the project POIG.01.01.02-14-013/09 co-financed by the European Union under the European Regional Development Fund.

REFERENCES

1. Learning Multilingual Subjective Language via Cross-Lingual Projection (2007)
2. Balahur, A., Steinberger, R., Kabadjov, M., Zavarella, V., van der Goot, E., Halkia, M., Pouliquen, B., Belyaeva, J.: Sentiment analysis in the news. In: Chair), N.C.C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., Tapias, D. (eds.) Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10). European Language Resources Association (ELRA), Valletta, Malta (may 2010)
3. Bay, S.D., Pazzani, M.J.: Detecting change in categorical data: mining contrast sets. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 302–306. KDD '99, ACM, New York, NY, USA (1999), `http://doi.acm.org/10.1145/312129.312263`
4. Chair), N.C.C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., Tapias, D. (eds.): Is Sentiment a Property of Synsets? Evaluating Resources for Sentiment Classification using Machine Learning. European Language Resources Association (ELRA), Valletta, Malta (may 2010)
5. Davidov, D., Rappoport, A.: Effcient unsupervised discovery of word categories using symmetric patterns and high frequency words. In: COLINGACL (2006)

6. Deese, J.: The associative structure of some common English adjectives. Journal of Verbal Learning and Verbal Behavior 3(5) (1964)
7. Esuli, A., Sebastiani, F.: Sentiwordnet: A publicly available lexical resource for opinion mining. In: Proceedings of LREC (2006)
8. Grefenstette, G., Qu, Y., Evans, D.A., Shanahan, J.G.: Validating the Coverage of Lexical Resources for Affect Analysis and Automatically Classifying New Words along Semantic Axes. Springer. Netherlands (2006)
9. Hatzivassiloglou, V., McKeown, K.R.: Predicting the semantic orientation of adjectives. In: Proceedings of the eighth EACL conference. pp. 174–181 (1997)
10. Kamps, J., Marx, M., Mokken, R.J., Rijke, M.D.: Using wordnet to measure semantic orientations of adjectives. In: The proceedings of LREC. vol. IV, pp. 1115–1118 (2004)
11. Novak, P.K., Lavrač, N., Webb, G.I.: Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. J. Mach. Learn. Res. 10, 377–403 (June 2009)
12. Osgood, C.E., Suci, G.J., Tannenbaum, P.H.: The Measurement of Meaning. University of Illinois Press (1967)
13. Przepiórkowski, A.: The IPI PAN Corpus: Preliminary version. IPI PAN (2004)
14. Przepiórkowski, A., Woliñski, M.: The unbearable lightness of tagging: A case study in morphosyntactic tagging of polish. In: The Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03), EACL 2003 (2003)
15. Riloff, E., Wiebe, J., Wilson, T.: Learning subjective nouns using extraction pattern bootstrapping. In: Proceeings of the Seventh CoNLL conference at HLT-NAACL. pp. 25–32 (2003)
16. Turney, P., Littman, M.: Measuring praise and criticism: Inference of semantic orientation from association. ACM Transactions on Information Systems 21, 315–346 (2003)
17. Whitelaw, C., Navendu, G., Argamon, S.: Learning subjective language. Computational Linguistics (2005)
18. Wiebe, J.: Learning subjective adjectives from corpora. In: AAAI-00 Proceedings. pp. 735–740 (2000)

ALEKSANDER WAWER
INSTITUTE OF COMPUTER SCIENCE,
POLISH ACADEMY OF SCIENCE,
UL. JANA KAZIMIERZA 5, 01-238 WARSZAWA, POLAND
E-MAIL: <AXW@IPIPAN.WAW.PL>

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 25–39
RECEIVED 31/10/11 ACCEPTED 09/12/11 FINAL 13/06/12

Hindi Subjective Lexicon Generation using WordNet Graph Traversal

PIYUSH ARORA, AKSHAT BAKLIWAL, AND VASUDEVA VARMA

International Institute of Information Techonology, India

ABSTRACT

With the induction of UTF-8 unicode standards, web content in Hindi language is increasing at a rapid pace. There is a great opportunity to mine the content and get insight of sentiments and opinions expressed by people and various communities. In this paper, we present a graph based method to build a subjective lexicon for Hindi language, using WordNet as a resource. Our method takes a pre-annotated seed list and expands this list into a full lexicon using synonym and antonym relations. We show two different evaluation strategies to validate the Hindi Lexicon built. Main contribution of our work 1) Developing a Subjective lexicon of adjectives using Hindi WordNet. 2) Developing an annotated corpora of Hindi reviews.

KEYWORDS: *Hindi Language, Sentiment Analysis, Adjective Polarity Lexicon, WordNet, Graph Traversal*

1 INTRODUCTION

In past 8–10 years, we have seen an enormous increase in web content in Hindi language. This information is important to be mined for the use of/by researchers, industries and government(s). A large number of advertising industries and recommendation systems work on understanding the people likings and tastes from this content. Most of the earlier work targets sentiment and opinion analysis in resource rich languages like English. Our work addresses the problem of identifying sentiments and opinions from user generated content in Hindi and builds a model (Subjective Lexicon) using Hindi WordNet.

Hindi language has approx. 409 million native speakers as in 1999[1] and with unicode (UTF-8) standards for Hindi introduced, web pages catering Hindi is increasing on a rapid pace. There are many websites which cater information in Hindi[2] and to the best of our knowledge there are very few works [1, 2] in this field for Indian languages. This part of the web hasn't been explored much in the direction of sentiment and opinion analysis.

In this paper, we present a method of building a subjective lexicon for Hindi language with dependency only on WordNet and a small pre-annotated seed list. Using WordNet and simple graph traversal method we construct the subjectivity lexicon. In our method, initially a small seed list of words is decided along with their polarity. Using WordNet this seed list is populated based on the synonyms and antonyms of the words in the list. Here, we make an assumption that synonyms possess similar polarity and antonyms show opposite polarity.

The road map for rest of the paper is as follows: Existing related works are presented in Section 2. Section 3 presents a comprehensive view of the approach proposed in this research work. Section 4 gives details about the lexicon generated using this proposed method for Hindi language. Section 5 gives a description of product review dataset for Hindi language. In Section 6, we describe the different methods of evaluation used in this research. In Section 7, we discuss the results and limitations of this system. Section 8 presents the conclusions along with directions for our future work.

2 Related Work

Research in the field of sentiment analysis is done at various levels: Document Level [3, 4], Sentence Level [5–9] and Word or Phrase Level [10, 11].

In 1966, IBM developed the General Inquirer system [12], which marked the beginning of sentiment extraction from plain text. This system was termed as content analysis research problem in behavior science and comprised of 11789 words, with each word having at least one instance.

[1]en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

[2]hindiblogs.org, hindigear.com, bbc.co.uk/hindi

In 1998, the authors of [13] developed a method to predict semantic orientation of adjectives. Their idea consisted in predicting the semantic orientation of adjectives based on the nature of conjunctive joining the two adjectives. A log-linear regression model uses these constraints to predict whether conjoined adjectives are of same or different orientations, achieving 82% accuracy in this task when each conjunction is considered independently.

In 2002, Turney [4] extended the work [13] to other POS-tags. Turney used adverbs and nouns along with adjectives for performing opinion classification on reviews. He achieved 84% accuracy on automobile review classification compared to 66% on movie reviews.

For English, a good amount of work is done in the lines of generating subjective lexicon. SentiWordNet [14, 15] was developed in year 2006 by Esuli and Sebastiani. It contains four Part-of-Speech tags namely adjectives, adverbs, verbs and nouns with ~2 million words out of which 3% are adjectives. Each word is assigned three scores positive, negative and objective (Equation 1). SentiWordNet was built using WordNet and a ternary classifier. Their classifier is based on "bag of synset" model which uses manually disambiguated glosses available from the Princeton WordNet Gloss Corpus.

$$positive\ score + negative\ score + objective\ score = 1. \qquad (1)$$

Banea *et. al.* [16] proposed a bootstrapping method for building subjective lexicon for under-resourced languages. Their method build a subjective lexicon using a small seed list (60 words), an online dictionary (Romanian Dictionary) and a small annotated corpora. They used word level similarity (LSA and PMI) to filter words. In their bootstrapping method the initial seed list was manually selected and contained 60 words, which were evenly distributed among adjectives, adverbs, nouns, and verbs.

Kamps *et. al.* [17] tried to determine sentiments of adjectives in WordNet. In this work, they divided adjectives into four major categories and used base words (to measure relative distance) depending on the category. For category Evaluative their base words were "good" and "bad", for category Activity their base words were "active" and "passive", etc. The polarity orientation of a word 'w' belongs to range [-1,1], -1 for words on bad side and 1 for words on good side. Based on this method, they populated a total of 1608 words in all four categories with avg. correctness of 67.18% for English.

Kim and Hovy [18] proposed a method of identifying and analyzing judgement opinions. This was a four step process in which first step was recognizing the opinion. For identifying the opinion they introduced an algorithm to classify a word as positive, negative or objective which was based on WordNet.They made an assumption which was to add synonyms of a word with the same polarity as the source word. To avoid words with multiple meaning (dual nature) they applied a method to identify closeness of a word to each category (positive, negative, objective). For their proposed method to give high recall the initial seed list should be large enough and with wide variety of words.

Rao and Ravichandran [19] presented an extensive study on the problem of detecting polarity of words. They considered bi-polar classification of words i.e. a word can be either positive or negative. They performed semi-supervised label propagation in graph for polarity detection of words. Each of these words represent a node in the graph whose polarity is to be determined. They focused on three languages mainly English, French and Hindi but claim that their work can be extended to any other language for which WordNet is available.

As far as Indian Languages are concerned, we can see small amount of work done in Hindi and Bengali. Das and Bandhopadyay [1] developed SentiWordnet for the Bengali language. They applied word level lexical-transfer technique to each entry in English SentiWordNet using an English-Bengali Dictionary to obtain a Bengali SentiWordNet. This process resulted in 35,805 Bengali entries. In [20], authors devised four strategies to predict the sentiment of a word. First approach, an interactive game which in turn annotated the words with their polarity. Second approach, using Bi-Lingual dictionary for English and Indian Languages. Third approach, wordnet expansion using synonym and antonym relations, but their article missed the approach they followed for this expansion. Fourth approach, learning from pre-annotated corpora.

Joshi *et. al.*[2] created H-SWN (Hindi-SentiWordNet) using two lexical resources namely English SentiWordNet and English-Hindi WordNet Linking [21]. Using WordNet linking they replaced words in English SentiWordNet with equivalent Hindi words to get H-SWN.

Our work is directed towards the Hindi Language. It is related to works by Kim and Hovy [18] and Rao and Ravichandran [19]. Kim and Hovy restricted their assumption to synonyms; we extend the relation to antonyms. Rao and Ravichandran performed bi-polar classification; we extend it to a third level: objectivity. In this work, we use Hindi WordNet [22] to obtain the polarity of adjectives for Hindi Subjective Lexicon.

3 Algorithm

Our algorithm makes a hypothesis of traversing WordNet like a graph where every word in WordNet is imagined as a node in graph. This graph will be an undirected graph and will be highly connected but not fully connected. In this graph, nodes are connected to each other on synonym and antonym relations. Kim and Hovy, 2006 [18] made a simple assumption that synonyms carry the same sentiment as the word. We extend his assumption to antonyms, we assume antonyms carry opposite polarity. Each node will be connected to many other nodes i.e. each node will have many in-links and many out-links. This graph has three basic connected components (positive, negative and objective). Words can be connected using two kinds of edges:

1. Simple Edge: An edge which connects two words in the same (different) domain and represents a synonym (antonym) relation with a given condition that each word should belong to non overlapping region.
2. Cross Edge: An edge which connects two words based on synonym (antonym) relation and atleast one among these words lies in the overlapping region.

Cross Edges among these connected components produce words which have ambiguous (dual) nature.

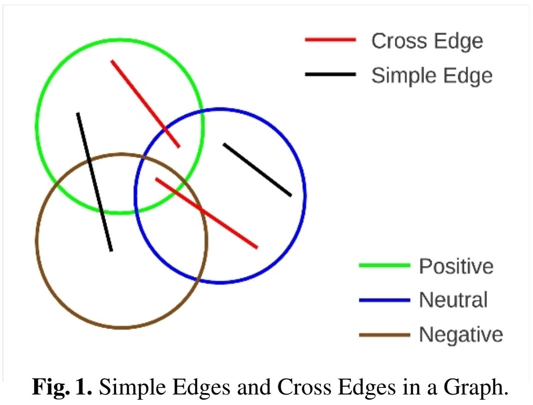


Fig. 1. Simple Edges and Cross Edges in a Graph.

Fig. 1 explains Simple Edges and Cross Edges pictorially. In Fig. 1 each circle represents a connected component and overlapping zone contains words which are ambiguous by nature.

Here, we use Hindi WordNet and a list of words (seed list) which is pre-annotated based on polarity. Our seed list contains 15 objective, 15 positive and 15 negative words. Each word in this seed list will be expanded on synonym and antonym relations. We can consider this expansion as a Breadth First Traversal of a graph. In this traversal method all the siblings (nodes at any depth d) are expanded before any node at the next level (depth d+1) is expanded. We make use of queue data structure to maintain the order in which the nodes (words) are introduced or expanded. This method helps us to ensure that each node is expanded only once and all the synonyms of a word are traversed at the same time.

In our method we have 2 lists, one is temporary and the other is final list. The initial seed list which contains 45 words is copied to temporary seed list with the polarity. Now every time we get a word (a structure which contains a pair of seed and polarity) from the temporary seed list by de-queuing it from the list, we check for this word if it exists in the final seed list or not. If this word is in the final seed list then we don't populate this word further, we just add the current polarity of this word to the polarity in the final list. But if this word is not in the final list, we do three things

1. Add this word to the final list with the current polarity
2. Find out all the synonyms of this word and en-queue them in the temporary seed list with the polarity same as the source word.
3. Find out all the antonyms of this word and en-queue them in temporary seed list with opposite polarity. (P -> N, O -> O, N -> P).

We continue this process till all the words in the temporary seed list are explored or in other words till the temporary seed list becomes empty. When the temporary seed list becomes empty the final seed list contains adjectives and against each adjective we have string of P's, N's and O's. Based on this we decide the final polarity of the word. Say for a word 'x' in the final seed list we have string 's' made of P's, N's and O's.

$$\begin{aligned}
Length\,of\,string\,(s) &= Len \\
Number\,of\,P's\,in\,s &= nP \\
Number\,of\,N's\,in\,s &= nN \\
Number\,of\,O's\,in\,s &= nO \\
Positive\,polarity\,of\,x &= nP/Len \\
Negative\,polarity\,of\,x &= nN/Len \\
Objective\,polarity\,of\,x &= nO/len
\end{aligned} \tag{2}$$

For a pseudocode, refer to Algorithm 1.

4 HINDI SUBJECTIVE LEXICON

Lexicon[3] built using the above mentioned approach for Hindi language contains 8048 words in all. Out of 8048 words 2521 are positive, 3060 are negative and 2467 are neutral. For each word, our lexicon provides three scores: positive, negative and objective. The sum of these scores amounts to 1.

$$(Positive + Negative + Objective)Score = 1 \qquad (3)$$

We validate this lexicon by two different methods which are explained in Section 6.

5 PRODUCT REVIEW DATA

This dataset[3] was translated from English to Hindi using Google[4] translate[5]. We translated pre-annotated Amazon product reviews [23] of length $\leq$ 25. We took this threshold of 25 words in order to avoid (reduce) translations errors. After translating the product reviews we asked human judges to manually validate the translation. Table 1 summarizes the data (reviews) generated by translation.

Table 1. Product Review Data Summary

Total Positive Reviews	900
Manually Corrected Reviews	350
Total Negative Reviews	900
Manually Corrected Reviews	350
Total Annotated Reviews	350 + 350

[3]This Resource is in the initial stage of development and is avaliable for non-commercial and research usage on request. Request should be made to any of the authors.

[4]http://translate.google.com/

[5]We made an assumption that while translation sentiment bearing words are translated correctly without any loss or modification of sentiments.

Algorithm 1 Algorithm for Populating SeedList using WordNet to generate Subjective Lexicon

```
1: $InitialSeedList = \{45\ words\}\ (15 \times objective, positive, negative)$
2: $//\ Each\ word\ is\ a\ structure\ which\ contains\ a\ pair:\ Seed,\ Polarity$
3: $FinalSeedList = \{\}$
4: $TempSeedList = \{\}$
5: $TempSeedList = InitialSeedList$
6: while $TempSeedList \neq EmptyList$ do
7:     $Word = TempSeedList.pop()\ //\ pop\ the\ first\ word\ out\ of\ the\ list$
8:     $Seed = Word[0]$
9:     $Polarity = Word[1]$
10:    if $Seed \in$ FinalSeedList then
11:        $FinalSeedList[Seed] = FinalSeedList[Seed] + Polarity$
12:    else
13:        FinalSeedList[Seed]=Polarity
14:        SynonymSet = All the synonyms of Seed
15:        AntonymSet = All the antonyms of Seed
16:        for all $synonyms \in$SynonymSet do
17:            $TempSeedList.append(synonym : Polarity)$
18:            $//\ Polarity\ will\ be\ P/N/O$
19:        end for
20:        for all $antonyms \in$AntonymSet do
21:            $TempSeedList.append(antonym : OppPolarity)$
22:            $//\ OppPolarity\ will\ be\ P\ if\ Seed\ has\ Polarity\ N$
23:            $//\ OppPolarity\ will\ be\ N\ if\ Seed\ has\ Polarity\ P$
24:            $//\ OppPolarity\ will\ be\ O\ if\ Seed\ has\ Polarity\ O$
25:        end for
26:    end if
27: end while
28: $//\ Against\ Each\ adjective\ in\ the\ FinalSeedList\ we\ have\ a\ string$
29: $//\ of\ P's,\ N's,\ and\ O's,\ which\ contains\ the\ polarity\ of\ that\ word$
30: for all $adjectives \in$FinalSeedList do
31:    $S = FinalSeedList[i]\ //\ Here\ i\ is\ an\ adjective\ and\ S\ is\ the\ string$
32:                                 $//\ of\ polarity\ for\ that\ adjective$
33:    $nP = Number\ of\ P's\ in\ S$
34:    $nN = Number\ of\ N's\ in\ S$
35:    $nO = Number\ of\ O's\ in\ S$
36:    $Len = length\ of\ S\ //\ Note:\ nP + nN + nO = Len$
37:    $PositivePolarity = nP/Len$
38:    $NegativePolarity = nN/Len$
39:    $Objectivity = nO/Len$
40:    $//Note:\ PositivePolarity + NegativePolarity + Objectivity = 1$
41: end for
```

6 Evaluation

One of the major task while proposing a new method is evaluation. In these kind of systems we mainly evaluate by human judgement or by classifying some pre-annotated text. These are a few methods which are commonly used for validation.

1. Human Judgement: This method is usually opted for languages which are scarce resource languages. In this method, some manual annotators are appointed whose task is to annotate the lexicon generated and later, taking the majority vote of annotators the system generated lexicon is validated.
2. Classification: In this method of evaluation, we classify pre-annotated reviews/blogs using our system generated lexicon and find precision, recall, F1 Scores, etc to show the correctness. This strategy is generally used for resource rich languages or for those languages for which we have pre-annotated data.
3. Validating Against Existing Resources: In this strategy of evaluation, we find the accordance of lexicon generated using our approach with a lexicon which is already proposed and accepted by the research community. This strategy of evaluation is used for languages which are resource rich.

Subsequent sub-sections explain two methods which we used to evaluate the lexicons generated by our system.

6.1 *Human Judgement*

In this method of evaluation, we hired five manual annotators[6] who are language experts in Hindi. We asked each annotator to tag the words generated by our system on the scale of 3 (negative:-1, neutral:0, positive:1). After getting the list annotated by all the annotators, we had five votes for each word and we took the majority call. Table 2 reports accordance of Hindi lexicon generated using our system with manual annotation.

Reason behind low mutual agreement among the annotators is that many words in Hindi show ambiguous nature. Their polarity depends on the sense in which they are used. This ambiguous nature is highlighted in Fig. 2.

[6]None of the authors were annotators for this task.

Table 2. Results for Manual Agreement for Hindi Lexicon

Mutual agreement among the annotators	**70.48%**
Agreement of each annotator with our lexicon	
Annotator 1	66.08%
Annotator 2	64.01%
Annotator 3	**68.45%**
Annotator 4	66.70%
Annotator 5	68.34%
Overall Agreement of our lexicon with the annotators	**68.80%**

6.2 *Review Classification*

For this evaluation strategy, we performed classification on product review dataset described in Section 5. On this data, we performed unigram presence and simple scoring method classification. In unigram presence method, we count unigrams of positive, negative and objective polarity and assigned the polarity for which the count was highest. In simple scoring method, we summed the positive, negative and objective scores of each adjective and assigned the polarity of the dominant score. From every review we identified adjective and scored those adjectives using our lexicon. If an adjective was missing from our lexicon we considered the stemmed variant[7] [8] of that word for scoring. In addition to stemming we also performed negation handling. We identified the words with tag "NEG" using sliding window of 6 words and swapped the polarity (positive and negative) of adjectives in this range. Our sliding window, looked upto 3 words in both the directions (left and right) of this word. Table 3 reports the results of classification.

7 DISCUSSION

Results in Table 3 highlights the point that our scoring method performs better than mere presence counting of the adjectives. The following example shows how the presence counting failed in classification while the scores assigned to each word by our method correctly classified the review. "बहुत ही प्रतिभाशाली संगीतकारो का एक बिलकुल बेकार प्रयास".

[7]We used the stemmer which is bundled with Hindi WordNet API 1.2

[8]cfilt.iitb.ac.in/wordnet/webhwn/index.php

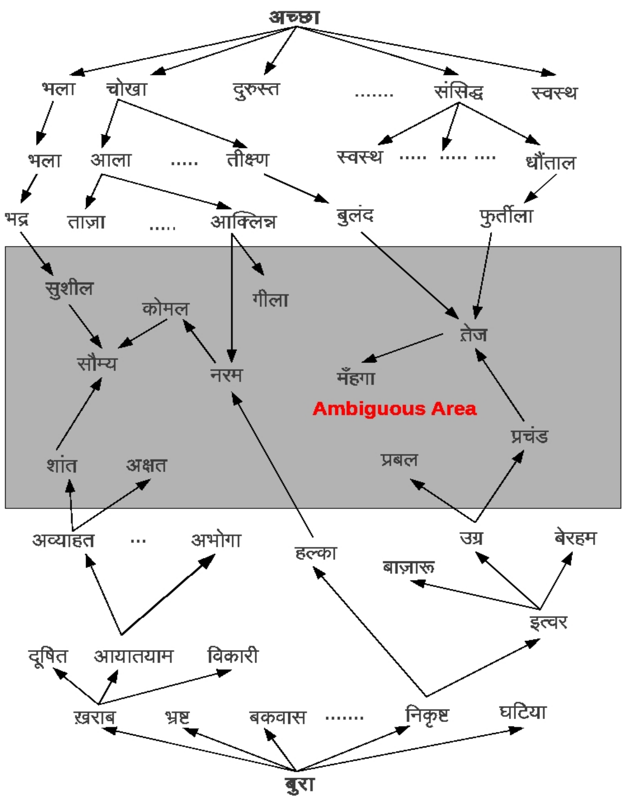


Fig. 2. The Graph Traversal of words in Hindi WordNet. The dark portion shows ambiguous words.

Adjectives in this example are "प्रतिभाशाली" (positive) and "बेकार" (negative). If we account for presence of adjectives for classification, this review becomes neutral. However, scores generated using our system $(+0.75, -0.0, 0.25)$ for "प्रतिभाशाली" and $(+0.0, -1.0, 0.0)$ for "बेकार" with overall score for the review as $-0.25([0.75 - 0.0] + [0.0 - 1.0])$, classify the review correctly (as negative).

Using the proposed strategy for negation handling, results in Table 3 show $\sim 3\%$ improvement in classification of reviews. We proposed the use of stemmer to identify the root word for adjectives which were

Table 3. Results for Product Review Classification using Lexicon generated by our approach

Method	Accuracy
Adjective Presence	
Baseline	65.50
Baseline + Negation Handling	68.67
Baseline + Stem	67.17
Baseline + Stem + Negation Handling	70.80
Adjective Scoring	
Baseline	67.33
Baseline + Negation Handling	70.00
Baseline + Stem	71.00
Baseline + Stem + Negation Handling	**74.10**

present in the review but went missing from our lexicon. Stemming also showed an improvement of ∼3% in classification of reviews. Table 4 lists a few mapping of words to their stemmed form.

Table 4. Words and their stemmed (root) words

Word(s)	Stemmed Word
छोटे	छोटा
अच्छी, अच्छे	अच्छा
बड़ी	बड़ा
हल्के	हल्का
लंबे	लंबा

There are a few limitations and issues with the current version of algorithm proposed above.

– To handle adjectives which were present in reviews and were missing from our lexicon, we performed stemming. If an adjective was missing from our lexicon we stemmed the adjective to get its root word. Instead of using a stemmer if a morph is used, then we expect results to improve.
– The current version of this algorithm does not perform Word Sense Disambiguation (WSD).
– Scope of the system proposed above is dependent on the initial seed list used to populate the WordNet. If we choose the seed list in a

careful manner with the help of linguistic experts, the results and scope of the Lexicon thus generated would be better.

8 Conclusion and Future Work

We proposed a graph based method to generate the subjectivity lexicon for Hindi and explored how the synonym and antonym relations can be exploited using graph traversal. Our method is language independent and just uses only one resource (WordNet) for Lexicon generation. As a part of this research, we worked on Hindi language. The lexicon generated using our proposed algorithm contains 8048 words and it achieved 74% accuracy on classification of reviews and 69% in agreement with human annotators for Hindi.

In future, this work can be extended to incorporate Word Sense Disambiguation (WSD) to emphasize more on senses of a word. Another extension can be morphological variants which could result in better accuracy for words which have dual nature. We experimented only with adjectives and this work can be extended for other parts of speech.

References

1. Das, A., Bandyopadhyay, S. In: SentiWordNet for Bangla. Linguistic Data Consortium for Indian Languages (LDC-IL) (2010)
2. Joshi, A., R, B.A., Bhattacharyya, P.: A fall-back strategy for sentiment analysis in Hindi: a case study (2010)
3. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP). (2002) 79–86
4. Turney, P.: Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews (2002)
5. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: KDD. (2004) 168–177
6. Intelligent, T.W., Wilson, T.: Annotating opinions in the World Press. In: In SIGdial-03. (2003) 13–22
7. min Kim, S.: Determining the sentiment of opinions. In: In Proceedings of COLING. (2004) 1367–1373
8. Wiebe, J.M., Bruce, R.F., O'Hara, T.P.: Development and use of a gold-standard data set for subjectivity classifications. In: Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics. ACL '99, Stroudsburg, PA, USA, Association for Computational Linguistics (1999) 246–253

9. Yu, H., Hatzivassiloglou, V.: Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In: Proceedings of the 2003 conference on Empirical methods in natural language processing. EMNLP '03, Stroudsburg, PA, USA, Association for Computational Linguistics (2003) 129–136
10. Agarwal, A., Biadsy, F., Mckeown, K.R.: Contextual phrase-level polarity analysis using lexical affect scoring and syntactic n-grams (2009)
11. Wilson, T.: Recognizing contextual polarity in phrase-level sentiment analysis. In: In Proceedings of HLT-EMNLP. (2005) 347–354
12. Stone, P.J., Dunphy, D.C., Smith, M.S., Ogilvie, D.M.: The General Inquirer: A Computer Approach to Content Analysis. MIT Press (1966)
13. Hatzivassiloglou, V., McKeown, K.R.: Predicting the semantic orientation of adjectives. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics. ACL '98, Stroudsburg, PA, USA, Association for Computational Linguistics (1997) 174–181
14. Baccianella, S., Esuli, A., Sebastiani, F.: SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In Chair), N.C.C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., Tapias, D., eds.: Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10), Valletta, Malta, European Language Resources Association (ELRA) (may 2010)
15. Esuli, A., Sebastiani, F.: SENTIWORDNET: A publicly available lexical resource for opinion mining. In: In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC-06). (2006) 417–422
16. Carmen Banea, R.M., Wiebe, J.: A bootstrapping method for building subjectivity lexicons for languages with scarce resources. In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odjik, J., Piperidis, S., Tapias, D., eds.: Proceedings of the Sixth International Language Resources and Evaluation (LREC'08), Marrakech, Morocco, European Language Resources Association (ELRA) (may 2008) http://www.lrec-conf.org/proceedings/lrec2008/.
17. Kamps, J., Marx, M., Mokken, R.J., Rijke, M.D.: Using wordnet to measure semantic orientation of adjectives. In: National Institute for. (2004) 1115–1118
18. min Kim, S., Hovy, E.: Identifying and analyzing judgment opinions. In: Proceedings of HLT/NAACL-2006. (2006) 200–207
19. Rao, D., Ravichandran, D.: Semi-supervised polarity lexicon induction. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics. EACL '09, Stroudsburg, PA, USA, Association for Computational Linguistics (2009) 675–682
20. Das, A., Bandyopadhyay, S. In: SentiWordNet for Indian Languages. Asian Federation of NLP (AFNLP) (2010)
21. Karthikeyan, A.: Hindi English WordNet linkage. (2010)

22. Dipak Narayan, Debasri Chakrabarti, P.P., Bhattacharyya, P.: An experience in building the Indo WordNet—a WordNet for Hindi. In: First International Conference on Global WordNet. (2002)
23. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In: ACL. (2007)

PIYUSH ARORA
SEARCH AND INFORMATION EXTRACTION LAB, LTRC,
INTERNATIONAL INSTITUTE OF INFORMATION TECHONOLOGY,
HYDERABAD, INDIA
E-MAIL: <PIYUSH.ARORA@RESEARCH.IIIT.AC.IN>

AKSHAT BAKLIWAL
SEARCH AND INFORMATION EXTRACTION LAB, LTRC,
INTERNATIONAL INSTITUTE OF INFORMATION TECHONOLOGY,
HYDERABAD, INDIA
E-MAIL: <AKSHAT.BAKLIWAL@RESEARCH.IIIT.AC.IN>

VASUDEVA VARMA
SEARCH AND INFORMATION EXTRACTION LAB, LTRC,
INTERNATIONAL INSTITUTE OF INFORMATION TECHONOLOGY,
HYDERABAD, INDIA
E-MAIL: <VV@IIIT.AC.IN>

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 41–55
RECEIVED 18/10/11 ACCEPTED 09/12/11 FINAL 15/06/12

Summarizing Public Opinions in Tweets

NIBIR NAYAN BORA

KIIT University, India

ABSTRACT

The objective of Sentiment Analysis is to identify any clue of positive or negative emotions in a piece of text reflective of the authors opinions on a subject. When performed on large aggregations of user generated content, Sentiment Analysis may be helpful in extracting public opinions. We use Twitter for this purpose and build a classifier which classifies a set of tweets. Often, Machine Learning techniques are applied to Sentiment Classification, which requires a labeled training set of considerable size. We introduce the approach of using words with sentiment value as noisy label in a distant supervised learning environment. We created a training set of such Tweets and used it to train a Naive Bayes Classifier. We test the accuracy of our classifier using a hand labeled training set. Finally, we check if applying a combination of minimum word frequency threshold and Categorical Proportional Difference as the Feature Selection method enhances the accuracy.

KEYWORDS: *Sentiment classification, supervised learning, Twitter*

1 INTRODUCTION

A simple search for “Kindle 2”[1] by Google returns over 700 million results, mostly consisting of product descriptions and user reviews. Any user would be overwhelmed with such huge amounts of content, making

[1] http://amazon.com/kindle

it almost impossible to scan through all of it to find what is crucial. With the rise in the number of social networking, blogging and microblogging websites, and the ease with which a user can submit information on these sites, the internet today has become an unmanageable accumulation of user generated content. Users, differing in social, political and cultural background, share their personal opinion on various subjects, discuss current issues and write about events in their life. What is much needed in such a situation is a system which can extract significant information from such large sets of data; that is, give us more aggregated results.

Sentiment Analysis or Opinion Mining, the study of computationally determining whether a given piece of text is indicative of positive or negative sentiment, is one way of summarizing large aggregations of text. Sentiment classification, when performed on user generated textual content, find many applications. Knowing how users feel about a product or service can help in business decisions for corporates. Political parties and social organizations can collect feedback about their programs and legislation. Artists, musicians and other entertainment icons can reach out to their fans and assess the quality of their work. Broadly, it can serve as an automatic polling system, relieving any manual intervention.

Twitter[2], one of the most popular microblogging websites among the internet community, serves as a good platform for sentiment analysis because of its large user base from different sociocultural zones. Users update short messages (called Tweets) within a 140 character limit on Twitter. It contains huge number of tweets, with millions added each day, which can be easily collected through its APIs (Application Program Interface), making it convenient to build a large training set. Usually, machine learning techniques are applied to sentiment classification, in which a classifier is required to be trained on a labeled training set. This is called supervised learning. However, owing to its nature and the number of tweets that can be collected, it is a challenging task to manually label a training set of such magnitude.

In this paper, we introduce the novel approach of labeling large sets of tweets using sentiment suggestive words as noisy labels. Using this method we create an annotated dataset of 1.5 million tweets, which is used to train a machine learning classifier. The accuracy of the classifier is evaluated and compared to previous similar techniques. Further, a combination of minimum word frequency threshold and categorical proportional difference is discussed as the Feature Selection method enhances

[2] http://twitter.com

the accuracy of the classifier. We also show the aggregated results of our classifier when tried on a few search queries.

The rest of the paper is organized as follows. Related works are discussed in section 2. Our approach of labeling and sentiment classification is described in section 3. The results and discussions of the experiment are presented in section 4. Finally we conclude about our work in section 5.

2 Related Work

A number of prior work has been directed towards sentiment classification on blog posts [1], reviews [2,3] and tweets [4,5,6]. However, blog posts and reviews differ from tweets, first, because of their size, and second, because tweets follow an entirely different language model, which does not ensure correctness or consistency of grammar and spelling. Because of Twitter's 140 character limit, tweets are more likely to contain spelling and grammatical errors. Users, sometimes, strips off letters from words (e.g. *whr*) to fit their message within this limit. Twitter users also use a lot of Internet slang and abbreviations (e.g. *OMG*, *ASAP*, etc.) and emoticons. Emoticons, also known as *smileys* are glyphs constructed using the characters available on a standard keyboard, representing a facial expression of an emotion.

Various approaches of sentiment classification has be studied in the past years. Initially focused on lexicon based methods (Das and Chen [7]) and unsupervised hand made algorithms (Turney [8]), the study later moved on to supervised machine learning techniques (Pang et al. [2]) and unsupervised clustering algorithms (Yessenov and Misailović [3]). Recent researches in sentiment analysis has incorporated Natural Language Processing techniques as well as Feature Selection methods to further enhance the accuracy of the classifiers. While most of these researches were concerned with classification at document level, Mishne and de Rijke [1] analyzed global mood levels at an aggregate level. This is quite similar to our study, where we classify tweets at document level, but we discuss how such an classification could be applied to a set of tweets to determine aggregated results.

Our approach of using sentiment suggestive words is similar to the use of emoticons as noisy labels, first introduced by Read [9]. A noisy label is an element within the piece of text which provides information about the class to which the text belongs. This approach was later used by Go et al. [4], and Pak and Paroubek [5] on a Twitter corpus. In such an approach, a set of emoticons is manually classified as bearing positive or negative sentiment. The occurrence of any of these emoticons in a tweet causes it to be labeled as belonging to the corresponding class. Go et al. [4] created an annotated training set using this technique. However, using emoticons as noisy labels, prevents their use while classifying new tweets. We address this drawback by using sentiment suggestive words as noisy label and hence permitting the use of emoticons during classification. Duurkoop [6] used a similar approach, however, he used hashtags instead of emoticons as noisy label in his training data.

Feature selection, used in order to improve the performance of the classifier, is a method to select a portion of the feature set, generated by the trainer, which is most likely to serve in classification. Keefe and Koprinska [10] evaluated a range of feature selection methods using both Naive Bayes classifier and Support Vector Machine on a movie review corpus. In their study, they found the categorical proportional difference (CPD) to outperform other feature selection methods. CPD has also being studied by Simeon and Hilderman [11] in their study on text categorization. Yessenov and Misailović [3] considered various feature extraction and feature reduction methods in their model. They observed an increase in the performance of their classifier while considering only words that appear most frequently in the corpus. Pak and Paroubek [5] used a similar approach, an entropy based method, for feature selection. However, these feature selection methods haven't been studied on a twitter corpus. We evaluate CPD along with a minimum word frequency threshold as the feature selection method for our classifier.

3 The Experiment

3.1 *Our Approach*

Our study was directed towards building a classifier which could classify tweets at document level, i.e. each tweet individually. The classifier would then be utilized to analyze a collection of tweets to arrive at aggregated results. A training set of 1.5 million tweets was built by collecting tweets using the Twitter Search API[3]. This annotated training set was then used to train a Naive Bayes classifier, treating each tweet as a bag-of-unigrams feature. In the bag-of-unigrams model, each tweet is considered as an unordered collection of words, disregarding grammar

[3] https://dev.twitter.com

Table 1. List of positive and negative sentiment words

Positive sentiment words	Negative sentiment words
amazed, amused, attracted, cheerful, delighted, elated, excited, festive, funny, hilarious, joyful, lively, loving, overjoyed, passion, pleasant, pleased, pleasure, thrilled, wonderful	annoyed, ashamed, awful, defeated, depressed, disappointed, discouraged, displeased, embarrassed, furious, gloomy, greedy, guilty, hurt, lonely, mad, miserable, shocked, unhappy, upset

and their order of occurrence. We used words with sentiment value (e.g. *cheerful*, *shocked*, *disappointed* etc.) as noisy labels to automatically label the training set. The feature extraction process follows a series of cleaning and preprocessing steps on the tweet before tokenizing it. An attempt was made to increase the accuracy of the classifier by using two feature selection methods, namely, minimum word frequency threshold and categorical proportional difference. To test the accuracy of the classifier, a hand labeled test set was used.

3.2 *The Corpus*

A set of 40 words was prepared, each indicating certain sentiment value. Each of these words were then manually categorized as being positive or negative. Most of these words describe a certain mood or emotion, or are in the past tense verb form, most likely to be used by an author to express his or her feelings. e.g. *ashamed* in *"I was ashamed of what I did"*. This list has been intuitively selected and is in no way exhaustive. The complete list of positive and negative sentiment words, used in our experiments, is given in Table 1.

A tweet is labeled as positive if it contained any of the positive sentiment words, or negative if it contained any of the negative sentiment word. For example, a tweet containing the word 'thrilled' will be labeled as a positive tweet, whereas a tweet containing 'annoyed' will be labeled as a negative tweet. Table 2 shows a few tweets labeled in this manner. In disputable situations when a tweet contains both a positive and a negative sentiment word, the tweet is discarded.

The twitter search API allows to retrieve the most recent tweets based on a search query. However, it returns only 100 tweets per page and up to 15 such pages. The API also limits the number of requests per hour from an IP (Internet Protocol) address. Tweets were collected using the positive

Table 2. Example of a few labeled tweets. Last column shows the matched sentiment word.

Tweet	Label	Word
`Arrived in Basel. Brilliant sunny weather. I'll go to the botanical gardens first. It's a wonderful place to think and write.`	positive	`wonderful`
`I'm really unhappy with myself at this point and i'm seriously at a breaking point and i'm on the verge of relapse i'm trying so hard`	negative	`unhappy`
`Toy Story 3 is hilarious. I love the scene where Ken is modelling for Barbie! Fab stuff :-)`	positive	`hilarious`
`i should really eat something, but i'm just not a fan when it's this hot and miserable. :/`	negative	`miserable`

and negative sentiment words as the query term. Considering the hourly limits and allowing enough time for the requests to yield unique tweets, our system sent out a search request for each word every 30 minutes.

A large accumulation of tweets was built during the period from June 24, 2011 to July 5, 2011. All retweets were removed as they could cause an unwanted redundancy in the training set. Retweets are tweets that are posted by a user by copying another user's tweet. They are marked by the presence of the characters 'RT' in the tweet. The matching sentiment word found in each tweet was also removed. This was done so that the classifier is not trained with a bias for the set of sentiment words used for labeling. The tweets were then labeled according to the category in which the matching sentiment word falls. This way, a training set of 1.5 million (1,464,638, precisely) tweets consisting of 668,975 positive tweets and 795,661 negative tweets was created.

3.3 *The Naive Bayes Classifier*

Naive Bayes classifier is a probabilistic model which estimates the probability that a tweet belongs to a specific class (positive or negative class, in this experiment). Naive Bayes has been used because of its low computational overhead and ease of use, yet performing well in many Natural Language Processing tasks, as indicated by Lake [12]. In general, the probability that a tweet T, belongs to class C, can be calculated using Bayes Theorem, as follows

$$P(T|C) = \prod_i P(w_i|C) \tag{1}$$

where $P(w_i|C)$ is the probability of the feature w_i occurring in class C. Finally, the class C is assigned to the tweet T, whichever yields the maximum $P(C|T)$.

3.4 *Unigram Feature Extractor*

The following preprocessing steps were applied to the Tweets:

- Remove URLs (e.g. *http://bit.ly/aDkhG*) and user mentions. User mention is a way to tag a user in a tweet. It is represented by a '@' symbol followed by the username (e.g. *@nibirbora*).
- Replace emoticons with a token representing the emotion expressed by it. Table 3 is a list of few emoticons and their meanings.

Table 3. Emoticons and their meanings

Emoticons	Meaning
>:] :-) :) :o) :] :3 :c) :> =] 8) =) :} :^)	smile
>:D :-D :D 8-D 8D x-D xD X-D XD =-D =D =-3 =3	laugh
:'(;*(:_(	cry
>:[:-(:(:-c :c :-< :< :-[:[:{	frown
>;] ;-) ;) *-) *) ;-] ;] ;D	wink
>:o >:O :-O :O	surprise
D:< >:(>:-C >:C >:O D-:< >:-(:-@ :@ ;('_' D< :L	angry

- Tokenize words: split the tweet at spaces and punctuation marks.
- Remove stop words, e.g. *the*, *is*, *at*, *which*, *on*.
- Replace words with repeated letters. Users sometimes arbitrarily repeat certain letters in a word to put more emphasis on it (e.g. *happpppyyyyyy*). We replace a word with any letter occurring more than

twice with two words, one in which the repeated letter occures once and twice in the second. E.g., the word *happppppyyyyyy* will be replaced with four words: *hapy*, *hapyy*, *happy*, *happyy*.
- Stem words to their roots, so that grammatical inflections are removed. We use the Porter Stemming algorithm [13] for this purpose.

3.5 *Feature Selection*

We use Categorical Proportional Difference along with a minimum word frequency threshold as our feature selection method. This two step feature selection process is discussed next.

MINIMUM WORD FREQUENCY THRESHOLD. In the first step, all features with frequency below a minimum threshold frequency are remove from the feature set. The threshold is set as a percentage calculated on the maximum frequency of any feature in the feature set. The accuracy of the classifier is checked at various minimum threshold percentages.

CATEGORICAL PROPORTIONAL DIFFERENCE. Categorical Proportional Difference (CPD), a measure of how equal two numbers are, can be used to find the features that occur mostly in either one of positive or negative class of tweets. The positive and negative frequencies of a feature are used to calculate it's CPD, by an equation suggested by Keefe and Koprinska [10] as follows:

$$CPD = \frac{|Positive_f - Negative_f|}{Positive_f + Negative_f} \tag{2}$$

If a feature is prevalent in either positive tweets or in negative tweets then it's CPD will be close to one, whereas if it occurs almost evenly in both positive and negative tweets then its CPD will be close to zero. A high CPD indicates that the feature is worth considering for classification. For example if the word "wife" appears in exactly as many positive tweets as negative tweets then finding the word "wife" would not contribute in classifying new tweet and its CPD score will be zero. Conversely, if the word "birthday" appears in only positive tweets then finding the word "birthday" in a new tweet would give us a good clue that the tweet is positive, and it would have a CPD score of one.

CPD is used for feature selection by removing any features whose CPD score is less than some threshold value. The accuracy of the classifier was checked at various threshold value.

Table 4. Effect of feature selection on the feature set (initial feature size is 319,719).

CPD	Minimum word frequency threshold			
thresholds	1.000%	0.100%	0.010%	0.001%
0.000	1,357	6,616	27,487	319,719
0.125	883	4,553	21,042	296,861
0.250	533	2,902	15,417	287,159
0.375	258	1,643	10,409	270,971
0.500	128	903	6,919	26,615

Table 4 shows the effect of the feature selection process on the feature set. The values indicate the feature set size after applying feature selection. The corresponding accuracies for each of these values are presented later.

3.6 *Test Set*

To determine the accuracy of the classifier, a test set was created. A portion of the training set was randomly selecting, from which, tweets which do not suggest any positive or negative sentiment were manually removed. The remaining tweets were then hand labeled. The final test set consisted of 198 positive sentiment tweets and 204 negative sentiment tweets.

3.7 *Aggregate Classifier*

As pointed out previously, the document level classifier is eventually used to find aggregated results. This is done by feeding a set of tweets to the classifier, which then classifies each document separately, and calculates the number of documents marked as positive and negative as a percentage over the total number of tweets in the set.

4 Results & Discussion

The classifier showed an accuracy of 77.61%, on the hand labeled test set, when tested without any feature selection method. However, it reached a maximum accuracy of 83.33% when feature selection methods were incorporated. This value was achieved when the minimum word frequency threshold was set to 0.001% and the CPD threshold to 0.25, reducing the size of the feature set to 287,159, which is about 90% of the original (319,719 features). It was noticed that our feature selection method considerably increased the accuracy of the classifier. The accuracy at various minimum word frequency (MWF) thresholds and CPD thresholds are presented in Figure 1 and Table 5.

Clearly, the accuracy decreases gradually with an increase in the MWF threshold, whereas, over the range of CPD thresholds, it shows a peak at CPD value 0.25. Beyond this value the graph fall sharply, which could be attributed to the fact that more useful features are removed from the

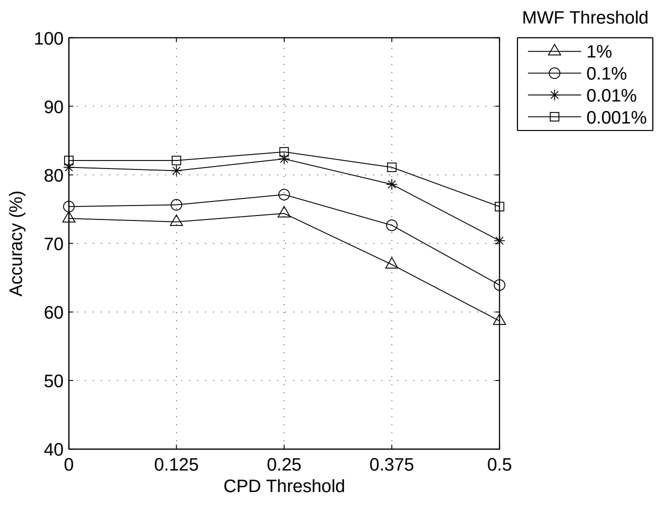


Fig. 1. Accuracy of the classifier at different minimum word frequency thresholds and CPD thresholds.

Table 5. Accuracy values (in percentage) at different minimum word frequency thresholds and CPD thresholds.

CPD	Minimum word frequency threshold			
thresholds	1.000%	0.100%	0.010%	0.001%
0.000	73.63	75.37	81.09	82.08
0.125	73.13	75.62	80.59	82.08
0.250	74.37	77.11	82.33	**83.33**
0.375	66.91	72.63	78.60	81.09
0.500	58.70	63.93	70.39	75.37

feature space. At lower CPD values (≤ 0.25), it was observed that the accuracy remained within a 10% range with change in the MWF threshold. Since considering features with frequency above a minimum threshold frequency did not contribute much to the accuracy of the classifier, we tested our classifier with CPD alone as the feature selection method. This yielded a maximum accuracy of 83.08%, almost similar to that which was achieved considering a minimum threshold frequency. Once again, this accuracy was achieved at the CPD threshold of 0.25, assuring it to be an effective choice for feature reduction. The results are shown in Figure 2 and Table 6.

The maximum accuracy of the classifier was similar to that reported by Pang et al. [2], i.e. 81% and Go et al. [4], i.e. 81.34%. However, it was less than that accounted in Pang & Lee's [14] later study, i.e. 86.4%, since we did not perform any subjectivity analysis. One notable benefit

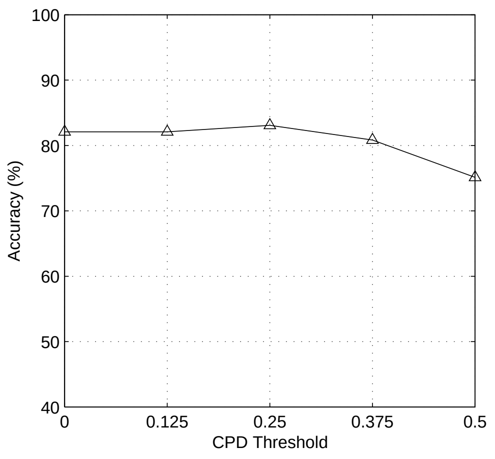


Fig. 2. Accuracy of the classifier at different CPD thresholds. (Without minimum word frequency threshold)

Table 6. Accuracy values (in %) at different CPD thresholds. (Without minimum word frequency threshold)

CPD thresholds	0.000	0.125	0.250	0.375	0.500
Accuracy	82.08	82.08	**83.08**	80.84	75.12

of our approach is that we consider emoticons in the classification process. Emoticons, which often are a strong indicator of sentiment in short texts, were ignored by Go et al. [4] while classifying tweets. It was observed that the best accuracy of our classifier is slightly higher than those reported by other studies involving Naive Bayes classification. This may be because our test set was biased towards our training set, owing to our method of test set generation.

While our accuracy matched that of Go et al. [4], who used emoticons as noisy labels, it was quite higher than that of Duurkoop [6] (around 70%) who used hashtags, indicating that sentiment suggestive words are more effective as noisy labels than hashtags.

Yessenov and Misailović [3] accounted for an increase in the performance of their classifier while considering only words that appear most frequently in the corpus. This is contrary to our observations where

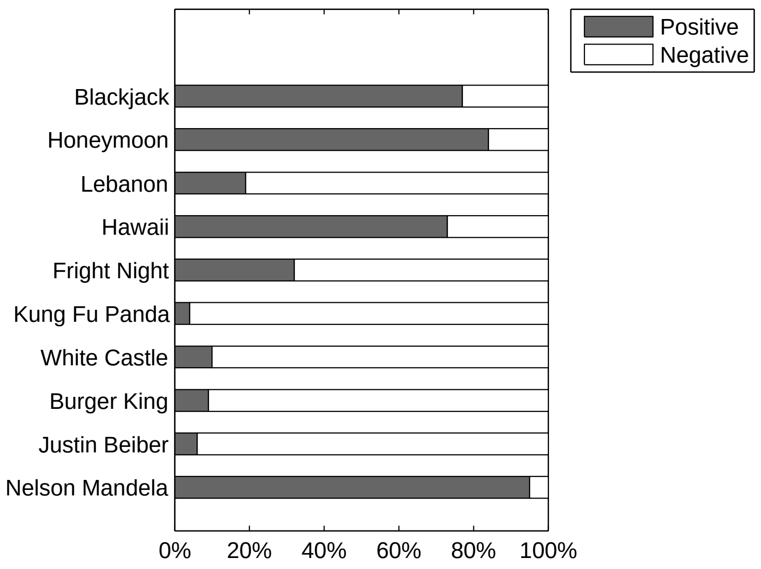


Fig. 3. Results of the classifier on a set of tweets. (Percentage positive and percentage negative)

Table 7. Results of the classifier on a set of tweets. (Values in number of tweets)

Query term	Positive	Negative	Total
Nelson Mandela	63	3	66
Justin Beiber	2	30	32
Burger King	5	49	54
White Castle	9	79	88
Kung Fu Panda	3	76	79
Fright Night	29	63	92
Hawaii	66	25	91
Lebanon	11	48	59
Honeymoon	76	15	91
Blackjack	44	13	57

we found that limiting the feature set with a minimum word frequency threshold does not contribute to increasing the accuracy of the classifier.

Figure 3 shows a few results of the aggregate classifier. A set of tweets related to the query terms shown in Table 7 were fed to the classifier. Table 7 also shows the number of total tweets analyzed and the number among them that were classified as positive and negative.

5 Conclusion

We built a sentiment classification tool which could accurately find the polarity (positive or negative) of a tweet, and can be used to analyze a collection of tweets to find aggregated results. We showed that sentiment suggestive words can be effectively used as noisy label for a Twitter corpus, and using this technique built a training set of 1.5 million tweets. This corpus was used to train a Naive Bayes classifier, who's accuracy was measured using a hand labeled test set. The maximum accuracy achieved was 83.33%, which is comparable to prior related studies. We also studied the use of a combination of minimum word frequency threshold and Categorical Proportional Difference as feature selection method. It was found that while CPD successfully increased the efficiency of the classifier (reaching a peak at CPD value 0.25), setting a minimum word frequency threshold did not.

Future work will include building a classification tool that can classify various emotions and not just identify positive and negative sentiment. Such a tool may be used to create a user interface which is reflective of the user's mood. We will also consider finding a way to eliminate tweets from the training set that does not indicate any sentiment value and also perform grammatical semantic analysis on the tweets to possibly increase the accuracy of classification.

ACKNOWLEDGMENT The author would like to thank Prachet Bhuyan of the School of Computer Engineering, KIIT University, Bhubaneswar, Orissa for his valuable comments and suggestions.

REFERENCES

1. Mishne, G., de Rijke, M.: Capturing global mood levels using blog posts. In: AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs, AAAI (2006) 145–152
2. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? sentiment classification using machine learning techniques. In: Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10. EMNLP '02, Stroudsburg, PA, USA, Association for Computational Linguistics (2002) 79–86
3. Yessenov, K., Misailović, S.: Sentiment analysis of movie review comments. 6.863 Spring 2009 final project, CSAIL, MIT (2009)
4. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. Processing (2009) 1–6
5. Pak, A., Paroubek, P.: Twitter as a corpus for sentiment analysis and opinion mining. In: Seventh International Conference on Language Resources and Evaluation, LREC 2010, La Valletta, Malta, May 19-21, 2010, Proceedings, Valletta, Malta, European Language Resources Association (ELRA) (2010) 1320–1326
6. Duurkoop, J.: Real-Time Happiness. Bachelor thesis, University of Amsterdam, Faculty of Science, Science Park 904, 1098 XH, Amsterdam (2010)
7. Das, S., Chen, M.: Yahoo! for amazon: Extracting market sentiment from stock message boards. In: In Asia Pacific Finance Association Annual Conf. (APFA). (2001)
8. Turney, P.D.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. ACL '02, Stroudsburg, PA, USA, Association for Computational Linguistics (2002) 417–424
9. Read, J.: Using emoticons to reduce dependency in machine learning techniques for sentiment classification. Proceedings of the ACL Student Research Workshop on ACL 05 **43**(June) (2005) 43
10. O'Keefe, T., Koprinska, I.: Feature selection and weighting methods in sentiment analysis. In: Proceedings of the 14th Australasian Document Computing Symposium. (2009)

11. Simeon, M., Hilderman, R.J.: Categorical proportional difference: A feature selection method for text categorization. In Roddick, J.F., Li, J., Christen, P., Kennedy, P.J., eds.: AusDM. Volume 87 of CRPIT., Australian Computer Society (2008) 201–208
12. Lake, T.: Twitter sentiment analysis. Technical report, Western Michigan University, Kalamazoo (2011)
13. Porter, M.: An algorithm for suffix stripping. Program **14**(3) (1980) 130–137
14. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Scott, D., Daelemans, W., Walker, M.A., eds.: ACL, ACL (2004) 271–278

NIBIR NAYAN BORA
SCHOOL OF COMPUTER ENGINEERING,
KIIT UNIVERSITY,
BHUBANESWAR, ORISSA, INDIA
E-MAIL: <NIBIRBORA@GMAIL.COM>

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 57–70
RECEIVED 17/11/11 ACCEPTED 09/12/11 FINAL 15/06/12

A Corpus-Based Method for Product Feature Ranking for Interactive Question Answering Systems

NATALIA KONSTANTINOVA, CONSTANTIN ORĂSAN, AND PEDRO PAULO BALAGE

University of Wolverhampton, UK

ABSTRACT

At times choosing a product can be a difficult task due to the fact that customers need to consider many features before they can reach a decision. Interactive question answering (IQA) systems can help customers in this process, by answering questions about products and initiating a dialogue with the customer when their needs are not clearly defined. For this purpose we propose a corpus-based method for weighting the importance of product features depending on how likely they are to be of interest for a user. By using this method, we hope that users can select the desired product in an optimal way. For the experiments a corpus of user reviews is used, the assumption being that the features mentioned in a review are probably more important for a person who is likely to purchase a product. To improve the method, a sentiment classification system is also employed to distinguish between features mentioned in positive and negative contexts. Evaluation shows that the ranking method that incorporates this information is one of the best performing ones.

KEYWORDS: *Wikipedia, sentiment analysis, interactive question answering, knowledge acquisition*

1 INTRODUCTION

Every day millions of people are confronted with a task of choosing a new product. Given how fast the technology is progressing, it is becoming more and more difficult to make this choice without any additional help. Some prefer using face-to-face communication with shop assistants, but what if this option is not available or you prefer online shopping? Interactive question answering (IQA) systems can assist customers in this process by providing answers to the questions about products and initiating a dialogue with the customers when their needs are not clearly defined. This is particularly relevant when a product has many features that need to be considered before buying it.

The use of an IQA system should facilitate the process of choosing a product and make it more efficient. However, in order to achieve this goal, the system should know which features are more important for the users and as a result should be given a priority in the decision process. Therefore it is essential to find some ranking methods to assist the navigation of the available features.

This paper proposes a corpus-based method for weighting the importance of product features using a corpus of reviews. Our assumption is that these texts will contain references to product features and, because they are reviews, they will focus on the features that are more likely to determine the purchase of the product. We test our method in the domain of mobile phones, however we believe that it can be employed to other domains describing products.

The remainder of the paper is structured as follows: The next section (Section 2) discusses the related work in the field. Section 3 presents the description of the experiment including its justification and ranking methods developed. The evaluation of the system, including the gold standard, evaluation metrics used and error analysis is presented in Section 4. The paper finishes with conclusions and directions for future work (Section 5).

2 Related Work

This paper addresses the problem of content management for interactive question answering systems which is related to dialogue managers which are part of dialogue systems. However, to the best of our knowledge there is no research similar to the one carried out in this paper. In addition, the novelty of our approach also comes from the fact that it lies at the intersection of several NLP fields such as information extraction, IQA and sentiment analysis. However, work in information extraction, sentiment analysis and interactive question answering can be considered as the most relevant to our research and is briefly presented next.

There are a number of projects focusing on feature extraction for sentiment analysis. The system described in [1] extracts opinion summary about products, but instead of getting the opinion about the product in general, the proposed method tries to produce an opinion summary about separate features. For this purpose, they mine product features discussed by the customers and rate each opinion as positive or negative. Later this information is used to produce feature-based summaries about the products. Authors of [2] aim at solving a similar problem but use multiple specifications of a product for further clustering and extracting of product features. It is also done in order to produce summaries describing the products. Opine [3] is an example of an unsupervised information-extraction system which mines reviews in order to build a model of important product features, their evaluation by reviewers, and their relative quality across products.

Different approaches were developed to address the problem of feature extraction—unsupervised [4] and semi-supervised methods [5], as well as topic modeling [6]. Some researches tried to build specialized domain ontologies manually in order to get better quality resources, but we are aware of only one ontology describing mobile phones [7].

Our research differs from the aforementioned works, because we do not focus on extracting features of the phone from the reviews. Instead, we are more interested in ranking already acquired lists of features using the available customer reviews. In this respect and keeping in mind the goal of our research, it is worth mentioning work previously done in the field of IQA.

Even though we are also not aware of applications in the field of IQA similar to our research, there are several IQA systems that address the problem of effective information management which can be considered relevant to work in question. These systems attempt to help users choose products and rely on constraint-based approaches ([8], [9], [10]).

The system developed by Yan Qu and Nancy Green [8] focuses on providing airline flight information. After the user fills in all the constraints, the system submits a query to the database and if the request was over-constrained, ways to relax the constraints are suggested. In contrast to our method, they do not employ any initial ranking of the constraints in order to set priorities for the use of some constraints.

The approach developed by [9] is used in several systems dealing with restaurant selection, MP3 player operation and navigation tasks. Their aim is to find efficient ways of managing a dialogue and providing a sufficient amount of information to users so that they are neither overwhelmed with too much information, nor left uncertain about some details. Their goal is to choose a single item out of a larger set of items, which is similar to the task we are dealing with. They focus on content optimization where dialogue strategies for dealing with query results are used. Rules governing dialogue steps based on the amount of results are manually constructed and thresholds are predefined. Even though the task is similar to choosing a product (a task we are addressing in this paper), they do not give priority to any of the features and use only quantitative information to make a decision about constraint relaxation and further suggestions. However, in this approach an ontology describing the data and the constraints is used, but constraints do not have any internal ranking.

The research in [10] is similar to [9], however they use predefined rules and simulate interactions with the user in order to further use this information for learning the best policies. They study the domain of in-car and in-home applications and provide examples of dialogues for choosing a song for a playlist. They handle the situations of under-constrained or over-constrained requests and learn ways to deal with them. However, the paper does not mention anything about the usage of an initial ranking of the constraints which we believe can help in the task of IQA.

The systems described above focus on the interaction in terms of the constraint-based systems, however none of them tries to rank the constraints or propose methods to make search for information more optimal in this way. In all the cases, either hand crafted or learnt policies are used to decide which dialogue move to take next. These systems try to act according the number of results they get, and on the basis of this information they attempt to relax the request or ask for additional constraints. We are more interested in suggesting new constraints to the customer and would like to select those that will help the user to choose a product in the quickest time. However, this aspect of the problem is not discussed in these research works.

3 Experiment

Given that our aim is to optimise the process of selecting a product on the basis of its features using an IQA approach, we evaluated several methods for ranking features. These methods are presented in this section and evaluated in Section 4. We start this section by providing a justification for the experiment carried out here.

3.1 *Justification of the experiment*

One way to identify the importance a feature plays in choosing a product is to collect and analyse a large number of interactions between a human and a sales assistant or a computer. This information can be used to learn the appropriate ranking of the features. However, this approach is labour intensive and time consuming, which makes it very expensive especially because it is domain dependent. As a result of the domain dependency, information gathering needs to be repeated every time a system is adapted for a new domain. For this reason, we propose a method which relies on user reviews to determine this ranking.

The underlying assumption of this method is that the most important features will also be mentioned frequently in the user reviews. Therefore, we believe it is possible to propose several weighting schemes which take a corpus of reviews and produce the ranking. Given that these reviews contain a large number of opinionated sentences, NLP techniques are being used to differentiate between positive and negative sentences. This is done in order to identify whether certain types of sentence (e.g. positive) are more likely to contain the necessary information to rank the features correctly.

3.2 *Ranking methods*

We developed several methods for the ranking of product features on the basis of their occurrence in our corpus. In this paper we use features of mobile phones, but the method can be adapted to any other products.

To carry out the experiments presented in this paper, we had to first identify features that can be of interest for users and therefore need to be ranked. Manual construction of such a list did not seem objective enough, and therefore we relied on the infoboxes present in Wikipedia pages describing products of the type of interest (i.e. in the case of this research pages describing mobile phones). The infoboxes contain brief tabular information summarising the content of the page and in the case of products quite often refer to the features of a product. For example, infoboxes in pages about phones may contain the feature “camera” and its corresponding value “5 megapixels”. By collecting these features, we built our list to be ranked. The values corresponding to the features were also collected as a way of identifying indirect references to the features in the text.

WAYS TO MATCH FEATURES: Once we managed to collect the list of features, we could investigate ways to rank them. It was decided to use NLP techniques to find the best ranking algorithm in order to avoid spending a significant amount of time and resources on collecting real customer interactions.

Given the fact that a feature can be expressed in several ways, we used several methods for matching the features extracted from the Wikipedia infoboxes with their occurrences in the texts. For all the ranking methods described in the next subsection, three types of matching methods were used:

- surface-based (also referred to as strict match),
- fuzzy matching (e.g. *battery life* and *lifespan*),
- values for features (e.g. *5 megapixels* and *camera*).

Surface matching implies a strict match between the string denoting a feature from the Wikipedia infoboxes and a string in the corpus. This matching technique does not allow any flexibility on how the feature is expressed in the text. Therefore this type of matching brings some limitations, as language is ambiguous and there are many ways to express the same thing using different surface representations. For this reason, we also implemented a *fuzzy matching* method which takes into consideration not only the surface form, but also considers synonyms extracted from WordNet [11] and manually compiled lists. Several of the problems identified with the first method were solved using fuzzy matching and are discussed in Section 4.6. At the same time, fuzzy matching introduces its own errors which are discussed in the same section.

Another way to improve the matching algorithms is to consider that a feature occurs in a text not only when it is directly mentioned, but also when values corresponding to a feature are used. Despite the appeal of this approach, there are values which are multiword expressions, so a strict matching would give a very low recall. For this reason, we used heuristics which consider a match successful if at least 60% of the text denoting a value was found. This helped us identify more information, but revealed the problem of overlapping features which will be further discussed in Section 4.6.

FREQUENCY-BASED RANKING: The first method explored relies on the frequency of a feature in our corpus of reviews in order to determine its importance. The assumption here is that the more frequently a feature is mentioned, the more important it is for the users. This approach was inspired by automatic summarisation [12]. Therefore, we extract frequency of each particular feature mentioned and use it as its score. For this purpose, all three different types of matching mentioned in the previous subsection were used.

OPINION-BASED RANKING: Given that we are dealing with a corpus of reviews, we thought it could be beneficial to use the polarity of the sentences contained in the reviews in the ranking process. For determining the polarity of a sentence, we use a lexicon-based algorithm based on the SO-CAL algorithm [13]. This method relies on a dictionary which contains words and their semantic orientation scores related to the sentiment expressed. This semantic orientation ranges from -4 to 4, where -4 stands for a totally negative word and 4 for a totally positive word. For our experiments we use the dictionary developed for the original method [14].

In the above mentioned method, the polarity of a sentence is measured as the sum of the semantic orientation present in the words. Those words and their part-of-speech are checked in the dictionary and the semantic orientation computed. Negation markers, modals and intensifiers change the polarity for the next word. The sentence is labeled as positive or negative if the overall semantic orientation is positive or negative. The sentences with score 0 are labeled as being neutral.

We developed two ranking methods based on the identification of the opinion in the text. The first of them takes into account only opinionated sentences and ignores the neutral ones. The assumption here is that the authors of the reviews will express opinions (positive or negative) about features which they find important to them. Frequency-based ranking was applied to the sentences that contain sentiment information. However, we should mention that we did not attach opinions to the particular features and just identified them at the sentence level.

Given that neutral sentences may contain information that could be useful for the ranking, a weighted ranking method which relies on opinion information was implemented. In this method, each occurrence of a feature in a neutral sentence receives a score of only 0.5, whereas an occurrence in an opinionated sentence gets a score of 1. In addition, two more experiments were run which considered only the positive and negative sentences for computing the ranking.

The next section presents the results of the evaluation.

4 Evaluation

4.1 *Corpus description*

For the experiments reported in this paper we compiled a corpus of reviews from the Epinions.com[1] website. Our research focuses on the development of an IQA system for mobile phones. For this reason, we collected reviews from the category *Cellular Phones* on the 21st October 2011. Our corpus contains 3,392 reviews (114,708 sentences) organised into two labels: "yes" and "no". These labels reflect the user's opinion about the product; whether the product is recommended or not. We have 2,437 reviews with the label "yes" and 955 with the label "no", but at this stage we do not use the user's opinion for the ranking. The method used to collect the corpus ensures that the approach can be easily applied to other domains.

4.2 *Gold standard*

For the evaluation purposes we conducted a separate experiment which was aimed at constructing the gold standard. We wanted to use humans' input to rank the features they find most important when choosing a phone. For this purpose, we developed a special drag-and-drop interface which allowed the users to choose the most important features. No special guidelines were given to participants except that they need to pick the 5 most important features for them from a given list. The features were displayed in random order.

In order to prepare the initial list for ranking, we manually checked the features collected from Wikipedia infoboxes and removed those that were difficult to understand without further explanation. We also had to limit the number of options we showed to a user, so that the interface stayed user-friendly and easy to use. For this reason, after discussion between the authors of this paper, it was decided to keep only 47 features we felt to be the most important. We collected a total of 170 answers and used this information to get a weighted ranked list of features by assigning to each feature a score that is equal to the number of times a feature was selected. Table 1 shows the top 20 features together with their frequencies.

[1] http://www.epinions.com/

Table 1. The top 20 features together with their frequencies

90	price	23	memory
81	battery	22	network data connectivity
57	operating system	21	camera
52	phone style	21	talk time
42	manufacturer	20	weight
31	size	19	keyboard
29	standby time	19	main screen
29	GPS	19	touchpad
25	connectivity	18	CPU
24	3g network speed	18	hardware platform

4.3 *Baseline*

In order to evaluate how effective our ranking methods are, we implemented two baselines. The first baseline considers the information from the infoboxes and ranks a feature on the basis of how many Wikipedia articles about mobile phones mention the feature and assigns it a value. The second baseline ranks a feature on the basis of how many times it is mentioned in the Wikipedia articles describing mobile phones. By using these baselines, we can see whether a corpus of reviews is beneficial to us.

4.4 *Evaluation metrics*

Our evaluation was based on comparing several rankings to each other, so we had to consider some formal metrics which will give us an objective number. We decided to choose two metrics that are commonly used to measure the association between two measured quantities.

The first one is the Kendall rank correlation coefficient and is commonly referred to as Kendall's tau coefficient [15]. It depends on the number of inversions of pairs of objects which would be needed to transform one rank order into the other [15]. Equation 1 describes the formula used for calculating Kendall rank correlation coefficient.

$$\tau = \frac{N_c - N_d}{\frac{1}{2} * n * (n-1)} \tag{1}$$

where N_c is the number of concordant pairs, N_d is the number of discordant pairs, whilst n is the total number of pairs. τ takes values between -1 and 1, where -1 means that two rankings are the reverse of each other and 1 shows that rankings are the same.

The second metric we used is Spearman's rank correlation coefficient or Spearman's rho and is a non-parametric measure of statistical dependence between two variables [16]. Spearman's rank takes into account differences between the ranks of each observation on the two variables and Equation 2 shows the way this metric can be calculated.

$$\rho = \frac{\sum_i (x_i - \overline{x}) * (y_i - \overline{y})}{\sqrt{\sum_i (x_i - \overline{x})^2 * (y_i - \overline{y})^2}} \tag{2}$$

Similar to the Kendall's tau, the Spearman's rho values range from -1 to $+1$, and the closer to +1 they are, the more similar the rankings are. The use of these metrics allowed us to output a score after comparing two lists and the results will be provided in the next section.

4.5 *Results*

As described in the previous sections, we have carried out several experiments to produce different rankings of the features. We compared our rankings to the gold standard and the results of this comparison can be found in Table 2. For the evaluation, we used both the full gold standard and only the first 20 items in the gold standard. The justification for the second list is that it is highly unlikely that a customer will be willing to be asked about more than 20 features before they reach a decision.[2]

In Table 2 the rows *Baseline*$_1$ and *Baseline*$_2$ correspond to the two baselines introduced in Section 4.3. As can be seen, the results obtained with the two baselines are among the lowest indicating that using a corpus such as Wikipedia articles is not useful. The three rows with labels starting with *Frequency from reviews* contain the results obtained using just frequency of features in the reviews, but employing different feature matching method. The remainder of the rows contain the results of the methods that use the opinion classifier and different feature matching methods.

[2] In reality, we hope that by using the ranking methods presented in this paper and the interactive question answering system that we are currently developing, the number of questions will be much lower.

Table 2. The evaluation results

Method	**Full list**		**Top 20 items**	
	τ	ρ	τ	ρ
Baseline_1	-0.084	-0.155	0.017	0.019
Baseline_2	0.009	0.010	0.146	0.177
Frequency from reviews, exact match	0.220	0.326	0.187	0.252
Frequency from reviews, fuzzy match	0.116	0.164	0.209	0.292
Frequency from reviews, values match	0.245	0.357	0.470	0.612
Frequency from opinionated sentences, exact match	0.218	0.326	0.241	0.357
Frequency from opinionated sentences, fuzzy match	0.165	0.245	0.209	0.317
Frequency from opinionated sentences, values match	0.241	0.346	0.513	0.647
Weighted frequency, exact match	0.159	0.235	0.166	0.211
Weighted frequency, fuzzy match	0.207	0.298	0.230	0.332
Weighted frequency, values match	0.051	0.083	0.123	0.160
Frequency from negative sentences, exact match	-0.051	-0.078	0.016	0.008
Frequency from negative sentences, fuzzy match	-0.115	-0.172	-0.026	-0.056
Frequency from negative sentences, values match	-0.024	-0.045	-0.053	0.134
Frequency from positive sentences, exact match	0.011	0.018	0.123	0.130
Frequency from positive sentences, fuzzy match	0.175	0.253	0.155	0.222
Frequency from positive sentences, value match	0.125	0.197	0.021	0.000

4.6 *Error analysis*

Analysis of the results reveals that the best performing methods are the ones using either the frequency of the features in the full corpus of reviews or the frequency of features only in the opinionated sentences. In both cases, the values of the features are used for matching. These results hold both for the full gold standard and when only the top 20 items are considered. The rest of the results are considerably lower, especially when only the top 20 items are considered. Strangely enough, the method which gives a weight of 0.5 to features that appear in sentences that do not have a polarity, and which is somehow between the two best performing methods in terms of how features are scored, performs rather poorly regardless of the matching method used. The same happens if we use only the positive or negative sentences.

Our experiments revealed several problems to be addressed in order to get better results. One of the first issues we had to address when implementing the matching algorithm was the possibility to refer to the same feature in several different ways. For example the feature *operating system* can be referred to using “Operating System”, “operatingsystem” or “os”. Even though we used WordNet and manually compiled lists, it is unlikely that we managed to cover all the possible ways people use to refer to a feature. For this reason, the fuzzy matching method is not always very precise. Related to this problem is the fact that the list of values of a feature is likely to grow over time. Unless these values are listed in Wikipedia and our matching algorithm gets updated there is no way to capture the mention of a corresponding feature in a review.

Another problem with our approach is related to the ambiguity of the features. E.g., the features “standby time” and “usage time” have very similar meaning. This situation becomes even more problematic when the features are considered out of the context, as in the case of the experiment carried out to produce the gold standard. In light of this, word sense disambiguation-like methods could be considered to find out whether two similar expressions refer to the same feature on the basis of their context.

Another problem related to matching of features is with pairs such as “camera” and “video camera”. When using only strict matching, it is difficult to decide whether the users just described a photo camera or whether they are referring to a photo-video camera. This problem becomes more serious when both forms are used in the text and “camera” is coreferential with “video camera”. The only way to address this problem is to employ a coreference resolver.

The use of WordNet to obtain synonyms introduced a fair amount of errors as well. For example, for the feature “carrier” some of the synonyms are “postman”, “carrier wave”, “mailman” and “attack aircraft carrier” which are completely unrelated to the features of mobile phones. This is due to the fact that the word used to refer to this feature is far too general and therefore ambiguous. At the other extreme are the features such as “hardware platform” which are too specific and do not appear in WordNet. For this reason, it will be necessary to produce a better list of synonyms for the features.

5 Conclusions and Future Work

This paper addressed the problem of feature ranking for interactive question answering systems which help customers to choose the right product for them. Two baselines and several ranking methods were evaluated against a gold standard collected from users. The Kendall rank correlation and the Spearman’s rank correlation coefficients were applied in order to provide an objective evaluation of the ranking methods applied. An experiment showed that two of the ranking methods proposed perform far better than any other methods. The evaluation also confirmed the fact that using a corpus of reviews is beneficial for feature ranking. The results were further improved by using only the opinionated sentences for scoring features.

Error analysis revealed that a large number of the problems we experience are due to the fact that features can be expressed in text using different expressions. For this reasons more refined methods for identifying occurrences of the features in a text should be explored, including the use of coreference resolution. The weighting method currently used relies on frequency, however other methods for counting features should be investigated as well.

Finally, our motivation is to optimise the dialogue between a user and an IQA system for selecting mobile phones. In light of this, the best way to prove the usefulness of the ranking methods is to carry out an extrinsic evaluation. This type of evaluation will be considered in the future.

ACKNOWLEDGEMENTS The research was partially supported by Unilever. We thank all those who contributed to building the gold standard.

REFERENCES

1. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proc. of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. KDD'04, New York, NY, USA, ACM (2004) 168–177
2. Meng, X., Wang, H.: Mining user reviews: from specification to summarization. In: Proc. of the ACL-IJCNLP 2009 Conference Short Papers. ACLShort'09, Stroudsburg, PA, USA, Association for Computational Linguistics (2009) 177–180
3. Popescu, A.M., Etzioni, O.: Extracting product features and opinions from reviews. In: Proc. of the conference on Human Language Technology and Empirical Methods in Natural Language Processing. HLT'05, Stroudsburg, PA, USA, Association for Computational Linguistics (2005) 339–346
4. Raju, S., Pingali, P., Varma, V.: An unsupervised approach to product attribute extraction. In: Proc. of the 31th European Conference on IR Research on Advances in Information Retrieval. ECIR'09, Berlin, Heidelberg, Springer-Verlag (2009) 796–800
5. Zhai, Z., Liu, B., Xu, H., Jia, P.: Clustering product features for opinion mining. In: Proc. of the fourth ACM international conference on Web search and data mining. WSDM'11, New York, NY, USA, ACM (2011) 347–354
6. Zhai, Z., Liu, B., Xu, H., Jia, P.: Constrained lda for grouping product features in opinion mining. In: Proc. of the 15th Pacific-Asia conference on Advances in knowledge discovery and data mining – Part I. PAKDD'11, Berlin, Heidelberg, Springer-Verlag (2011) 448–459
7. Junwu, Z., Bin, L., Fei, W., Sicheng, W.: Mobile ontology. JDCTA: International Journal of Digital Content Technology and its Applications **4**(5) (2010) 46–54

8. Qu, Y., Green, N.: A constraint-based approach for cooperative information-seeking dialogue. In: Proc. of the Second International Natural Language Generation Conference. (2002)
9. Varges, S., Weng, F., Pon-Barry, H.: Interactive question answering and constraint relaxation in spoken dialogue systems. Natural Language Engineering **15**(1) (2007) 9–30
10. Rieser, V., Lemon, O.: Does this list contain what you were searching for? Learning adaptive dialogue strategies for interactive question answering. Natural Language Engineering **15**(1) (January 2009) 55–72
11. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database. The MIT Press (1998)
12. Luhn, H.P.: The automatic creation of literature abstracts. IBM Journal of research and development **2**(2) (1958) 159 – 165
13. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. Computational Linguistics (2011) 1–41
14. Taboada, M., Anthony, C., Voll, K.: Methods for creating semantic orientation dictionaries. In: Proceedings of 5th International Conference on Language Resources and Evaluation (LREC), Genoa, Italy (2006) 427–432
15. Abdi, H.: Kendall rank correlation. In Salkind, N., ed.: Encyclopedia of Measurement and Statistics. Thousand Oaks (CA): Sage (2007) 508–510
16. Maritz, J.: Distribution-free statistical methods. Science Paperbacks. Chapman and Hall (1984)

NATALIA KONSTANTINOVA
RESEARCH GROUP IN COMPUTATIONAL LINGUISTICS,
UNIVERSITY OF WOLVERHAMPTON,
WOLVERHAMPTON, UK
E-MAIL: <N.KONSTANTINOVA@WLV.AC.UK>

CONSTANTIN ORĂSAN
RESEARCH GROUP IN COMPUTATIONAL LINGUISTICS,
UNIVERSITY OF WOLVERHAMPTON,
WOLVERHAMPTON, UK
E-MAIL: <C.ORASAN@WLV.AC.UK>

PEDRO PAULO BALAGE
RESEARCH GROUP IN COMPUTATIONAL LINGUISTICS,
UNIVERSITY OF WOLVERHAMPTON,
WOLVERHAMPTON, UK
E-MAIL: <PEDROBALAGE@GMAIL.COM>

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 71–88
RECEIVED 30/10/11 ACCEPTED 09/12/11 FINAL 13/06/12

Detecting Irregularities in Blog Comment Language Affecting POS Tagging Accuracy

MELANIE NEUNERDT, BIANKA TREVISAN,
RUDOLF MATHAR, AND EVA-MARIA JAKOBS

RWTH Aachen University, Germany

ABSTRACT

Studying technology acceptance requires the survey and analysis of user opinions to identify acceptance-relevant factors. In addition to surveys, Web 2.0 poses a huge collection of user comments regarding different technologies. Extracting acceptance-relevant factors and user opinions from such comments requires the application of Natural Language Processing (NLP) methods, particularly Part-of-Speech (POS) tagging. Applied to typical blog language POS tagging often suffers from high error rates. In this paper, we present multiple user-specific studies of blog comments to analyze the relation between blog language and performance of NLP methods. We propose an approach, which leads to enhancement of POS tagging and lemmatizing quality. Furthermore, we present an ontology-based corpus generation tool to improve the identification of topic- and user-specific blog comments. Utilizing the generation tool, a corpus dealing with mobile communication systems (MCS) is exemplarily created. Furthermore, we analyze and transform the identified comments into structured datasets.

KEYWORDS: *Natural Language Processing, Part-of-Speech Tagging, weblog, user writing style, ontology search, corpus generation.*

1 INTRODUCTION

Typical instruments used in acceptance research are questionnaires, interviews, or focus groups. The according outcomes reveal user opinions about a particular topic such as MCS. Nevertheless, traditional methods in acceptance research have the major shortcoming of being subject to numerous methodological effects. In surveys, respondents tend to answer dishonestly (*Social-Desirability-Response-Set*). As a result, *artificial* or *falsified data* is collected. Moreover, the utilization of traditional surveys calls for high user willingness, hence collecting a sufficient amount of data is a very arduous and time-consuming task. However, this type of data provides the advantage of being structured in a predefined manner, which ensures the availability of information as well as efficient data access. As a complement to traditional methods in acceptance research, we propose an innovative approach in which natural language discourses from web data, such as blog comments, are analyzed with the aim of identifying frequently discussed topics in a particular field and current user evaluations on this topic. Particularly, this approach benefits from the fact that the data is *natural* or *authentic*, that it is accessible, and that it might be downloaded quite efficiently.

However, the analysis of natural language discourses is problematic since Internet users tend to a more colloquial formulation or expression style. More precisely, the language used in blogs suffers from numerous lexical, syntactic, semantic, stylistic, and typographical phenomena, e.g., unconventional use of punctuation marks such as *?!?!*, novel typographical means of evaluation such as :-), or frequent use of interjections such as *haha*. Common NLP methods such as POS tagging cannot process these text type-specific phenomena correctly and, in consequence, high error rates appear. As a result, no exact statements can be made. Usually, POS tagging is the first step of text processing for further text analysis. The output can be used for further NLP processing steps, e.g., opinion detection. Therefore, a high POS tagging accuracy is very important for further investigations. Modern taggers achieve a per-word-accuracy of 97.53% when tagging German newspaper corpora [1]. Unfortunately, the accuracy drops significantly when applying taggers to web corpora [2]. First, low tagging accuracy is caused by the use of topic-specific terms and abbreviations, e.g., *Datenbandbreite (data bandwidth), iPhone, UMTS*. Second, blog comments are non-standardized texts, and characterized by different users' writing styles. Therefore, POS tagging has to be adapted to blog-specific linguistic phenomena. Our work focuses on the identification of causes for POS tagging decision errors in blog comments. For this purpose, an ontology-based corpus generation tool is developed which is used to create a topic-specific corpus. The corpus is analysed for blog-specific linguistic phenomena.

The paper is structured as follows. In Section 2, the ontology-based corpus generation tool is introduced. Afterwards, the created corpus is presented in Section 3. In Section 4, empirical results of the corpus analysis on blog-specific linguistic phenomena are shown. In section 5, we present a first sketch of a POS tagger adapted to the language used in blog comments. Section 6 presents an example for blog comment transformation into suitable data representation. Finally, we present our conclusion.

2 Ontology-Based Corpus Generation

The increasing number of user-generated web content provides a large amount of opinionated data. However, people express their opinion differently and use different terminology in written web discussions. Hence, it becomes hard to access and extract topic-specific user opinions by simple keyword search. Particularly, usage of ontology-based search that considers keyword relations, such as synonymy, leads to an increase of quantity and quality of retrieved search results. Our tool named CROW can be used to selectively search for blog comments. It offers the ability to create a corpus according to a predefined ontology. The resulting corpus serves for further linguistic analysis and mathematical calculations.

2.1 *Ontology principles*

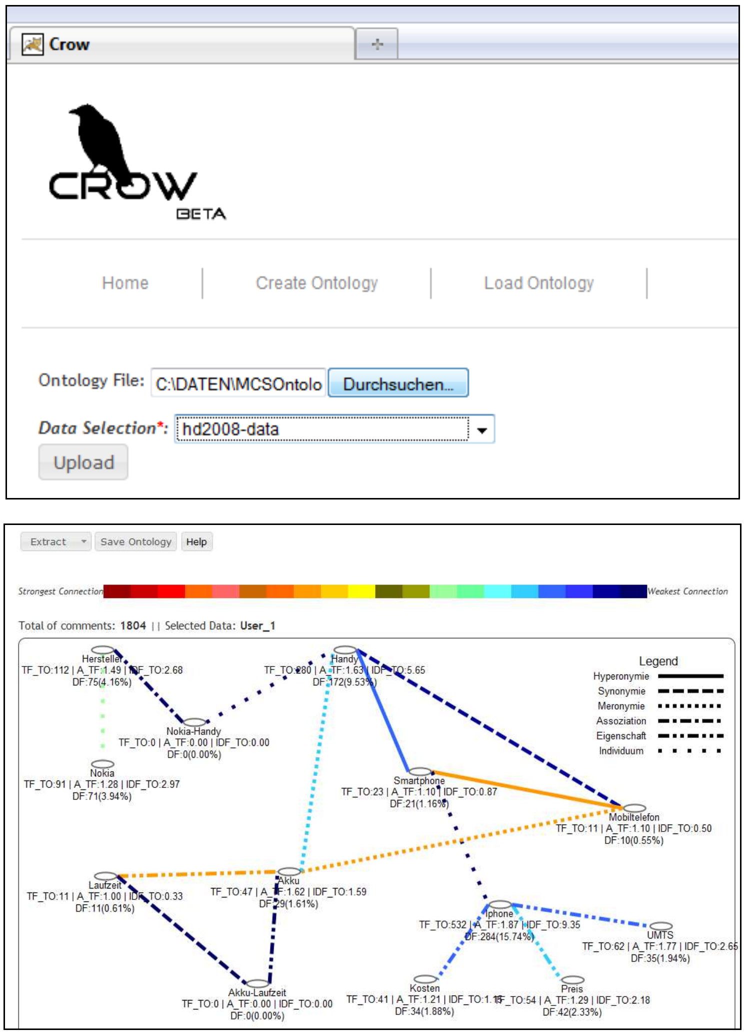
Ontologies play an important role in the field of knowledge engineering and semantic web research. Typical applications are ontology learning [3] and ontology-based focused crawling [4,5]. Ontologies contain a collection of concepts represented by terms that exist in a certain domain. The relations are determined according to linguistic usage or to human semantic association, respectively. Thus, ontologies can be described by a directed graph where the nodes represent concepts and the edges represent semantic relations. In this work, the edges describe the dependency between MCS components, properties, and instances. The ontology consists of four relation types: (1) Hierarchical and (2) non-hierarchical relations as described in the literature [6,7], (3) attributive relations, and (4) type-token relations.

(1) Hierarchical relations are hyperonymy and meronymy. With these relations, over- and subordinated concepts (e.g. phone as a hyperonym of mobile phone) as well as part-whole relations between concepts (e.g. display as a part of mobile phone) of MCS can be expressed [6].
(2) A synonymy describes a non-hierarchical relation. The relation links concepts (terms) that have an identical or similar meaning, e.g., mobile phone and cell phone. In the ontology, synonyms are summarized into a term set (synset) [7].
(3) Attributive relations indicate which properties or utilization-types are ascribed to an object. The relation property indicates object properties, e.g., robustness and longevity. With the relation association concepts are connected to each other that have no lexical-semantic connection. Rather, they are related to each other on the basis of user experiences or usage types.
(4) The type-token relation instance assigns real world examples or class representatives to classes, e.g., *iPhone* a respresentative for *cell phones*.

2.2 *Tool functionalities*

CROW is a web application providing a graphical user interface. The application enables the user to specify an ontology manually or to reload it from the storage; see Figure 1. Furthermore, a comment corpus can be selected or filtered by user name or time stamp.

Different edge types illustrate different relation types. Various statistic scores for the underlying comments and the concept related terms are calculated and plotted. In addition to the overall number of comments, e.g., the *document frequency* (*DF*) including single concepts and combination of related concepts is computed. It shows the co-occurrence of concepts in user discussions. For all related concepts, the correlation coefficient is calculated between the *term frequencies* (*TF*). The correlation strength is indicated by the color of the relation line according to a given color table; see Figure 1. These values support the user evaluating relevant blog comments and give an impression of the used terminology regarding MCS. Furthermore, users can use a particular ontology path for comment extraction according to the



Ontology File: C:\DATEN\MCSOntolo

*Data Selection**: hd2008-data

Total of comments: **1804** || Selected Data: **User_1**

Fig. 1. User interface and ontology visualization (CROW).

statistical results and their interpretations. For instance, the correlation coefficient allows identification of how often different users use a synonym. Hence, the topic-selection can be refined and a blog comment corpus with high relevance to MCS terminology can be generated.

Two types of comment extraction options are provided: First, the intersection set of comments covers all comments including all concepts of the selected ontology path (sub ontology). Second, the union set of comments contains at least the occurrence of one concept in the selected sub ontology.

3 Blog Comment Corpus

For blog comment analysis, a set of approximately 166 thousand German blog comments posted from January 2008 to December 2009 is considered. In a number of preprocessing steps, posted comments are bowdlerized from enclosing webpage elements, html-tags, and corresponding meta information, e.g., user name is extracted and added as metadata to the comment. Table 1 illustrates some statistics about the corpus collection, particularly, in terms of covered users and their blogging frequency. Comparing the statistical values for both years shows that the data corpus for 2008 and 2009 follows a very similar distribution.

Table 1. Comment corpus statistics.

	Data 2008	Data 2009
Articles	1,252	1,289
Comments	84,203	81,831
Users	10,474	9,509
#Comments per article	67	63
#Comments per user	8	9

For further analysis, we use the ontology-based corpus generation tool described in Section 2. Therefore, a specific topic of MCS is focused. As an example, we select the topic *Handy (cell phone)*; the sub ontology is created manually and is part of the whole MCS ontology. This ontology is used to extract the MCS corpus.

Based on the generated MCS corpus, all users are ranked in decreasing order according to their posting frequency. We assume that users who post regularly tend to develop their own writing style. Among these users non-standardized expressions, colloquial expressions, and emoticons are used more frequently. Moreover, grammar rules are violated more often than in comments posted by users that post only once or twice. Considering the posting frequency distribution, the 12 most active users are selected for further statistical and linguistic analysis. The 3,249 comments of the 12 users form the analysis corpus with a total of 675,762 tokens. The sample corpus is used to identify and evaluate acceptance-relevant factors or user opinions considering the design of user devices, e.g., cell phones.

4 Analysis of Writing Style Irregularities

Blog-specific writing styles lead to processing errors in NLP methods, e.g., POS tagging. Analysing and evaluating users' writing styles aimes at enhancing NLP methods with respect to blog comment processing and, in particular, adapting existing approaches to the language in blog comments for automatic opinion extraction. To detect user opinions, all text characters, e.g., emoticons, are important. Hence, bowdlerizing comments is not a sufficient solution for our task. The goal is to investigate causes for decision errors in NLP methods and handle those text irregularities.

Opinion detection in the field of text retrieval is still a challenging task [8,9]. Two different approaches are used for opinion detection and classification: First, machine learning-based approaches based on training data. And second, lexicon-based approaches, which use a lexicon of sentimental words, e.g., list of positive and negative words, provided by linguistic quantization and weightings [10,11].

4.1. *User-specific statistical analysis*

In this section, some statistical analysis is performed to identify non-standardized text-patterns in blog comments. We address the task of detecting different writing styles, which need to be considered for enhancing the accuracy of NLP methods. Blog comment users evaluate objects by using different non-standardized evaluative expressions. Therefore, we choose features according to three different types of writing style: 1) emoticon usage, 2) usage of colloquialisms, and 3) punctuation marks usage. Representatives or examples of the different expression types are counted by frequency, which is the basic and most popular feature set used in text classification and clustering tasks [11,12]. Consequently, we count the frequencies of all features. To make the results comparable, we normalize each feature value by the feature-specific maximum, determined over all users.

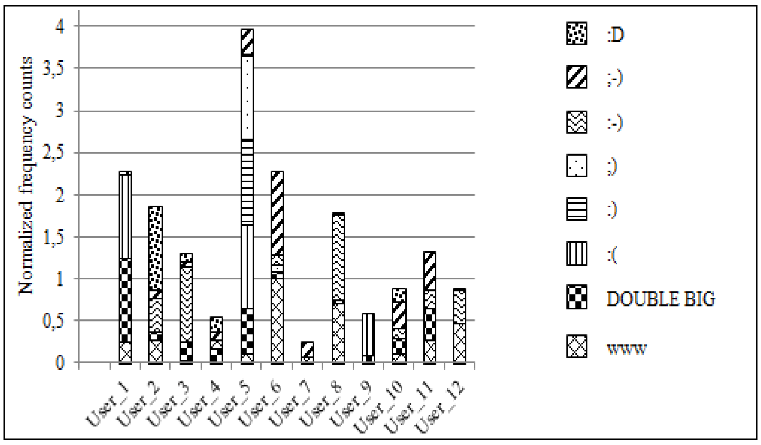


Fig. 2. User-specific emoticon usage.

Emoticon usage. To measure the occurrence of emoticons, we count the six most common emoticons composed of two and three characters. Considering the goal of opinion extraction, three emoticons with positive expression (*:D, :), :-)*) two ironic emoticons (*;), ;-)*), and one emoticon with negative expression (*:(*) are considered. Empirical results show that only 5 out of 70 (< 10%) users do not use any emoticons to express their opinion. Figure 2 shows the distribution of used emoticons for the first 12 users based on the comment corpus described in Section 3.

Colloquialisms. A large amount of comment data shows that German grammar is generally not maintained. Users are very modest in using capitalization and produce numerous orthographical errors and letter transpositions. Furthermore, they tend to shorten words, e.g., *ne, draus (hence), drum (therefore),* and introduce new terms to express their opinions. Therefore, we suggest measuring the level of colloquialism and introduce a number of different features.

Firstly, we use count of words features to create manually a small list of terms typically used in comments, e.g., *lol, haha, hehe, nö, ne, naja*. Secondly, we count the number of complete capitalized words and words where the first two letters are capitalized, e.g., *TElefon.* The second type of words has a high occurrence in user comments due to fast writing style. Since there are numerous capitalized acronyms where at least the first two letters are capitalized, e.g., *WLAN, UMTS, GBit,*

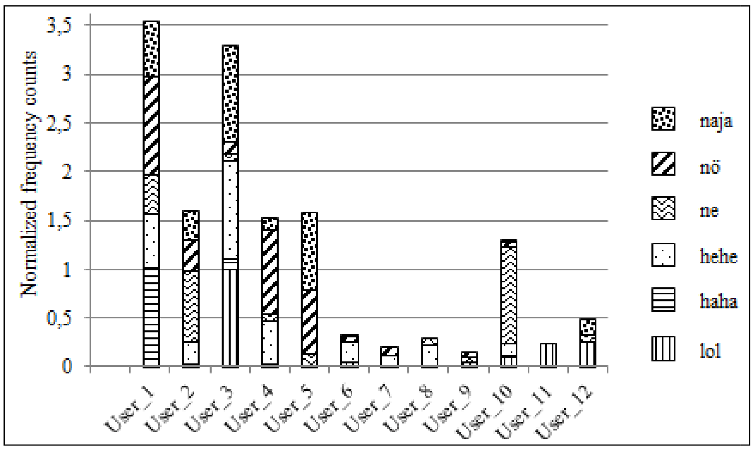


Fig. 3. User-specific colloquial expressions.

this has to be considered in counting. Therefore, we construct a list with acronyms that are related to the context of MCS. Our data corpus, including articles and comments, serves as basic component to determine context-relevant acronyms. The list is generated automatically using some regular expressions. Finally, we count forgotten space characters after dots and commas, due to users' carelessness. In order to avoid counting of digits, including commas or punctuations. Figure 3 shows the results.

Punctuation usage. To describe the punctuation usage, we use frequency counts of various punctuation marks. Users disregard punctuations on one hand and introduce new ways of punctuation to express the intensity of their opinion on the other hand. Therefore, we distinguish between two types of punctuations: single punctuations used in the conventional manner according to the German punctuation rules and multiple punctuations which indicate evaluative utterances, e.g., *???, !!!, *, and ‘.* Multiple punctuations are measured by counting bigrams and trigrams in the comment. The results are depicted in Figure 4.

To measure the degree of irregularity, we consider all feature types together. Therefore, we sum up the feature values for each user. The result is depicted in Figure 5.

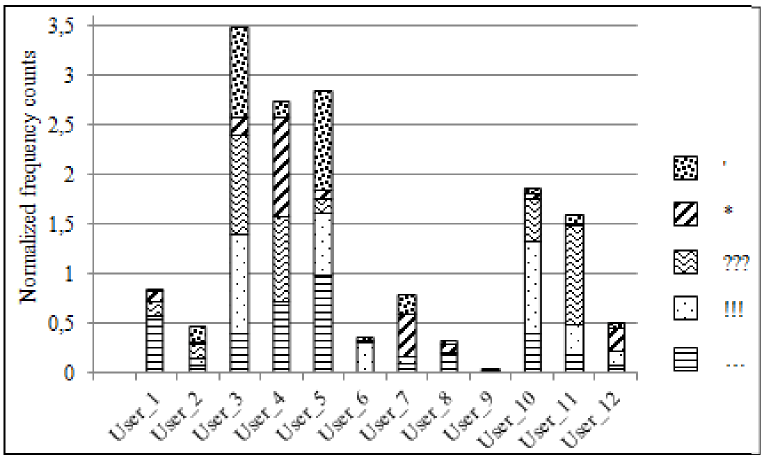


Fig. 4. User-specific punctuation usage.

4.2. *User-specific evaluation style analysis*

Written user comments in weblogs are characterized by text type-specific expressions and formulation styles, e.g., colloquial or ironic expressions. In Section 4.1 statistical analysis concerning the usage of specific non-standardized tokens/expressions is performed, whereas in this section evaluative expressions containing more than one word are analysed. The focus is to detect sequences of evaluative expressions in blog comments, which need to be considered for the extraction of user opinions. Some evaluative expressions are collected systematically and defined for the purpose of automatic processing.

The blog comments are analysed manually using the content analysis software MaxQDA. The manual analysis is a necessary preliminary data investigation; the results form the linguistic knowledge basis for the subsequent automatical extraction of user evaluations. Each user comment of the corpus is sifted in order to identify evaluative expressions. If an evaluative expression is discovered, all relevant parts of the expression are marked and categorized according to the literature [13,14,15]. The text segments are assigned to the following categories: Noun phrase, I-sentence, dialection, weighting, irony, negation, onomatopoeia, idiom, relationalization, rhetorical question, and comparison.

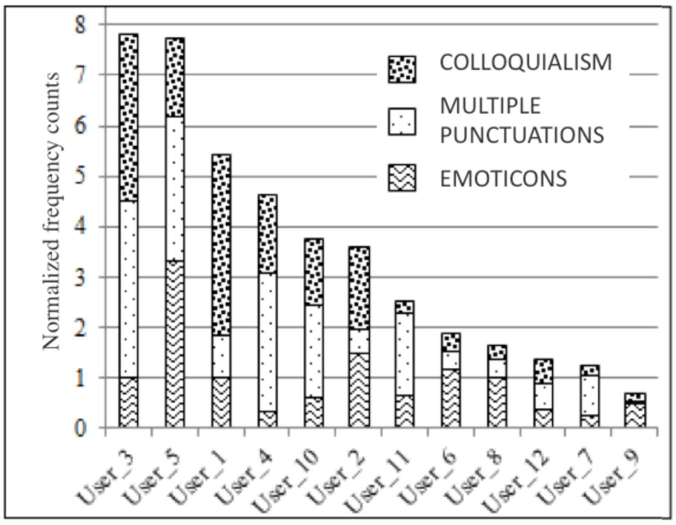


Fig. 5. User ranking: grammatical accuracy combining all feature types.

(1) Noun phrase: Expressions that consist of a valuing adjective and a noun;
(2) I-sentence: Expressions that express the stance and attitude of the user;
(3) Dialection: Expressions that are weakened or reinforced by question-answer phrases;
(4) Weighting: Expressions, that indicate the weighting of the evaluation aspect or topic;
(5) Irony: Expressions in which the writer says the opposite of what he means. Statements contrary to regular, shared knowledge and opinions of society;
(6) Negation: Expressions that get a negative rating by using a negation particle;
(7) Onomatopoeia: Expressions in the form of a loud imitation of a natural or other non-linguistic acoustic phenomenon;
(8) Idiom: Evaluative expressions that consist of a language typical phrase;
(9) Relationalization: Expressions that weaken an opinion or statement or put it into perspective;
(10) Rhetorical question: Expression, in which a question is used to intensify an opinion;

(11) Comparison: Expressions, in which two or more objects or evaluation aspects are compared with each other.

To ensure the reliability of the results, two independent coders analyze the comments. Lastly, the coders check their categorization results in order to reach coding agreement (*intercoder reliability*). The evaluation of the categorization results is performed numerically: the coding frequency for each type of evaluative expression and user are counted. The results of the analysis are summarized in Table 2.

Table 2. Users with the highest number of evaluative expression and corresponding categories.

	User 7	User 12	User 1	User 5	User 2
#blog comments	370	39	475	316	418
#tokens	12,3213	9,632	10,5764	53,364	53,148
Noun phrase	15	2	2	14	6
I-sentence	35	61	21	13	22
Dialection	7	0	0	0	0
Weighting	20	3	10	12	10
Irony	15	37	8	2	5
Negation	5	0	3	2	3
Onomatopoeia	0	1	7	1	1
Idiom	6	5	2	7	3
Relationalization	4	0	1	2	7
Rhetorical question	6	7	11	1	4
Comparison	18	3	16	20	10
#Total codings	131	119	81	74	71

Regarding Table 2, it is evident that users evaluate differently. Furthermore, the frequencies of evaluative expressions differ a lot. For instance, 39 comments of User 12 build only a small fraction of the corpus. Nevertheless, 119 evaluative expressions are identified in the user's posted comments. Compared to the other users, User 12 evaluates on MCS more frequently. The ratio between the amount of evaluative expressions and the number of analysed comments is in balance for the other users. Furthermore, we observe that there is a kind of user preference for the expression of certain evaluative expression types. While User 7, 12, and 5 most often use noun phrases, I-sentences, weightings, and irony, users 1 and 2 use more frequently evaluative types like rhetorical questions. Examples 1 to 3 show a selection of user utterances from the analyzed corpus.

Ein Fass ohne Boden. / A bottomless pit. (1)
(Idiom, User 12)

Wo gibt es perfekte Geräte, die für alle genau richtig sind? / Where there are perfect devices that are just right for all? (2)
(Rhetorical question, User 1)

App Store liefert dann noch diverse "nice to have" Erweiterungen. / App Store still provides several "nice to have" extensions. (3)
(Noun phrase, User 7)

Evaluative expressions like these cause processing errors. For instance, the idiom in (1) would not be recognized as a phrase. Instead, each word would be extracted and processed separately. In the same way, analysis and evaluation errors would occur for (2). Furthermore, the use of anglicism is problematic (3), since POS tagging tools do not reliably recognize foreign words.

5 Enhancement of POS Tagging Results

A number of approaches aim at the enhancement of NLP methods by means of preprocessing with the goal to bowdlerize comments. Since we want to detect user opinions, all text characters, e.g., emoticons and multiple punctuations, are important for interpretation. Therefore, our approach does not remove characters but further enables NLP methods to handle such irregularities [16]. To analyze the accuracy of NLP methods applied to blog comments, we choose a POS tagger as an example. A POS tagger annotates every word with a POS tag and a lemma. A tool for automatic German text corpora annotation is the TreeTagger, developed at the University of Stuttgart [17]. The TreeTagger adds a POS tag according to the Stuttgart-Tübingen Tagset (STTS) and a lemma according to a special lexicon to each word [17].

In our approach, the TreeTagger is improved by using a blog-specific lexicon (BS lexicon) as well as a topic-specific lexicon (MCS lexicon). The BS lexicon contains blog-specific expressions and is created according to the results of sections 4.1 and 4.2. Analysing the POS tagging results without the BS lexicon shows that comments indeed suffer from a high number of incorrect POS tags and unknown lemmas, but the results for neighboring words in the same sentence are not negatively affected. Hence, integrating blog-specific expressions by means of a lexicon enhances POS tagging results. Table 3 shows a tagged sentence part starting with a word not contained in the lexicon.

Table 3. Extract of the tagged corpus.
Incorrect POS tags are marked with a frame.

	Without auxiliary lexicon		**With auxiliary lexicon**	
Token	**POS tag**	**Lemma**	**POS tag**	**Lemma**
Hmm	NN	<unknown>	ITJ	Hm
,	$,	,	$,	,
komisch	ADJD	komisch	ADJD	komisch
dass	KOUS	dass	KOUS	dass
ich	PPER	ich	PPER	ich
[...]				
HSDPA	NN	<unknown>	NE	HSDPA
und	KON	und	KON	und
EDGE	NN	<unknown>	NE	EDGE
Sender	NN	Sender	NN	Sender

The example shows that the colloquial expression *Hmm* does not affect the POS tagging results in the remaining sentence. Thus, it is sufficient to define the term in the BS lexicon. Tagging results considering the MCS and BS lexicon are shown in the right part of the table. The MCS corpus is used to create the auxiliary lexica, whereas the comments posted by the selected 12 users are not considered. Moreover, the selected comments of the 12 users serve as a test sample for lexicon evaluation. Finally, the lexicon contains about 2,000 topic-specific terms related to MCS [18]. The MCS lexicon works as an auxiliary lexicon and complements the embedded standard lexicon of the TreeTagger.

The evaluation part (a test sample of about 3,500 comments and 390,000 token) is tagged with (a) and without (b) using the MCS lexicon and BS lexicon. As a result, the mean number of tagging errors dropped from (a) 9.44% to (b) 7.58% (mean value). Table 4 illustrates the improvements that are achieved.

In the first column of Table 4, the error rates applying the standard TreeTagger are shown. The ranking according to these error rates is strongly related to the ranking shown in Figure 4, which confirms our assumption that the POS tagging accuracy is strongly related to the degree of irregularity in users' writing style. Furthermore, in the rightmost column the improvements for different users are depicted.

Table 4. User-specific POS tagging results (in %).

	Tagging Error Rates		
	without auxiliary lexica	with auxiliary lexica	Improvement
User 1	7.64	5.95	1.69
User 2	9.69	6.91	2.78
User 3	12.15	9.09	3.06
User 4	9.32	7.87	1.45
User 5	12.20	10.16	2.04
User 6	8.74	6.68	2.06
User 7	7.91	5.91	2.00
User 8	8.59	7.23	1.34
User 9	10.64	8.71	1.93
User 10	11.22	8.52	2.70
User 11	10.36	8.35	2.01
User 12	7.98	6.90	1.08
Average	9.7	7.69	1.86

For some users, a very high improvement in the tagged data is achieved when using the auxiliary lexica (e.g. User 3, User 10); in contrast, the tagged results of other users change little (e.g. User 9, User 12). In total, the tagging accuracy for each user is improved.

6 Dataset Generation

The overall goal of this work is to extract user opinions from comments and to generate suitable datasets for the integration into an acceptance model. By means of the ontology-based corpus generation tool, some user comments dealing with user devices (cell phones) as a sub-topic of MCS are extracted. With respect to the results in Sections 4.1 and 4.2, the extracted user comments are evaluated with the aim of creating user-specific datasets. Table 5 depicts dataset examples for two users evaluating five components of particular user devices. For representation of acceptance strength, we choose a scale from 1 (low acceptance) to 6 (high acceptance). The mapping is performed according to used features described in Section 4.1, e.g., emoticons or multiple exclamation points, and evaluative expressions listed in Section 4.2. Value -99 denotes that no information on the stated issue is available. Furthermore, interpreting user expressions allows for the extraction of some demographic information; see Table 5, column 2.

Table 5. Example datasets constructed based on user comments.

User	Occupation	#Comments	Object	Display	Material	Usability	Camera	Battery	Reference
1	Emp-loyed	1,804	iPhone	-99	2	3	1	4	Nokia (E 51), iPod
			Nokia E51	-99	2	1	1	-99	
5	Emp-loyed	796	iPhone	6	6	5	4	-99	Siemens S55
			Siemens S55	3	-99	5	-99	-99	

7 Conclusion

In this work, we presented a study of text type-specific writing styles in blog comments to identify causes for NLP decision errors. By means of an ontology tool, a topic-specific blog comment corpus is created. Based on this corpus, we perform an analysis of text type-specific linguistic phenomena such as punctuation marks, emoticon usage, and colloquial expressions. Results show that the degree of grammatical or stylistic irregularity in blog comments differs significantly for different users. As an example for NLP methods, TreeTagger results based on comments for 12 different users are presented and discussed. Combination of the TreeTagger with topic- and blog-specific lexica enhances the tagging results for blog comments. The results show that the error rates as well as improvements are strongly related to the degree of irregularity in the users' writing style.

Nevertheless, further improvement, particulary with respect to opinion detection in blog comments, is desirable. A still unsolved task is the correct tagging of emoticons and multiple punctuation marks, which is crucial for opinion detection. Therefore, the adaption of the tokenizer is necessary and new training of the TreeTagger with annotated blog comments is required. Future work will further enhance POS tagging results for blog comments. Moreover, a gold standard for blog comments will be created.

Acknowledgments. This work was partially supported by the Project House HumTec at RWTH Aachen University, Germany.

REFERENCES

1. Schmid, H.: Improvements in Part-of-Speech Tagging with an Application to German. In: ACL SIGDAT-Workshop, pp. 47–50 (1995)
2. Giesbrecht, E., Evert, S.: Is Part-of-Speech Tagging a Solved Task? An Evaluation of POS Taggers for the German Web as Corpus. In: 5th Web as Corpus Workshop (2009)
3. Aleksovski, Z., Klein, M., ten Kate, W., van Harmelen, F.: Matching unstructured vocabularies using a background ontology. In: EKAW. Springer (2006)
4. Ehrig M., Maedche, A.: Ontology-focused crawling of web documents. In: Symposium on Applied Computing, pp. 1174–1178. ACM, New York (2009)
5. Zheng, H.-T., Kang, B.-Y., Kim, H.-G.: An ontology-based approach to learnable focused crawling. Information Sciences 178, 4512–4522 (2008)
6. Roman, S.: Eine Ontologie für die Grammatik. Modellierung und Einsatzgebiete domänenspezifischer Wissensstrukturen. In: KONVENS, pp. 125–129 (2006)
7. Fellbaum, C.: Wordnet. In: Poli, R., Healy, M., Karneas, A. (eds), pp. 231–243 (2010)
8. Missen, M., Boughanem, M., Cabanac. G.: Challenges for Sentence Level Opinion Detection in Blogs. In: 8th ICIS, pp. 347–351. IEEE Press (2009)
9. Missen, M.M., Boughanem, M.: Using WordNet's Semantic Relations for Opinion Detection in Blogs. In: ECIR, pp. 729–733. Springer-Verlag, Berlin, Heidelberg (2009)
10. Taboada, M., Brooke, J., Toloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. Computational Linguistics 37, pp. 267–307 (2011)
11. Sebastian, F.: Machine learning in automated text categorization. ACM Computing Surveys 34, 1–47 (2002)
12. Mishne, G., Glance, N.: Leave a Reply: An Analysis of Weblog Comments. In: 3rd Annual Workshop on the Weblogging Ecosystem at WWW (2006)
13. Bednarek, M.: Language patterns and ATTITUDE, Functions of Language 16, 165–192 (2009)
14. Bednarek, M.: Evaluation in Media Discourse: Analysis of a Newspaper Corpus. Continuum, London, New York (2006)
15. Alston, W.P.: Illocutionary acts and linguistic meaning. In: Tsohatzidis, S.L. (eds), pp. 29– 49 (1994)
16. Ruiz-Rube, C. M. Cornejo, J., Dodero, M., Garcia, V.: Development Issues on Linked Data Weblog Enrichment. In: 4th MTSR, pp. 235–246. Springer, Berlin, Heidelberg (2010)
17. Schiller, A., Teufel, S., Stöckert, C., Thielen, C.: Guidelines für das Tagging deutscher Textkorpora mit STTS. Technischer Bericht, Institut

für maschinelle Sprachverarbeitung, Universität Stuttgart und Seminar für Sprachwissenschaft, Universität Tübingen (1999)

18. Trevisan, B. Jakobs, E.-M.: Talking about mobile communication systems: Verbal comments in the web as a source for acceptance research in large-scale technologies. In: IPCC, pp. 93–100 (2010)

MELANIE NEUNERDT
INSTITUTE FOR THEORETICAL INFORMATION TECHNOLOGY,
RWTH AACHEN UNIVERSITY,
SOMMERFELDSTR. 24, 52074 AACHEN, GERMANY
E-MAIL: <NEUNERDT@TI.RWTH-AACHEN.DE>

BIANKA TREVISAN
INSTITUTE FOR THEORETICAL INFORMATION TECHNOLOGY,
RWTH AACHEN UNIVERSITY,
SOMMERFELDSTR. 24, 52074 AACHEN, GERMANY
E-MAIL: <B.TREVISAN @TK.RWTH-AACHEN.DE>

RUDOLF MATHAR
INSTITUTE FOR THEORETICAL INFORMATION TECHNOLOGY,
RWTH AACHEN UNIVERSITY,
SOMMERFELDSTR. 24, 52074 AACHEN, GERMANY
E-MAIL: <MATHAR@TI.RWTH-AACHEN.DE>

EVA-MARIA JAKOBS
INSTITUTE FOR THEORETICAL INFORMATION TECHNOLOGY,
RWTH AACHEN UNIVERSITY,
SOMMERFELDSTR. 24, 52074 AACHEN, GERMANY
E-MAIL: <E.M.JAKOBS@TK.RWTH-AACHEN.DE>

Information Retrieval and Information Extraction

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 91–106
RECEIVED 31/10/11 ACCEPTED 09/12/11 FINAL 13/06/12

Puzzle Out the Semantic Web Search

DORA MELO[1], IRENE PIMENTA RODRIGUES[2],
AND VITOR BEIRES NOGUEIRA[2]

[1] *Instituto Politécnico de Coimbra and CENTRIA, Portugal*
[2] *Universidade de Évora and CENTRIA, Portugal*

ABSTRACT

The increase in web popularity has created the demand for systems that help the users find relevant information easily. Question Answering systems made it possible to ask questions and retrieve answers using natural language queries, rather than the keyword-based retrieval mechanisms used by current search engines. In this paper we propose a Cooperative Question Answering System that integrates natural language processing, ontologies and information retrieval technologies in a unified framework. It accepts natural language queries and is able to return a cooperative answer based on semantic web resources. Our system resorts to ontologies not only for reasoning but also to find answers and is independent of prior knowledge of the semantic resources by the user. The natural language question is translated into its semantic representation and then answered by consulting the semantics sources of information. The system is able to clarify the problems of ambiguity and helps finding the path to the correct answer. If there are multiple answers to the question posed, they will be grouped according to their semantic meaning, providing a more cooperative answer to the user.

KEYWORDS: *Natural Language, Ontology, Question Answering, Semantic Web*

1 INTRODUCTION

The tremendous development in information technology led to an explosion of data and motivated the need for powerful yet efficient strategies for data mining and knowledge discovery. Ontologies and the semantic web [1] became a fundamental methodology to represent the conceptual domains of knowledge and to promote the capabilities of semantic question answering systems [2]. These systems by allowing search in the structured large databases and knowledge bases of the semantic web can be considered as an alternative or as a complement to the current web search. In ontology-based question answering system, the knowledge based data, where the answers are sought, has a structured organization, the question answer retrieval of ontology knowledge base provides a convenient way to obtain knowledge for use, but the natural language need to be mapped to the query statement of ontology.

There is a gap between users and the semantic web: it is difficult for end-users to understand the complexity of the logic-based semantic web. Therefore it is crucial to allow a common web user to profit from the expressive power of semantic web data models while hiding its potential complexity. There is a need for user-friendly interfaces that scale up to the web of data and support end-users in querying this heterogeneous information source. Consistent with the role played by ontologies in structuring semantic information on the web, ontology-based question answering systems allows us to exploit the expressive power of ontologies and go beyond the usual "keyword-based queries".

The increase in web popularity has created the demand for systems that help the users find relevant information easily. Question Answering systems made it possible to ask questions and retrieve answers using natural language queries, rather than the keyword-based retrieval mechanisms used by current search engines. Question answering systems provide concise answers to natural language questions posed by users in their own terminology [3]. Those answers must also be in natural language in order to improve the system's usability and provide a better user friendly interface.

In this paper we propose a cooperative question answering system that receives queries expressed in natural language and is able to return a cooperative answer, also in natural language, obtained from resources on the semantic web (ontologies and OWL2 descriptions). The system starts a dialogue whenever there is some question ambiguity or when it detects that the answer is not what the user expected. Our proposal includes deep parsing, use of ontologies and other web resources such as the WordNet [4] and the DBpedia [5]. (Deep parsing is directly based on property grammars. It consists, for a given sentence, in building all the possible subsets of overlapped elements that can describe a syntactic category. A subset is positively characterized if it satisfies the constraints of a grammar.)

Our goal is to provide a system that is independent of prior knowledge of the semantic resources by the user and is able to answer cooperatively to questions posed in natural language. The system maintains the structure of the dialogue and this structure provides a context for the interpretation of the questions, includes implicit content such as spatial and temporal knowledge, entities and information useful for the semantic interpretation, like discourse entities used for anaphora resolution, on finding what an instance of an expression is referring to.

This paper is organized as follows. First, in Section 2, we present a brief overview on cooperative question answering and some related work, highlighting the main differences to the proposed system.

Afterwards, in Section 3, we introduce the proposed system, describing the main components of its architecture. In parallel, we present an example as an illustration of the system functionality. Finally, in Section 4, we present the conclusions and the future work.

2 A Brief Overview on Cooperative Question Answering and Related Work

Question answering may be seen as the task of automatically answering a question posed in natural language. To find the answer to a question, a question answering system may use either a pre-structured database or a collection of natural language documents. Therefore, a question answering system provides precise answers to user questions by consulting its knowledge base.

In order to provide users with accurate answers, question answering systems need to go beyond lexical-syntactic analysis to semantic analysis and processing of texts and knowledge resources. Moreover, question answering systems equipped with reasoning capabilities can derive more adequate answers by resorting to knowledge representation and reasoning systems like Description Logic and Ontologies.

A query language for OWL based on Prolog is presented in [6]. The author proposes a way of defining a query language based on a fragment of Description Logic and a way of mapping it into Prolog by means of logic rules. An illustration of a question answering system for the Portuguese language that uses the web as a database, through meta-search on conventional search engines can be seen in [7]. This system uses surface text patterns to find answers in the documents returned by search engines.

Another example of a question answering system where domain knowledge is represented by an ontology can be found in [8]: it is presented an interface system for question answering Chinese natural language that runs through a natural language parser. In [9], the author illustrate the capabilities for cooperative response generation implemented in Artificial Intelligence systems.

PowerAqua [10] is a multi-ontology-based question answering system that takes as input queries expressed in natural language and is able to return answers drawn from relevant distributed resources on the semantic web. PowerAqua allows the user to choose an ontology and then ask natural language queries related to the domain covered by the ontology. The system architecture and the reasoning methods are completely domain-independent, relying on the semantics of the ontology, and the use of generic lexical resources, such as WordNet.

A cooperative answer [11] to a query is an indirect answer that is more helpful to the user than a direct, literal answer would be. A cooperative answer may explain the failure of a query to produce results and/or suggest follow-up queries. In the case where a query does produce results, a cooperative answer may provide additional information not explicitly requested by the user. Cooperative answers arose in the context of natural language question answering and they were originally motivated by the desire to follow the conventions of human conversation in human machine interactions performed in natural language.

Advanced reasoning for question answering systems raises new challenges since answers are not only directly extracted from texts or structured databases but also constructed via several forms of reasoning in order to generate answer explanations and justifications. Integrating knowledge representation and reasoning mechanisms allow, for example, to respond to unanticipated questions and to resolve situations in which no answer is found in the data sources. Cooperative answering systems are typically designed to deal with such situations by providing useful and informative answers. These systems should identify and explain false presuppositions or various types of misunderstandings found in questions.

The representation of questions with generalized quantifiers as in [12] allows the use of various natural language quantifiers like all, at least 3, none, etc. Moreover, the question evaluation also resorts to logic programming with constraints.

In [13] we find a declarative approach to represent and reason about temporal contextual information. In this proposal each question takes place in a temporal context and that context is used to restrict the answer.

The fundamental techniques for computing semantic representations for fragments of natural language and performing inference with the result are presented in [14]. The primary tools used are first-order logic and lambda calculus, where all the techniques introduced are implemented in Prolog. The authors also show how to use theorem provers and model builders in parallel to deal with natural language inference.

An overview of cooperative answering in databases is presented in [15]. A logic-based model for an accurate generation of intensional responses within a cooperative question answering framework is proposed by the author of [16]. The author developed several categories of intensional forms and a variable-depth intensional calculus that allows for the generation of intensional responses at the best level of abstraction and shows that it is possible to generate natural responses on a template basis.

The same author in [17] presents an approach for designing a logic based question answering system, WEBCOOP, that integrates knowledge representation and advanced reasoning procedures to generate cooperative responses to natural language queries on the web. This project was developed on a relatively limited domain that includes a number of aspects of tourism (transportation) and requires the development of a knowledge extractor from web pages (similarly to a knowledge extractor operating on passages resulting from an information retrieval component) and the elaboration of a robust and accurate question parser. The responses provided to users are built in web style by integrating natural language generation techniques with hypertexts in order to produce dynamic responses. Natural language responses are produced from semantic forms constructed from reasoning processes.

Our proposal is a friendly, simple and cooperative question answering system. The main difference is the cooperative way that it obtains and answers the natural language questions posed by the user. We interact with the user in order to disambiguate and/or to guide the path to obtain the correct answer to the query posted, whenever this is possible to do by the reasoner. We also use cooperation to provide more informed answers. The answers is presented in natural language and have to clarify what the system can infer about the question from the knowledge domain. Therefore, the cooperative answer provided by our system has to explain the failure of a query to produce results and/or suggest follow-up queries. In the case where a query does produce results, the cooperative answer genetrated by our system will provide additional information not explicitly requested by the user.

3 The Proposed System Architecture

This section presents the architecture and functionality of our system. In the system model the main components are: Semantic Interpretation, Ontology Discovery, Semantic Evaluation and Discourse Controller.

Very briefly, the proposed system receives a natural language question and translates into a semantic representation using Discourse Representation Structures[3] (DRS). Then, after consulting the semantics sources of information, provides a natural language answer. If there are multiple answers to the question posed (or to the similar questions for which DBpedia contains answers), they will be grouped according to their semantic meaning, providing a more cooperative, informative and clear answer to the user. Therefore, we consider that our system provides a user friendly interface.

The framework to develop our system was Prolog with several extensions and libraries. Among the reasons for such choice is the fact that there is a wide range of libraries for querying and processing of ontologies OWL2, WordNet has an export for Prolog and there are extensions that allow us to incorporate the notion of context into the reasoning process. Moreover, Wielemaker [18] provides a study for query translation and optimization more specifically the SeRQL RDF query language, where queries are translated to Prolog goals, optimized by reordering literals. Finally, in [19] the authors describe how to develop a semantic web application entirely in Prolog.

Our system architecture is presented in Figure 1 and to help its understanding we describe the main components in the following subsections.

3.1 *The Semantic Interpretation Module*

Semantic analysis (or interpretation) is built using first-order logic [20] extended with generalized quantifiers [21]. We take special care with the discourse entities in order to have the appropriate quantifier introduced by the determinant interpretation. At this step, the syntactic structure of the question is rewritten into a DRS, that is supported by Discourse Representation Theory [22].

[3] For us a DRS is a set of referents, universally quantified variables and a set of conditions (first-order predicates). The conditions are either atomic (of the type $P(u_1, ..., u_n)$ or $u_1 = u_2$) or complex (negation, implication, disjunction, conjunction or generalized quantifiers).

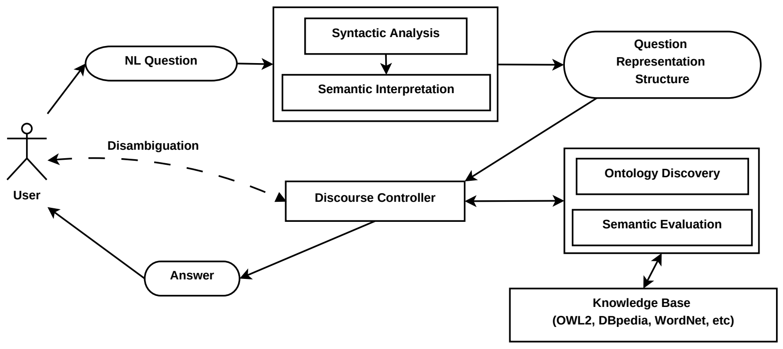


Fig. 1. Question Answering System Architecture

The implementation of this component follows an approach similar to the one for constructing a question answering system over documents databases proposed in [23]. The system consists of two separated modules: preliminary analysis of the documents (information extraction) and processing questions (information retrieval). This system is looking for processing the corpus and the questions, supported by theories of computational linguistics: syntactic analysis (grammatical restrictions) using deep parsing, followed by semantic analysis using the theory of discourse representation and finally the semantic (pragmatic) interpretation using ontology and logical inference.

As an illustration, consider the question ”All French romantic writers have died?”. The syntactic analysis generates a derivation tree, obtained from grammatical interpretation, that is rewritten according to a set of rules and integrated into a DRS, expressed in Prolog facts. In our study, it is stated by the following representation structure:

```
drs([all-X, exist-Y],
  [writer(Y), french(Y), romantic(Y), die(X)],
    [is(X,Y)]).
```

where the referent of the discourse is `all-X`, with `X` being a universally quantified discourse entity; the main predication of the question is `is(X,Y)`; the presupposed predications are `writer(Y)`, `french(Y)`, `romantic(Y)`, `die(X)`, with `Y` being an existential quantified discourse entity. The system has to find and check, for those entities `Y` that verify all the question presupposed conditions, if all entities `X` (that are entities `Y`) verify the main predication condition. If this is true, the answer to the question will be affirmative and, in order to provide a more informative answer, the system also present a list with all french, romantic, writers resource entities that died.

3.2 *The Ontology Discovery Module*

The Ontology Discovery is guided by the Discourse Controller to obtain the extension of sentence representation along with the reasoning process. The reasoning context and the question meaning will change whenever the Discourse Controller reaches a dead end.

This system module looks for similarities between labels according to their string-based, taking into account abbreviations, acronyms, domain and lexical knowledge. To maximize recall, the ontology search looks for classes, properties or instances that have labels matching a search term either exactly or partially and, if an answer is not achieved, each term in the query is extended with its synonyms, hypernyms and hyponyms obtained from WordNet [24]. Afterwards we extract a set of semantic resources which may contain the information requested.

Continuing the example of the previous section, in order to obtain the extension of sentence representation along the reasoning process, the system has to find the classes, properties or instances that have labels matching the search terms “writer”, “french”, “romantic” and “died”, either exactly or partially.

For instance, concerning the term “writer”, the system finds the DBpedia class `Writer`,[4] with property domain `Work` and domain range `Person`. These domains inform the system of the class properties and can confirm whether this is related with the question, if not will be thrown away and a new search will be made. For instance, at the grammatical interpretation step, one of the presupposition found was that the entities that verify the question have to be persons. So, if the class `Writer`, does not have a relation with the class `Person`, or can’t be applied to persons, at the phase of semantic interpretation it wouldn’t be added to the set of facts that represent the information provided by the question and wouldn’t be considered in the construction of the answer. The DBpedia class `birthPlace`[5] (an entity of type `ObjectProperty`, with property domain `Person` and domain range `Place`) that represents the place where some person was born, can represent the term

[4] `http://dbpedia.org/ontology/Writer`
[5] `http://dbpedia.org/ontology/birthPlace`

“french”. This term is also interpreted as a “person of France” and has as a direct hypernym the term “country” (obtain from WordNet), so the system also has to find the classes, properties or instances of all similar meanings to the initial term that could lead the system to the correct answer. Regarding the term “romantic”, the system finds the DBpedia resource `Romanticism`[6] (an entity of type `Thing`, an instance of property `movement` [7]). Finally, the DBpedia has a class `deathDate` [8] (an entity of type `DatatypeProperty`, with property domain `Person` and domain range `date`) that represents the death date of a person. The relation between the terms “die” and “death” can be made by searching the WordNet, where the term “die” can be interpreted as a “decease,” that in turn have as synonym the term “death”.

The next step is the construction of query(ies) needed to verify the initial question. If the question does not have an answer, a set of similar questions is constructed. Querying the WordNet, the system obtains similar terms to those that compose the initial question. This set of similar questions will enrich the knowledge domain and helps the interpretation of the original question or in the construction of its answer. If this set of new questions leads the system to different answers, we are in the presence of an ambiguity and the user is invoked to clarify it. If the system did not find any correspondence to a word and its derivatives, the user is informed and can clarify the system by reformulating the question or presenting other query(ies).

3.3 *The Semantic Evaluation Module*

Semantic evaluation is intended to be the pragmatic evaluation[9] step of the system, where the question semantic is transformed into a constraint satisfaction problem. This is achieved by adding conditions that constrain the discourse entities. Moreover, this extra information (regarding the question interpretation) can help the Discourse Controller to formulate a more objective answer.

[6] `http://dbpedia.org/resource/Romanticism`
[7] `http://dbpedia.org/property/movement`
[8] `http://dbpedia.org/ontology/deathDate`
[9] The pragmatic evaluation is the capability of to judge or calculate the quality, importance, amount or value of problems solutions that are solved in a realistic way which suits the present conditions rather than obeying fixed theories, ideas or rules.

The semantic evaluation must reinterpret the semantic representation of the sentence based on the ontology considered in order to obtain the set of facts that represent the information provided by the question. Therefore, the process responsible for the semantic evaluation receives the DRS of the question and interprets it in a knowledge base with rules derived from the ontology and the information contained in the knowledge base like DBpedia and WordNet.

Back to our example, to solve the constraint problem the Dialogue Controller generates and poses questions such "Who are the French romantic writers?" to the question answering system, whose representation structure is

```
drs([wh-X,exist-Y],
  [writer(Y), french(Y), romantic(Y), person(X)],
    [is(X,Y)]).
```

First and according to the domain knowledge, the interpreter will transform the conditions of the DRS into OWL, i.e., will construct the relative conditions based on the ontology. For instance, the condition `ontology_writer` will represent the DRS condition `writer`. Therefore, the new representation structure[10] for the question is

```
  drs([wh-X,exist-Y],
    [ontology_writer(Y), ontology_french(Y),
      ontology_romantic(Y), ontology_person(X)],
        [is(X,Y)]).
```

After obtaining this new set of DRS, the terms of the ontology will be interpreted as usual Prolog predicates. Then, by applying the unification mechanism of Prolog the system will obtain the following entities that verify the question: Francois-Rene de Chateaubriand, Alphonse de Lamartine, Alfred de Musset, Victor Hugo and Henri-Marie Beyle, Stendhal.

3.4 *The Discourse Controller Module*

The Discourse Controller is a core component that is invoked after the natural language question has been transformed into its semantic repre-

[10] The condition `ontology_term` represents the class, property or instance in the ontology that is the meaning of the term. If the interpreter has more than one possible ontology conditions for each term then will get several DRS rewritten with the terms of the ontology.

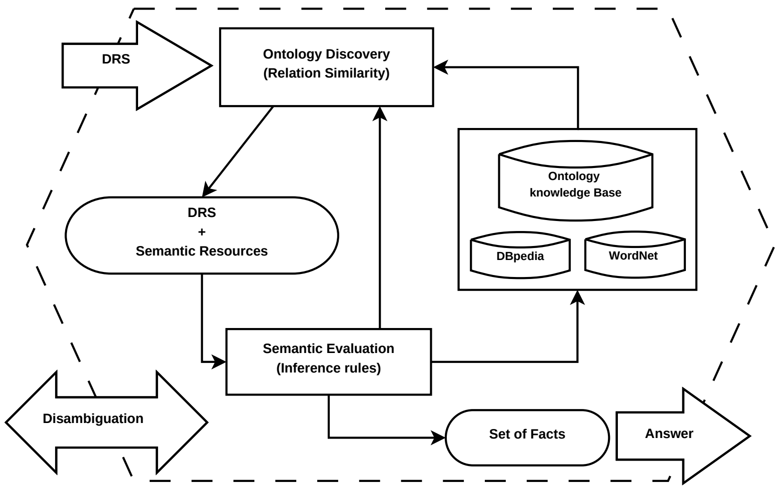


Fig. 2. Discourse Controller Module Architecture

sentation. Essentially the Discourse Controller tries to make sense of the input query by looking at the structure of the ontology and the information available on the semantic web, as well as using string similarity matching and generic lexical resources (such as WordNet).

In Figure 2, we represent the architecture of the Discourse Controller. In outline, after transforming the natural language question into its semantic representation the Discourse Controller is invoked and controls all the steps until the end, i.e until the system can return an answer to the user. More specifically, the Ontology Discover is invoked in order to provide the extension of sentence representation. If the ontology representation of a term is not found, the Discourse Controller is alerted and the user is called to clarify it. When the extension of the sentence representation is complete, the Discourse Controller adds to his knowledge a set of semantic resources.

Afterwards, the Semantic Evaluation is invoked. In this step, the question semantic is transformed into a constraint satisfaction problem, by adding conditions that constraint the discourse entities. This extra information can help the Discourse Controller to formulate a more objective answer. If in the interpretation of all the information leads the Discourse Controller to an empty answer or to multiple answers, the user is called to clarify it and may be necessary to re-invoke the Ontology Discover.

The process is finalized when the Discourse Controller is able to return an answer to the question posed by the user.

The Dialogue Controller deals with the set of discourse entities and is able to compute the question answer. It has to verify the question presupposition, choose the sources of knowledge to be used and decide when the answer has been achieved or to iterate using new sources of knowledge. The decision of when to relax a question in order to justify the answer and when to clarify a question and how to clarify it also taken by in this module.

Whenever the Discourse Controller isn't sure how to disambiguate between two or more possible terms or relations in order to interpret a query, it starts a dialogue with the user and asks him for disambiguation. The clarification done by the user will be essential for the Discourse Controller, this way obtaining the right answer to the query posed by the user. For instance, the question "Where is the Taj Mahal?", 'Taj Mahal' could be mapped into the name of a Mausoleum, a Casino Hotel or an Indian Restaurant and only the user can clarify about the intended meaning. The more cooperative and interactive the Discourse Controller is, the closer it will be to the correct answer.

Another important aspect of the Discourse Controller is to provide a friendly answer to the user. The answer should be as close as possible to the natural language. For instance, the question answering system has to respond "yes" or "no" when the user posed the query "Is Barack Obama the President of the USA?". In this case, the answer will be "yes". However, the answer must be more informative for the user. Some concepts are defined in the temporal context, even if implicitly, and the answer should be more clear and informative. For instance, the term 'President', in the context of the question, is defined as the title of head of state in some republics and has an associated duration for the mandate, a start date (date of election, date on taking office), and an end date of the mandate. So the answer to the question "Is Barack Obama the President of the USA?" should be "Yes, Barack Obama is the actual President of USA", that is more cooperative and informative.

For the cases where the answer to a question of type Yes/No is "No", the Discourse Controller will return a complete answer, clarifying the negation. If we consider the question "All the capitals of Europe have more than 200,000 inhabitants?" that has a "No" as an answer, the system will construct the proper answer that clarify the user and will return "No, 9 capitals of Europe have less than or equal to 200,000 inhabitants".

If there are multiple answers to the question posed by the user (or to the similar questions for which DBpedia contains answers), they will be grouped according to their semantic meaning, providing a more cooperative and clean answer to the user. To do so, the discourse controller has to reason over the question and construct the answer. For the question ”Where is the Taj Mahal?”, the user is called to clarify the system about the ambiguity of the question: Taj Mahal is a Mausoleum, a restaurant or Casino Hotel; and consider that the user is not able to clarify it or he simply wants that the system returns all possible answers. When the system has all the answers to all possible interpretations for the question posed by the user, the Discourse Controller will list the answer according to their semantic meaning:

```
Mausoleum Taj Mahal is in Agra, India
Casino hotel Taj Mahal is in Atlantic City, NJ,
     USA
Indian Restaurant Taj Mahal is in New Farm,
     Brisbane, Australia
Indian Restaurant Taj Mahal is in 7315 3rd Ave.,
     Brooklyn, NY, USA
```

Our dialogue system has as main objective the use of interaction to obtain more objective and concrete answers. It is not used only to clarify the problems of ambiguity, but also to help finding the path to the correct answer. Making the dialogue system more cooperative makes one able to get closer to the answer desired by the user. In many cases, the user is the only one who can help the system in the deduction and interpretation of information.

4 Conclusions and Future Work

We presented a proposal of a cooperative semantic web question answering system that receives queries expressed in natural language and is able to return a cooperative answer, also in natural language, obtained from semantic web resources (ontologies and OWL2 descriptions). The system is able of dialoguing when the question has some ambiguity or when it detects that the answer is not what user expected. Our proposal includes deep parsing and the use of ontologies and other web resources such as the WordNet and the DBpedia.

As future work, we intend to answer questions that are more elaborate and/or more difficult. Moreover, we also plan to extend to the Portuguese natural language. For this purpose it will be necessary to enrich the knowledge domain with concepts that may be deduced from the initial domain. Although the system is intended to be domain independent, it will be tested in a number of domains, with special relevance to the wine and the movies, since for these fields there are many resources available in the semantic web. We also plan to build a DRS generator, that builds the question semantics and retains additional information that allows the Discourse Controller to provide a more adequate and informed answer. We contemplate about enlarging the knowledge base with other ontologies in order to support open domain question answering and take advantage of the vast amount of heterogeneous semantic data provided by the semantic web.

REFERENCES

1. Horrocks, I.: Ontologies and the semantic web. Communications of the ACM **51**(12) (December 2008) 58
2. Guo, Q., Zhang, M.: Question answering based on pervasive agent ontology and Semantic Web. Knowledge-Based Systems **22**(6) (August 2009) 443–448
3. Hirschman, L., Gaizauskas, R.: Natural language question answering: The view from here. Natural Language Engineering **7**(4) (2001) 275–300
4. Fellbaum, C.: WordNet: An electronic lexical database. The MIT press (1998)
5. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J.: Dbpedia: A nucleus for a web of open data. The Semantic Web **4825**(Springer) (2007) 722–735
6. Almendros-Jiménez, J.M.: A Prolog-based query language for OWL. Electronic Notes in Theoretical Computer Science **271** (March 2011) 3–22
7. Rabelo, J., Barros, F.: Pergunte! uma interface em português para pergunta-resposta na web. Master's thesis, Informatics Center, Federal University of Pernambuco, Brazil (2004) 1114–1117
8. Guo, Q.: Question answering system based on Ontology. In: Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on, IEEE (2008) 3347–3352
9. Wahlster, W.: Cooperative access systems. Future Generation Computer Systems **1**(2) (1984) 103–111
10. Lopez, V., Motta, E.: Poweraqua: Fishing the semantic web. Semantic Web: Research and Applications (2006)
11. Corella, F., Lewison, K.: A brief overview of cooperative answering. Journal of Intelligent Information Systems **1**(2) (October 2009) 123–157

12. Rodrigues, I., Quintano, L., Ferreira, L.: Nl database dialogue question-answering as a constraint satisfaction problem. In: 18th Intl. Conf. on Applications of Declarative Programming and Knowledge Management (INAP'09), Univ. Évora (November 2009)
13. Nogueira, V., Abreu, S.: Temporal contextual logic programming. Electronic Notes in Theoretical Computer Science **177** (2007) 219–233
14. Blackburn, P., Bos, J.: Representation and inference for natural language: A first course in computational semantics. Center for the Study of Language and Information (2005)
15. Minker, J.: An overview of cooperative answering in databases. Flexible Query Answering Systems (1998) 282–285
16. Benamara, F.: Generating intensional answers in intelligent question answering systems. Natural Language Generation (2) (2004) 11–20
17. Benamara, F.: Cooperative question answering in restricted domains: the WEBCOOP experiment. In: Proceedings of the Workshop Question Answering in Restricted Domains, within ACL. (2004)
18. Wielemaker, J.: An optimised Semantic Web query language implementation in Prolog. Logic Programming (2005) 128–142
19. Wielemaker, J., Hildebrand, M., van Ossenbruggen, J.: Using Prolog as the fundament for applications on the semantic web. Proceedings of ALP-SWS2007 (2007) 84–98
20. Hodges, W.: Classical logic I: first-order logic. The Blackwell guide to philosophical logic (2001) 9–32
21. Barwise, J., Cooper, R.: Generalized quantifiers and natural language. Linguistics and philosophy **4**(2) (1981) 159–219
22. Kamp, H., Reyle, U.: From Discourse to Logic. Volume 42 of Studies in Linguistics and Philosophy. Kluwer (1993)
23. Quaresma, P., Rodrigues, I., Prolo, C., Vieira, R.: Um sistema de Pergunta-Resposta para uma base de Documentos. Letras de Hoje **41**(2) (2006) 43–63
24. Witzig, S., Center, A.: Accessing WordNet from Prolog. Artificial Intelligence Centre, University of Georgia (2003) 1–18

DORA MELO
ISCAC,
INSTITUTO POLITÉCNICO DE COIMBRA AND CENTRIA,
PORTUGAL
E-MAIL: <DMELO@ISCAC.PT>

Irene Pimenta Rodrigues
Departamento de Informática,
Universidade de Évora and CENTRIA,
Portugal
E-mail: <ipr@di.uevora.pt>

Vitor Beires Nogueira
Departamento de Informática,
Universidade de Évora and CENTRIA,
Portugal
E-mail: <vbn@di.uevora.pt>

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 107–119
RECEIVED 24/10/11 ACCEPTED 09/12/11 FINAL 13/06/12

From Co-occurrence to Lexical Cohesion for Automatic Query Expansion

HAZRA IMRAN[1] AND ADITI SHARAN[2]

[1]*Jamia Hamdard, India*
[2]*Jawaharlal Nehru University, India*

ABSTRACT

Designing an efficient Information Retrieval System (IRS) is still a big challenge and an open research problem. To overcome some of the problems of Information retrieval system, researchers have investigated query expansion (QE) techniques to help users in formulating better queries and hence improve efficiency of an Information Retrieval System. The scope of this work is limited to Pseudo Relevance Feedback based Query Expansion. Most of the work done in Pseudo Relevance Based Automatic query expansion is based on selecting the terms using co-occurrence based measures, which has some inherent limitations. Keeping in view limitations of co-occurrence based query expansion; we have tried to explore the utility of lexical based measures for expanding the query. This paper investigates the use of query expansion based on lexical links and proposes an algorithm for Lexical Cohesion Based Query Expansion (LCBQE). Based on theoretical justification and intensive experiments on TREC data set, we suggest that lexical based methods are at least as good as co-occurrence based measures and in some cases may work better than co-occurrence based measures. Depending on the nature of query, lexical based measures have great potential for improving the performance of an information retrieval system.

KEYWORDS: *Information Retrieval System, Pseudo Relevance Feedback, Automatic Query Expansion, Lexical Cohesion, Lexical Links.*

1 INTRODUCTION

An information retrieval system (IRS) is built to satisfy needs of a wide variety of users. Main objective of an IRS is to return maximum number of relevant documents corresponding to the user query, while retrieving minimum number of non relevant documents. However there are many problems in designing efficient IRS such as subjectivity, word mismatch problem and short query. Query expansion has been widely investigated as a method for improving the performance of information retrieval [8,11,12,18]. Theoretically, after query expansion, the performance of an IRS should improve. But practically, this is not always the case. Expanding a query may sometimes introduce a risk of query drift, in which the topicality of the original query may be changed, taking search into a different direction. Therefore a thorough research in Field of Query expansion is desired.

Automatic Query Expansion (AQE) refers to techniques that modify a query without user assistance. We have worked on automatic Query Expansion using Pseudo Relevance Feedback (PRF) which is similar to user relevance feedback but might be done without assistance from the user. (i.e., the approach might be fully automatic) [6]. A PRF based Automatic Query Expansion assumes that the top documents returned by the initial query are relevant ("pseudo-relevance feedback") and expansion terms are extracted from these top-ranked documents. Most of the work done in PRF based automatic query expansion is based on selecting the terms using co-occurrence based measures, which has some inherent limitations.

Keeping in view these limitations we tried to explore the use of lexical cohesion based methods for query expansion. The aim of this work was twofold: to understand the relationship between two document properties: its lexical cohesion and relevance to a query, secondly to investigate whether words that form lexical (cohesive) links between the contexts of query term instances in a document are also good query expansion (QE) terms.

The paper is organized as follows: in the next section (Section 2) basics of lexical cohesion and related terminology is introduced. In Section 3, the application of lexical cohesion for PRF based Automatic Query expansion is discusses and an algorithm for the same is provided. Section 4 presents experiments and their results. These results are analyzed thoroughly to evaluate the effectiveness of proposed method. Finally, Section 5 is dedicated for conclusion and provides suggestions for future work.

2 LEXICAL COHESION AND RELATED TERMINOLOGY

Lexical Cohesion is the cohesion that arises from semantic relationships between words. Segments of text, which are about the same or similar subjects, have higher lexical cohesion, i.e., share a larger number of semantically related or repeating words, than unrelated segments. The strength of lexical cohesion between two words can be useful in determining whether words are used in related contexts. Hence lexical cohesion has been used in identifying collocates. Sinclair and Jones [14] were the first to attempt corpus-based analysis of collocations based on lexical links. The major notions of collocation analysis were introduced in Sinclair [13] and systemized further in terms of 'node', 'collocate', 'window' and 'span'. A 'node' is defined as "an item whose total pattern of co-occurrence with other words is under examination". While a 'collocate' is "any item which appears with the node within a specified window." The term 'span' is used to refer to the stretch of text (with a predefined window size) around the node within which words are considered to be its collocates.

The identified collocates, as discussed above can be used to find the links between the words. Hoey used the term 'link' to denote an instance of repetition. A single instance of a lexical cohesive relationship between two words is usually referred to as a lexical link [1,5,9]. A lexical link is a relationship between two instances of the same lexeme (simple lexical repetition), its morphological derivatives (complex lexical repetition) or semantically related words (such as hyponyms, synonyms, meronyms, etc).

3 PSEUDO RELEVANCE FEEDBACK BASED QUERY EXPANSION BASED ON CO-OCCURRENCE AND LEXICAL COHESION

3.1 *Co-Occurrence based Query Expansion*

In the majority of works on pseudo-relevance feedback-based automatic query expansion, co-occurrence based approach has been used for selecting query expansion terms. These are the terms that are most frequently co-occurring with the query. Co-occurrence aspects can be captured in different ways. Two methods for extracting terms are used in this paper: one is based on Jacquard coefficient of co-occurring terms and another based on frequency of co-occurring terms [10].

The in depth analysis of co-occurrence based query expansion shows mix chances of success or failure. Thus major drawbacks of co-occurrence based automatic query expansion were investigated. One of the important limitations of co-occurrence based query expansion is that it selects frequently used terms for expansion. A frequently occurring term generally does not allow discriminating between relevant and irrelevant documents, so it is not good for expansion. If the co-occurring terms are selected from top ranked documents, discrimination does occur to a certain extent. However still there are chances that a term that is frequent in top n relevant documents is also frequent in entire collection.

Another limitation of co-occurrence based measure is that terms co-occurring with individual query terms are selected first. These results are then combined to select final expansion terms. In such cases if some query term dominates, most of the co-occurring terms may come from subset of these terms. In most of the cases it may be desirable to select those terms which are co-occurring with most of the query terms and at different instances of occurrence of query terms. This aspect can be captured by lexical links. Therefore, motivation of this work was to explore the use of lexical links for automatic query expansion.

3.2 *Query Expansion based on Lexical Links*

Lexical links [1,5,9] have been found useful in finding words that are related in certain context. However, most of the IRSs make use of lexical relations to a limited extent. The basic research question of this work is whether the use of lexical cohesion and lexical links for query expansion can improve performance of information retrieval. The use of lexical cohesion for query expansion is based on following intuition. A user query is likely to describe a relevant topic. Therefore in a relevant document terms correspond to same topic and they tend to cohere with each other and have similar collocation environment. Whereas in a non-relevant document, the occurrence of these query terms is not motivated by the presence of a relevant topic, but due to other factors, therefore they are less likely to occur in the same semantic context.

The query expansion methods presented in this paper rely on the method of calculating lexical cohesion between query terms' contexts in a document introduced by Vechtomova [16, 17]. In Vechtomova [16] it was suggested that simple lexical repetition alone performed as well as the use of repetition plus semantically related words (determined using Word Net).Their observation was that much difference was not found in the above two cases. Therefore in this work, simple lexical repetition is used to identify the lexical links.

Our main interest was in finding those terms from the top N relevant documents which are lexically cohesive to the context of query terms. Lexical cohesion between query terms' contexts is calculated by counting the number of lexical links between them. The context of a query term in a document is defined as a set of stemmed non-stop terms extracted from fixed-sized windows around each occurrence of the query term in the document. All the context terms of a query term are combined to form collocates for that query term. Collocation environments of the query terms were then compared to find terms having lexical links. The terms having lexical links were termed as link terms. These link terms were then re-ranked based on number of lexical links (Equation 1) and top N link terms were used to reformulate the initial query.

The weighting criteria used in this paper is also innovative in a sense that in most of the work studied by us the weights are given to expanded query terms based on their weight in document and not on the importance of the expanded query term with respect to original query term. In this work weights were assigned to expanded query terms basing on rank of the expanded query term, giving more importance to those terms that are more meaningful. For the expanded terms the following criteria was used for weighting k^{th} expansion term basing on number of lexical links:

$$weight(t_k) = \frac{number\_of\_lexical\_links(t_k)}{\max\limits_{1 \le i \le m}(number\_of\_lexical\_links(t_i))} \qquad (1)$$

Once weights are given to expanded query a new vector $\vec{eq}$ containing expansion terms can be constructed. Now the new query can be represented as

$$\vec{q} = \vec{q} + \vec{eq} \qquad (2)$$

Note that $\vec{eq}$ does not contain any of the original query terms.

On the basis of above motivation following idea we propose following algorithm for Lexical Cohesion based query expansion (LCBQE). The input to the algorithm is: documents and query; output is the expanded query. The algorithm is as follows:

```
– Represent documents and query in vector space model.
– Using tf-idf weighting scheme, give weights to the document and
  query terms.
– Match the query terms with the documents using okapi similarity
  measure.
– Retrieve the top $N$ documents corresponding to the initial query.
– For each top $N$ documents do:
      For each query term $q_i$ do:
      – Identify a snippet around each instance that contains 3 non-
        stop words before and after it in document.
      – Add the snippet to the collocates of the query term.
– Find all link-terms from the collocates of all individual query terms.
– Rank the link terms by idf.
– Extract the top $N$ link terms.
– Assign weight to the link term based on the number of links.
– Expand the initial query with these extracted link terms.
– Return the expanded query.
```

4 Experiments and Results

Experiments were conducted on TIPSTER document collection,(TREC data set) a standard test collection in the IR community. Volume 1 is a 1.2 GByte collection of full-text articles and abstracts. The documents came from the following sources:

- WSJ – Wall Street Journal (1986, 1987, 1988, 1989, 1990, 1991, and 1992),
- AP – AP Newswire (1988,1989 and 1990),
- ZIFF – information from Computer Select disks (Ziff-Davis, Publishing),
- FR – Federal Register (1988),
- DOE – short abstracts from Department of Energy.

For queries: 50 queries were used that were formed through 50 TREC topic sets. Average word length for query was 2.3 words.

4.1 *Experiments using Co-occurrence and Lexical Cohesion Based Query Expansion*

The purpose of experiments was to find out the usefulness of the lexical information for automatic query expansion and compare it with baseline and co-occurrence based query expansion. For Co-occurrence based query expansion Jaccard coefficient based and frequency based measures were used for selecting the expansion terms. The terms were then re-ranked with entropy based measure. For lexical based query expansion local collocates were extracted from the windows around every occurrence of the query terms in top *N* documents as discussed in above algorithm. Further link terms were extracted from local collocates. Top n link terms were used expanding the query. The weights of candidate expansion terms were assigned based on the number of lexical links.

Parameters used in the experiments were: number of top *N* documents, window size (for lexical cohesion based expansion), and number of top ranked terms used for expanding the query. Values of parameters were decided empirically after intensive experimentation. In our experiments we used top *N* documents = 100, window size as 3 non-stop words on either side of query term, and number of top ranked terms were 10. The results are shown through Table 1 and Figure 1.

Table 1. Results Without Query Expansion, Co-occurrence and Lexical Based Query Expansion

	Without query expansion	Co-occurrence Method		Lexical Link Method
		Jaccard	Freq	
Number of Queries (num_q)	50	50	50	50
Number of Retrieved Documents (num_ret)	5000	5000	5000	5000
Number of relevant documents (num_rel)	16386	16386	16386	16386
Number of relevant retrieved documents (num_rel_ret)	1156	1237	1220	**1256**
Map	0.0443	0.0602	.0592	**.0611**
gm_map	0.0079	0.0102	.010	0.01
Rprec	0.0973	0.1032	.1032	**0.1043**

Table 1 shows the result of experiments for the baseline (without query expansion), co-occurrence, and lexical cohesion based query expansion respectively. The best results are shown in boldface. The MAP of lexical is 0.0611, which is better than co-occurrence and base-line. Similarly, Rprec also shows better result when compared with other two methods.

Figure 1 shows the comparison of recall precision graph between unexpanded query and query expansion based on lexical links. From Figure 1 it is observed that expansion of initial query with statistically significant local collocates following pseudo relevance feedback results in significant performance improvement over unexpanded under the same conditions.

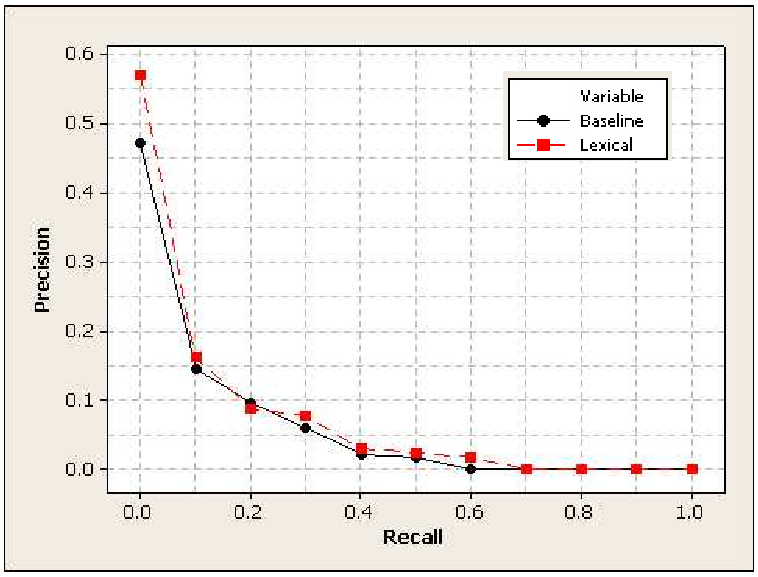


Figure. 1. Recall Precision Curve showing result of experiment for QE based on Base line and Lexical Cohesion

From paired t-test it can be concluded that the means differ at the 0.05 level of significance. The mean of lexical is significantly different from the baseline. (p<.005) The true difference is between 0.0078235 and 0.025844.

Apart from overall analysis of result an in depth analysis of result was done for individual queries, specially focusing on queries for which there was no significant improvement in result. Figure 2 shows the differences in average precision of individual queries expanded using LCBQE method from the unexpanded query. Here an attempt is to analyze the performance of individual queries from baseline to lexical method is done.

The results given in Figure 2 show that in a considerable number of cases lexical method yielded the most gain in performance (74% queries shows improvement). This suggests that there is a room for better improvement in the retrieval if lexical terms are used for expansion.

There result was analyzed for the queries for which result does not show significant improvement. Here, it was found that the lexical terms are not adding more meaning to the original query. Example in query 88 (*Crude oil Price Trends*) the lexical terms found were: *hishan, countries, megaprojects, 5084, 5209, 5273*. These terms are not adding significant meaning to the original query and hence the performance degrades.

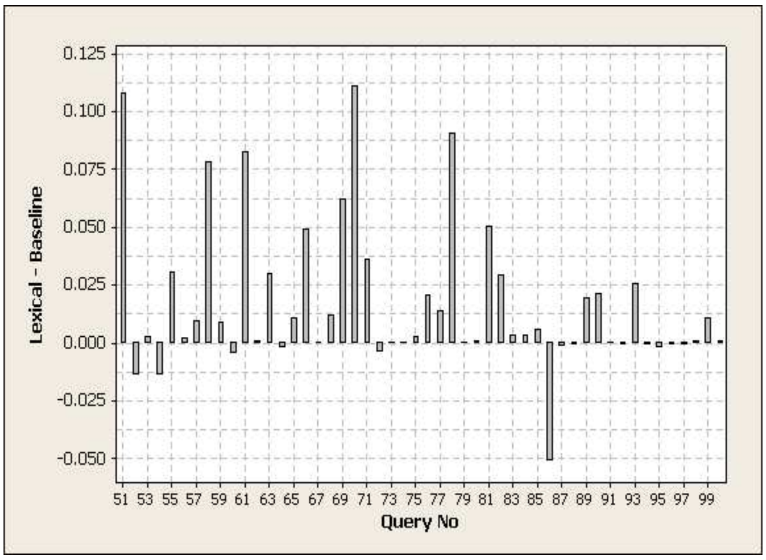


Figure. 2. Differences in average precision of individual queries after query expansion from the unexpanded query

It is observed that results obtained by lexical cohesion based methods are almost similar to those obtained by co-occurrence based automatic query expansion methods. As lexical cohesion based methods have not been explored much for automatic query expansion, the results of this work motivates us to perform in-depth analysis of these methods. An investigation into individual query performance presents another significant aspect of lexical based automatic query expansion method. This aspect as percentage of queries for which the result improved is the greatest in this method (74%). Therefore, it can be said that the chances of failure of this method in expanding the query are comparatively less than other methods.

4.2 *Analysis of Relation between Lexical Cohesion and the Relevance of Documents*

In order to gain some qualitative understanding of the relation between lexical cohesion and the relevance property of documents, the top N documents retrieved in initial run and retrieved after automatic query expansion (based on lexical links) were compared. It was examined on a small sample from the top *N* (100) documents. Although, it is not possible to generalize, experiments suggest that certain patterns of lexical links could be identified in the documents promoted and demoted after initial query expansion with lexically cohesive terms. In the set examined, it was noticed that documents that are promoted, contain most of the query terms, and there are several instances of each of them. It also appears that the instances of query terms are spread throughout the documents, i.e., they are not concentrated in isolated sections of the documents. As expected, the instances of query terms are well connected by lexical links in the promoted documents.

An example of this type of document is SJMN91-06252124, retrieved in response to query "Natural Language Processing" (query 66). There are many instances of the query terms (11 instances of "natural," 5 instances of "language" and 3 instances of Processing) in the text and they are extensively connected with each other by lexical links.

In the demoted documents, it was seen three different patterns. Some of the demoted documents are made up of disjoint pieces of text that cover separate and unrelated information. In second type of demoted document, query terms (such as "Natural Language Processing") occur but not all in the same context. In the third type of demoted documents, the query topic is treated marginally.

4.3 *Analysis of Terms Obtained for Co-Occurrence and Lexical Cohesion based AQE*

The quality of terms extracted with both the methods:-co-occurrence and lexical cohesion was analyzed. For example for Query 87 (*Criminal Actions Against Officers of Failed Financial Institutions*) the Co-occurring terms (*failed actions, enforcement, justice, charges, prosecutors, civil, convicted, investigation, alleged, attorney, jury*) are more relevant than the lexical terms (*officers, criminal, failed, actions, mijalis, hallada, khoo, neidorf, esm, lytl, zodiac, dmv, disheartening, mndrg*). However, this trend is only seen in 32% of queries. In rest 68% queries result is improving with lexical method. For example in query 67(Politically Motivated Civil Disturbances) the lexical terms (*pungently, hurbon, meacher, fitzhugh, corporacion, cispes, insinuated, mouawad, deceitful, amnesties*) appears to be more relevant than co-occurring terms (*liberties, unrest, violence, racial, criminal, rights, racially, blacks, riots, rioting*).

5 CONCLUSION

Most of the work done for PRF based query expansion till now has been based on using co-occurrence based measures for selecting expansion terms. Co-occurrence based query expansions has certain limitations. The research question investigated in the paper was whether the use of lexical link information can improve performance of PRF based Automatic Query Expansion and whether it can overcome some of the limitations of co-occurrence based query expansion.

This paper proposed a new method for query expansion: Lexical Cohesion Based Query Expansion (LCQBE) and provided an algorithm for the same. The experiments were done to compare the results with co-occurrence based query expansion. Experiments were performed on standard TREC data set. It was found that result of lexical based query expansion is generally as good as query expansion using co-occurrence methods, in some cases it is even better. As lexical cohesion based query expansion is a comparatively unexplored topic, there is an immense potential for exploring the usefulness of these methods in order to improve information retrieval efficiency.

An in depth analysis of the results obtained in this paper motivates us to work in new direction. In this work we may be able to find out different types of queries (categorized on some basis) for which one can guess that whether retrieval efficiency can be improved by co-occurrence based query expansion, lexical based query expansion, by none of them or both of them. An important contribution of this work can be that we can use a switch to decide a priori that which of the query expansion (co-occurrence or lexical) can be more useful for a specific query or the query should not be extended at all.

REFERENCES

1. Ellman, J., and Tait, J. (1998). Meta searching the web using exemplar texts: Initial results. Proceedings of the 20th BCSIRSG.
2. Halliday, M. A., and Hasan, R. (1976). Cohesion in English. Longman.
3. Hearst, M. (1994). Multi-paragraph segmentation of expository text. Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics.
4. Hirst, G., and St Onge, D. (1997). Lexical chains as representation of context for the detection and correction of malapropisms. MIT Press.
5. Hoey, M. (1991). Patterns of Lexis in Text. Oxford University Press.
6. Imran, H. and Sharan, A. (2010). A Framework for Automatic Query Expansion, Lecture Notes in Computer Science (LNCS), Volume 6318, 2010, Springer, 386–393.
7. Manabu, O., and Hajime, M. (2000). Query-biased summarization based on lexical chaining. Computational Intelligence, 578–585.
8. Molto, M., and Svenonious, E. (1991). Automatic Recognition of title page names. Information Processing and Management, 83–95.
9. Morris, J., and Hirst, G. (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text. Computational Linguistics, 21–48.
10. Rijsenberg, C. J. (1979). Information Retrieval (Second, Ed.) Butterworth Heinemann.
11. Sakai, T., and Robertson, S. E. (2001). Flexible pseudo-relevance feedback using optimization tables, Louisiana, 396–397.
12. Salton, G. (1998). Automatic text processing: the transformation, analysis, and retrieval of information by computer. Addison-Wesley.
13. Sinclair, J. M. (1991). Corpus, concordance, collocation. Oxford University Press.
14. Sinclair, J. M., and Jones, S. (1996). Lexis and Lexicography. Singapore: UniPress.
15. Stairmand, M. A. (1997). Textual context analysis for information retrieval. Procedings of the ACM SIGIR , 140–147.

16. Vechtomova, O. (2006). Noun phrases in interactive query expansion and document ranking. Information Retrieval, 399–420.
17. Vechtomova, O., Karamuftuoglu, M., and Robertson, S. E. (2006). On document relevance and lexical cohesion between query terms. Information Processing and Management, 1230–1247.
18. Witten, I., Moffat, A., and Bell, T. (1999). Managing Gigabytes: Compressing and Indexing Documents and Images. Morgan Kaufmann.

HAZRA IMRAN
DEPARTMENT OF COMPUTER SCIENCE,
JAMIA HAMDARD ,
NEWDELHI, INDIA
E-MAIL: <HIMRAN@JAMIAHAMDARD.AC.IN>

ADITI SHARAN
SCHOOL OF COMPUTERS AND SYSTEM SCIENCES,
JAWAHARLAL NEHRU UNIVERSITY,
NEW DELHI, INDIA
E-MAIL: <ADITISHARAN@MAIL.JNU.AC.IN>

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 121–136
RECEIVED 29/10/11 ACCEPTED 09/12/11 FINAL 15/06/12

Arabic Temporal Entity Extraction using Morphological Analysis

FADI ZARAKET AND JAD MAKHLOUTA

American University of Beirut, Lebanon

ABSTRACT

The detection of temporal entities within natural language texts is an interesting information extraction problem. Temporal entities help to estimate authorship dates, enhance information retrieval capabilities, detect and track topics in news articles, and augment electronic news reader experience. Research has been performed on the detection, normalization and annotation guidelines for Latin temporal entities. However, research in Arabic lags behind and is restricted to commercial tools. This paper presents a temporal entity detection technique for the Arabic language using morphological analysis and a finite state transducer. It also augments an Arabic lexicon with 550 tags that identify 12 temporal morphological categories. The technique reports a temporal entity detection success of 94.6% recall and 84.2% precision, and a temporal entity boundary detection success of 89.7% recall and 90.8% precision.

INTRODUCTION

This paper considers the problem of extracting temporal entities from Arabic text documents. Temporal entities are text chunks that express or infer temporal information. Some entities represent absolute time and dates such as 07/17/2011 or الخامس من آب ٢٠١٠ *ālḫāms mn ʾāb 2010* (August 5, 2010). Some entities represent relative time such as بعد خمسة أيام *bʿd ḫmst ʾayām* (after five days). Other entities represent temporal quantities such as عطلته ١٤ يوما *ʿṭlth 14 ywmā* (his vacation is 14 days).

Industry tools that extract temporal entities from Arabic texts exist [1,2,3]. However, the techniques underlying these tools have not been disclosed and evaluated academically yet. We present the *Arabic temporal entity extractor using morphological analysis* (ATEEMA). To the best of our knowledge, ATEEMA is the first open-source tool to perform the task of temporal entity extraction.

Temporal entity extraction from text includes the task of identifying the temporal chunks of text and then the task of normalizing a temporal chunk into a time quantity structure. Temporal entity recognition detects expressions that express time concepts. Temporal entity normalization understands the temporal chunck and extracts from it a time structure so that ١٩٠٠ *1900*, القرن التاسع عشر *ālqrn āltāsʿ ʿšr* (the nineteenth century), and ألف وتسعمئة بعد الميلاد *ʾalf wtsʿmʾyt bʿd ālmylād* (one thousand nine hundred after Christ) all have one normalized canonical form. Ultimately, one may want to store the normal form in a database and process it later.

Research on temporal entity extraction in those languages that use Latin alphabet, such as English, German, French, or Spanish, uses local grammars, finite state automata [4,5,6,7,8,9,10], and neural networks [11] to detect temporal entities. These techniques do not work well directly for Arabic due mainly to the rich morphology and high ambiguity rate of Arabic.

This paper focuses on the task of temporal entity recognition. We will target temporal entity normalization in future work. ATEEMA uses Arabic morphological analysis with part of speech (POS), gloss tagging, and augmented temporal tagging to capture morphological temporal features of Arabic text. ATEEMA passes the temporal morphological features to a knowledge-based finite state transducer that captures temporal entities and detect their boundaries.

In this paper we make the following contributions. **(1)** We present a novel technique for temporal entity extraction from Arabic text based on morphological analysis and finite state transducers. **(2)** We augment an Arabic lexicon with 550 temporal morphological tags that identify 12 separate temporal categories. And **(3)** We provide the first open source temporal entity extractor for Arabic.

Motivation

In this section we discuss several interesting applications that depend on temporal entity extraction to motivate our work. The detection of time stamps from the content of a digital document rather than its meta-data, such as the last modification and the creation dates of a file, is of interest to the research community. Li et al. from Microsoft filed a patent for their application that extracts authorship dates. They hypothesize that the last modification date of a document is not representative of its date of generation as documents may be uploaded or copied to collaborative websites and the meta-date gets changed to the upload date, which is rarely significant [11].

The work in [5] considers the problem of automatic assignment of event-time periods in documents such as newspaper articles, medical reports and legal documents to facilitate documemt retreival. Time periods of the events under consideration in a document may be of higher importance than the date of the writing of the document.

Extraction of temporal entities from news articles combined with a user profile can help augment the articles with information of interest to the user [4]. For example, the extracted temporal entities can form a navigation timeline that refers to other articles of relevance, and also help augment the articles with answers to queries such as “Where were I when the event took place?”, and “What other events took place at the same time in my neighborhood?”.

German researchers developed an appointment scheduling via emails (COSMA) application based on temporal entity extraction that takes as input the electronic calendar of several parties, automates the time consuming task of scheduling meetings, and reduces the number of correspondence needed to agree on a time [12].

When time entities are extracted, other data mining techniques can discover useful relationships such as the recognition of frequent temporal patterns in newspapers [13], the discovery of interesting events from time varying features in the news corpus [14], and the detection and tracking of new evolving events in the news [15].

Other benefits of temporal extraction include the comparison of parallel accounts and narrations of the same events such as several history books that address the same periods. With temporal entity extraction one can automatically check and detect historical inconsistencies if they exist. Extracted temporal entities can help to sort several different accounts of the same or similar events into a timeline. This can automate merging complementary and partial accounts of the same events. This analysis, such as merging the several narrations of the Bible, was performed manually.

Background

In this section we briefly describe morphological analysis, finite state transducers, and local grammars that are used to extract temporal entities from text.

Morphological analysis. Morphology considers the composition of a word from several *morphemes*. A morpheme is a *stem* or an *affix*. An affix is a *prefix, suffix,* or an *infix*. Prefixes and suffixes are attached to the beginning and end of the word respectively; while infixes introduce changes within the stem. A morpheme is associated with several tags such as *part of speech* (POS) and gloss tags. Words result from the concatenation of compatible morphemes. Morphological analysis helps to group together words which express similar notions such as شهر‘ شهرين‘ اشهر‘ الشهرين‘ الاشهر that all share a common stem with a gloss tag of 'month', and have affixes with useful glosses such as plural, dual, and definite article ال indicators.

Morphological analysis is is used in techniques that detect Arabic named entities such as proper names [16,17]. It is key and necessary in Arabic entity recognition due to the morphological richness of Arabic. For instance, consider the stem إنتهاء *ʾintihāʾ* (end) with the the suffix ه *-h* (it). Both variations إنتهائه *ʾintihāʾih* and إنتهاؤه *ʾintihāʾoh* are legal based on the context. In addition, morphology isolates clitics such as ف *fa* (so) and و *wa* (and) from other morphemes as in واليوم *wa-āl-yawm* (and + today).

Local grammars. *Local grammars* have been used to extract entities from text [4,5,7]. Local grammars are lexical and syntactic constraints expressed in regular expressions that precisely define the local neighborhood and context of an entity. The rules ignore the rest of the context such as the complete sentence, paragraph, and document. Local grammars simplify the detection of target entities when the entities can be expressed by local small scope rules. They relieve the user from the task of modeling the full language and the expense of understanding the whole context [18].

A local grammar for temporal English expressions includes several rules such as `(Num ≤ 12) "in the afternoon"` that can detect "five in the afternoon", `(Num ≤ 12) "p.m"` that can detect "5 p.m.", `"half past" (Num ≤ 12)` that can detect "half past one", and `(Num ≤ 23) ``:'' (Num ≤ 59)` that can detect "5:44". The quoted subexpressions within the rules are *frozen* expressions and `Num` captures numbers such as 'one', 'two', '1' and '2' [19].

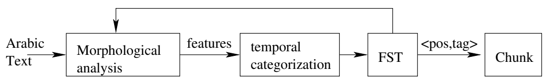


Fig. 1. ATEEMA flow diagram

Finite state transducers. A finite state transducer (FST) is a finite state machine with an input and an output tape. FSTs differ from finite state automata (FSA) in that they have an output tape while FSAs have accept states instead.

Formally, an FST is a tuple $M = (S, S_0, \Sigma, \Gamma, \delta)$ where S is the set of states, $S_0 \subset S$ is the set of initial states, Σ is the input alphabet, Γ is the output alphabet, and $\delta \subseteq S \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \times S$ is the transition relation. FSTs have been used extensively in text mining applications where the input is the text and the output is the delimiters of a chunk of text with an associated class [20]. FSTs are attractive due to their efficiency and ease of use.

ATEEMA

The diagram in Figure 1 shows how ATEEMA works. ATEEMA takes as input Arabic text and returns temporal entities therein. ATEEMA passes the input text string to an inhouse Arabic morphological analyzer. Both the analyzer and ATEEMA are both available as open source tools. The analyzer processes the input text and whenever it identifies a morpheme, it calls ATEEMA back with the current solution context and the found morpheme. Note that several morphological solutions may exist for the same input string.

The ATEEMA call back receives the solution context and the morpheme with the associated POS and gloss tags. It either adds the morpheme to a sequence of unresolved morphemes, or resolves the morphemes and produces a temporal category as input to the finite state transducer (FST). Note that this is necessary since words in the Arabic language are not necessarily separated by white space delimiters and thus one white space delimited token may contain several words and produce several categories.

The finite state transducer detects the temporal entities. ATEEMA uses a manually built finite state transducer to accomodate for **(1)** morphological variations which are frequent in Arabic text, and **(2)** ambiguous morphological solutions since Arabic is at least one order of mag-

Table 1. Temporal categories added to the lexicon of the Arabic morphological analyzer.

	TIME					*NUM*			*TIME_PREP*			
Tag	unit	relative	range	nominal	events	digit	word	range	point	relative	approximate	range
Count	32	55	16	94	10	20	152	-	83	54	12	22
Total	207					172			171			

nitude more ambiguous than Latin languages. For example, consider the following morphological variations of entities related to temporal expressions.

– آخِر، اخِر، أَواخِر، أَخِير *ʾāḫir, āḫir, ʾaawāḫir, ʾaaḫiyr* (last)
– بَدْء، بَدْأ، بَدْؤ، بَدْئ، ابْتِداء *badʾ, badʾa, badʾw, badʾy, ābtidāʾ* (start)

Also consider the ambiguous word إثنين *ʾiṯnyn* which means Monday and the number two. Techniques that use local grammars and work for Latin languages are restricted to regular expressions with frozen phrases and fail upon morphological variations. We believe that our approach can be extended to other morphologically rich languages as well as Latin languages to provide better results.

ATEEMA targets the detection of temporal expressions within the bounds of their temporal context. Consider the following examples.

– ٢٠٠٣ / ٩ / ٤ (4/9/2003): this is a straight forward date with no temporal context.
– بعد ٢٠٠٣ / ٩ / ٤ (*bʿd* after 4/9/2003): this is a date with a temporal preposition.
– في نيسان ٢٠٠٣ *fy nysān 2003* , (in April 2003): this is a partial date with a temporal preposition.
– بعد مرور نحو أربعة أشهر *bʿd mrwr nḥw ʾarbʿt ʾašhr* (after about four months passed): this is an approximate (نحو) range (مرور) with a temporal preposition.
– في ثمانينيات القرن الماضي *fy ṯmānynyāt ālqrn ālmāḍy* , (in the eighties of the last century): this is a temporal expression where the range is inferred from the suffix ـيات of ثمانينيات.

Temporal categories. The Arabic morphological analyzer produces morphemes with POS and gloss tags. ATEEMA is interested in temporal and numerical morphological features as well as temporal prepositions which occur within a neighborhood of temporal expressions. For this purpose we augmented the tags of the morphological analyzer to report the categories presented in Table 1.

Category *TIME* denotes explicit temporal tokens. They are infered from time units such as دقيقة *dqyqt* (minute) and ساعة *sāʿt* (hour), relative entities such as غد *ġd* (tomorrow), nominal entities as in day and month names, range entities as in seasons, and referential entities as in important events هجري *hǧry* . Category *NUM* denotes numerals and refers to digits or words such as ثمانين *ṯmānyn* (eighty), أربعة *ʾarbʿt* (four) and ٤ *4* . The prefixes and suffixes of these numbers may divide them into subcategories denoting range as in ثمانينيات. Category *TIME_PREP* denotes temporal prepositions that precede or follow time expressions. They are divided into relative prepositions such as قبل *qabil* (before) and منذ *mnḏ* (since), approximate prepositions such as نحو *nḥw* (about), range prepositions such as خلال *ḫlāl* (during), and point prepositions such as في *fy* (in). The Table reports also the number of lexicon items we tagged with temporal tags. We identified 550 lexicon entries out of 82,170 entries and annotated them with temporal tags. Note that these entries are stems and affixes and can be concatenated with other entities to generate more possible temporal tokens.

ATEEMA associates an order amongst the temporal morphological annotations and categories. ATEEMA uses the order to associate complex temporal entities with a tag that is the result of the concatenation of the morphological temporal tags and categories. This produces a semi-canonical temporal form that can be used in practice as a normalized temporal form.

The finite state transducer

Figure 2 shows a high level overview of the finite state transducer. The FST takes the detected temporal categories as input and also considers two state variables iw_t and iw_m that count the number of temporal categories met while in the *Time* and *Maybe Time* states, respectively. The actual number of states is proportional to θ_t and θ_m which are two thresholds that define the upper limits for the iw_t and iw_m counters, respectively.

The initial state *Nothing* denotes that no temporal entity is being processed. The *Maybe Time* state denotes the cases where the FST is not yet sure whether the currently processed text is a temporal entity. Finally the *Time* state denotes that the FST has recognized a temporal entity and is in the process of computing its boundary.

The transition to the *Maybe Time* state happens when the FST encounters prepositions or numbers but no direct time tokens. Once it detects a

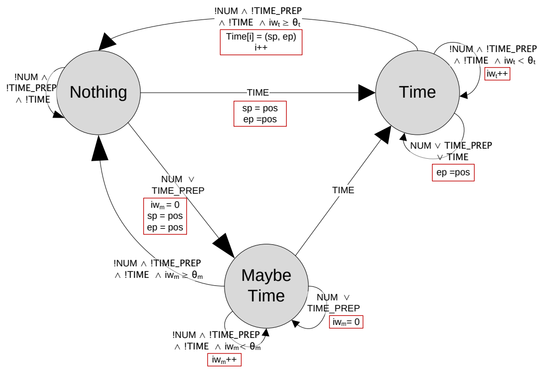

!NUM ∧ !TIME_PREP ∧ !TIME ∧ iwt ≥ θt
Time[i] = (sp, ep) i++
!NUM ∧ !TIME_PREP ∧ !TIME
Nothing
TIME
sp = pos ep =pos
Time
!NUM ∧ !TIME_PREP ∧ !TIME ∧ iwt < θt
iwt++
NUM ∨ TIME_PREP ∨ TIME
ep =pos
NUM ∨ TIME_PREP
iwm = 0 sp = pos ep = pos
TIME
!NUM ∧ !TIME_PREP ∧ !TIME ∧ iwm ≥ θm
Maybe Time
NUM ∨ TIME_PREP
iwm= 0
!NUM ∧ !TIME_PREP ∧ !TIME ∧ iwm< θm
iwm++

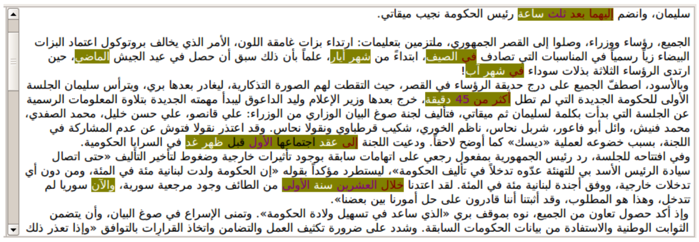
Fig. 2. FST for temporal entity extraction

direct time token, the FST transitions to the *Time* state. This might happen also directly from the *Nothing* state.

The two thresholds θ_m and θ_t encode the flexibility and tolerance of ATEEMA to recognize the several different temporal classes. The θ_m threshold specifies the number of non-temporal tokens that ATEEMA tolerates within a temporal entity. The threshold θ_m is tested before a transition takes place from the *Maybe Time* state to the *Nothing* state. In other words, if the FST suspected a temporal entity because of an indirect temporal feature (*NUM or TIME_PREP*), then the FST will give up that entity if it meets more than θ_m non-temporal tokens before meeting a direct temporal feature. For example, the word خمسين in the text دفع خمسين درهما ليشتري الحنطة والتمر *dfʿ ẖmsyn drhmā lyštry ālḥnṭt wāltmr* (He paid fifty Durhams to buy flour and dates) will activate a transition to a *Maybe Time* state but the rest of the text will drive the FST back to the *Nothing* state.

The θ_t threshold expresses the number of non-temporal words that might intervene between two temporal words in the same temporal entity. For example, the text بعد أربعة أشهر عجاف وخمس من أصعب السنوات *bʿd ʾarbʿt ʾašhr ʿǧāf wẖms mn ʾaṣʿb ālsnwāt* (after four bad months and five of the hardest years) contains non-temporal words (عجاف and من أصعب) that θ_t tolerates.

In Figure 2, actions that are performed on the different transitions are shown inside boxes near the transitions. The actions include setting the values of the counters iw_m, iw_t, and i. The counter iw_m denotes the number of non-temporal tokens met within the *Maybe Time* state. The



سليمان، وانضم إليهما بعد ثلث ساعة رئيس الحكومة نجيب ميقاتي.

الجميع، رؤساء ووزراء، وصلوا إلى القصر الجمهوري، ملتزمين بتعليمات: ارتداء بزات غامقة اللون، الأمر الذي يخالف بروتوكول اعتماد البزات
البيضاء زياً رسمياً في المناسبات التي تصادف في الصيف، ابتداءً من شهر أيار، علماً بأن ذلك سبق أن حصل في عيد الجيش الماضي، حين
ارتدى الرؤساء الثلاثة بذلات سوداء في شهر آب!
وبالأسود، اصطفّ الجميع على درج حديقة الرؤساء في القصر، حيث التقطت لهم الصورة التذكارية، ليغادر بعدها بري، ويترأس سليمان الجلسة
الأولى للحكومة الجديدة التي لم تطل أكثر من 45 دقيقة، خرج بعدها وزير الإعلام وليد الداعوق ليبدأ مهمته الجديدة بتلاوة المعلومات الرسمية
عن الجلسة التي بدأت بكلمة لسليمان ثم ميقاتي، فتأليف لجنة صوغ البيان الوزاري من الوزراء: علي قانصو، علي حسن خليل، محمد الصفدي،
محمد فنيش، وائل أبو فاعور، شربل نحاس، ناظم الخوري، شكيب قرطباوي ونقولا نحاس. وقد اعتذر نقولا فتوش عن عدم المشاركة في
اللجنة، بسبب خضوعه لعملية «ديسك» كما أوضح لاحقاً. ودعيت اللجنة إلى عقد اجتماعها الأول قبل ظهر غد في السرايا الحكومية.
وفي افتتاحه للجلسة، رد رئيس الجمهورية بمفعول رجعي على اتهامات سابقة بوجود تأثيرات خارجية وضغوط لتأخير التأليف «حتى اتصال
سيادة الرئيس الأسد بي للتهنئة عدّوه تدخلاً في تأليف الحكومة»، ليستطرد مؤكداً بقوله «إن الحكومة ولدت لبنانية مئة في المئة، ومن دون أي
تدخلات خارجية، ووفق أجندة لبنانية مئة في المئة. لقد اعتدنا خلال العشرين سنة الأولى من الطائف وجود مرجعية سورية، والآن سوريا لم
تتدخل، وهذا هو المطلوب، وقد أثبتنا أننا قادرون على حل أمورنا بين بعضنا».
وإذ أكد حصول تعاون من الجميع، نوه بموقف بري «الذي ساعد في تسهيل ولادة الحكومة». وتمنى الإسراع في صوغ البيان، وأن يتضمن
الثوابت الوطنية والاستفادة من بيانات الحكومات السابقة. وشدد على ضرورة تكثيف العمل والتضامن واتخاذ القرارات بالتوافق «وإذا تعذر ذلك

Fig. 3. Screenshot from ATEEMA.

FST increments iw_m if it was in the *Maybe Time* and a non-temporal word was detected, and resets it when entering the *Maybe Time* state or when a temporal word is detected. The counter iw_t denotes the number of non-temporal words met within the *Time* state and is updated in a similar fashion to iw_m.

The counter i is the index of the current temporal expression. Once a temporal expression is fully detected, the FST appends it to the `Time` structure and increments i. The actions also include setting the the start sp and end ep positions of the temporal expression. The symbol pos refers to the current position in the text. Figure 3 shows the output of ATEEMA with context sensitive coloring of the extracted temporal entities.

Heuristic optimizations

After a first run of ATEEMA, we made few observations that led us to introduce the following disambiguation heuristics.

Heuristic 1. We ignore the word العام *āl-ʿām* (the year) if it appears in the definite form (i.e. العام *āl-ʿām* instead of عام *ʿām*) and no other temporal words or prepositions appeared next to it. This is mainly because in newspapers, it is common to find the word العام *āl-ʿām* (the year) with a completely non-temporal meaning (i.e. general, common, and public) as in المال العام *āl-māl āl-ʿām* (public money) and الرأي العام *āl-raʾyi āl-ʿā-m* (public opinion). The same rule applies to الثانية *āl-ṯāniyat* (second) which may mean either a second rank (2^{nd}) or the time unit (second).

Heuristic 2. The word الأحد *āl-ʾaḥad* has both 'the one' and 'Sunday' meanings. When used in the indefinite form (i.e. أحد *ʾaḥad*), we interpreted the word as a number. When used in the definite form (i.e. الأحد *āl-ʾaḥad*), we interpreted the word as a day of the week.

Table 2. Temporal entity extraction accuracy results for Safir and Akhbar newspapers.

		Detection			Boundary			Statistics	
		recall	precision	F-score	recall	precision	F-score	words	entities
w/o heuri-stics	Safir 1	0.944	0.648	0.768	0.924	0.922	0.923	5,783	72
	Safir 2	0.926	0.680	0.784	0.829	0.881	0.854	13,457	108
	Akhbar	0.954	0.692	0.802	0.899	0.926	0.912	28,688	503
		0.949	0.685	0.796	0.890	0.918	0.904	47,928	683
with heuri-stics	Safir 1	0.944	0.850	0.895	0.924	0.931	0.928	5,783	72
	Safir 2	0.926	0.813	0.866	0.829	0.881	0.854	13,457	108
	Akhbar	0.950	0.847	0.895	0.908	0.926	0.917	28,688	503
		0.946	0.842	0.891	0.897	0.919	0.908	47,928	683

RESULTS

Similar to the literature, we chose newspapers as our evaluation corpora. Temporal entities are abundant in newspaper texts. We evaluated ATEEMA against text chosen arbitrarily from two issues of the Lebanese Assafir newspaper [1] and one issue of the Lebanese Al-Akhbar newspaper [2]. Table 2 shows the results for Safir 1, the youth section of the Safir 18/5/2011 issue, Safir 2 cultural section of the Safir 20/5/2011 issue, and Akhbar, the politics section of the Akhbar 17/6/2011 issue. As shown in the Statistics columns of Table 2 the three data sets consisted of about 48 thousand words and contained about 680 temporal entities. We evaluated the accuracy of ATEEMA by comparing the output of ATEEMA against a manually tagged version of the text.

We used recall and precision as our evaluation metrics. We report results for the detection and for the boundaries of temporal entities. Detection accuracy refers to the success of ATEEMA in detecting the temporal entities in the text of the newspaper. Boundary accuracy refers to correctly reporting the start and end positions of the extracted temporal entity in text.

Detection recall refers to the fraction of the temporal entities correctly detected against the total number of temporal entities available. Detection precision refers to the fraction of correctly detected expressions against the total number of extracted temporal entities. The same applies to boundary recall and precision measures. Intuitively, the precision measure denotes whether the system generated false positives.

[1] available online at *www.assafir.com*

[2] available online at *www.al-akhbar.com*

The upper rows of Table 2 show that ATEEMA without the heuristics has a high recall (about 95%) but a relatively low precision (about 69%) on the average. This means that ATEEMA detects almost all temporal expressions due to the flexible nature of the FST which supports captures most temporal expression structures.

The expressions ATEEMA missed such as خلال الحرب الأهلية اللبنانية *ẖlāl ālḥrb ālʾahlyt āllbnānyt* (during the Lebanese civil war), and بعد انهيار جدران برلين *bʿd ānhyār ǧdrān brlyn* (after the collapse of the walls of Berlin) needed a deep understanding of the text under consideration that is hard to automate.

ATEEMA also scored 89% and 91.2% for the boundary recall and boundary precision respectively. This means that ATEEMA is capable of detecting the great majority of the temporal entities without significantly over-approximating or under-approximating the boundaries of the entity.

The heuristics improved the precision of ATEEMA with practically no loss in the recall metrics as shown in the lower rows of Table 2. ATEEMA achieved more than 84% precision without extensively enumerating all the temporal expression structures. This is mainly due to the fact that temporal features such as time, numbers, and time prepositions occurring in a neighborhood of text are precise at capturing temporal entities when used with an adequate morphological generalization.

We observed that more than 60% of the detected entities exhibited one or more morphological variations. These entities would not have been detected without the use of the morphological analyzer.

ATEEMA reported false positives in cases similar to نحن بأمس الحاجة إلى *nḥn bʾams ālḥāǧt ʾilā* (we desperately need) since the morphological analyzer reported a temporal morphological feature for أمس as it also means yesterday. Local grammars can not address such failures as well.

RELATED WORK

Much research considered the detection of temporal entities in Latin text especially English [4,5,6,7,8,9,10,11]. Only commercial tools exist with temporal entity support for the Arabic Language [1,3,2].

Gross [21] studied collocations and lexically *frozen phrases* that can be modeled using formal grammar rules and proposed methods to represent the rules with finite state automata efficiently. A frozen phrase is "a phrase in which certain parts cannot be altered. These parts are subject to restricted syntactical variations without affecting the original meaning or function of the expression" [22]. Later Matthieu Constant uses local grammars for text parsing [19], and presented a finite state automaton that parses date expressions and that is able to capture expressions such as "five in the afternoon", "5 p.m.", and "half past one".

Llidó et al. [5] presented techniques to extract and normalize temporal entities using local grammars. The extraction phase uses a shallow semantic-syntactic parser. The normalization phase encodes the semantic meaning of the extracted phrases in terms of a hierarchical formal time model that supports temporal points, intervals, and relative temporal entities.

Koen argues that local shallow parsing is better suited for temporal entity extraction and that full grammar parsing can improve the results if applied afterwards to understand the context of located temporal entities [4]. He reported about 90% recall and precision on extracting date, time, interval and velocity entities. He reported 60% recall on extracting the less useful and less frequently occurring age entities. His work fills missing information in partial temporal entities with a reference extracted temporal entity such that the publication or transmission date of an article. It produces false results when a paragraph discusses events that took place at another date.

The Fact Extractor Workbench [23] uses regular expressions to model local grammars along with optimized capabilities such as caching. The work in [8] uses the IDE to extract temporal and other entities. It caches detected entities, uses the cached entities to complete partial entities, and presents better results for relative dates than that of [4]. However, when the detection of reference temporal entities is non-trivial and needs an understanding of the semantics of the document both approaches fail.

ATEEMA is similar to the work of Gross [21] and Constant [19] who encode their local grammars in efficient state machines. The ATEEMA transducer differs in that it is not grammar based, it does not use frozen phrases, and in that ATEEMA can capture temporal entities with varying structures without formally defining each structure.

ATEEMA differs from the rest of the local grammar based approaches in that it uses a manually optimized FST that takes as input a sequence of morphological features. Local grammars expressed in regular expressions have a less expressive power, are less tolerant to morphological variations, and are automatically translated into non-deterministic finite state machines that may be complex and of large size.

An information extraction core system SMES [10] addresses temporal entity extraction from German texts. SMES consists of a tokenizer that identifies fragment patterns, a lexical morphological analyzer that supports compound expressions, and a shallow parser based on finite-state automata that parses the text to extract temporal entities [12]. It reports a recall of 77% and precision of 88% when evaluated against a corpus of monthly reports about the ‘German IFOR mission in former Yugoslavia’ [10].

COSMA [12] builds on SMES to provide a system for appointment scheduling via Emails. COSMA resolves partial dates and relative temporal expressions by relating the underspecified expressions to their context. The context of an expression includes text of the email, previous email messages in the same conversation, and the temporal meta-data of the conversation. COSMA considers the contextual hierarchy in order until it resolves the expression. If the process fails, COSMA asks for clarification.

ATEEMA is similar to SMES [10] and COSMA [12]. It differs in that it uses an FST to detect a neighborhood for temporal entities. It does not use frozen words and it does not enumerate several regular expressions that extensively capture temporal entities since these are hard to extensively cover in Arabic because of the rich morphology.

Time Calculus for Natural Language (TCNL) [24] extracts and normalizes English temporal entities in scheduling-related emails. The expression ‘$\{|1_{mon}|@\{>= \_\}\}$’ expresses the phrase ‘the coming Monday’. The symbol ‘$\_$’ denotes the temporal anchor to which this relative expression relates; e.g. the timestamp of the email. The Temporal Expression Anchorer (TEA) subsystem determines the anchor and disambiguates the expression using the context, e.g. the tense of the nearest verb. Its favors the most recent event as the anchor of subsequent expressions. The system reported an accuracy of about 80% in normalizing correctly recognized temporal expressions.

Temporal entity detection for Arabic. The Rosette Entity Extractor by Basis Technology supports extracting entities including dates for 13 languages including Arabic [1]. Rosette is based on *aided statistical machine learning* where a computational linguist provides Rosette with information about contextual features to define an entity and a tagged learning set. Rosette then builds a statistical model for extracting the entity. This methodology is language independent and is easily extendable to a large number of languages and entity types including Arabic temporal concepts.

BBN IdentiFinder Text Suite is a named entity extraction tool that supports extraction of Arabic date and time expressions [2]. IdentiFinder seems to be based on statistical learning methods. Currently it supports English, Arabic, and Chinese. However, the methods used in this tool are language-independent just as Rosette.

Arabic Named Entity Extractor (ANEE) from COLTEC supports time and date [3] entity extraction. No information is disclosed about the underlying techniques except that it is based on linguistic NLP techniques along with statistical methods and claims to be unrestricted by simple look-up tables or rigid rules.

Up to our knowledge, ATEEMA is the first knowledge based temporal entity extractor for the Arabic language and unlike its commercial counterparts [1,2,3], ATEEMA does not use statistical learning.

Comparing results to related work

On a similar genre of evaluation datasets, namely newspapers, ATEEMA working against an Arabic dataset reported higher recall (95%) than [4] (90%) working against an English dataset. ATEEMA reported less precision (84% compared to 90%) resulting in about the same F-score without the need to extensively enumerate all possible temporal expression structures. Note that Arabic is much harder to tackle than English.

A more relevant comparison is to compare ATEEMA to the German temporal entity extractor where morphological analysis is also key [10]. ATEEMA working against an Arabic dataset scored 18% extra recall compared to [10] (77%) working against a German dataset and just 4% less precision (84.2% compared to 88%) difference which amounts to a 7% higher F-score for ATEEMA.

CONCLUSION

We presented ATEEMA, the first open source tool for Arabic temporal entity extraction. ATEEMA extracts temporal morphological features from Arabic text, classifies the features into temporal categories, and passes the features to a finite state transducer (FST). The FST detects temporal entities and uses tolerance parameters to detect the boundaries of the temporal entities. ATEEMA achieved very good results compared to counterpart tools over a set of newspaper text selected arbitrarily.

In the future we will explore extending ATEEMA to normalize the temporal entities. We will also explore applying the same technique to other morphologically rich languages and to Latin languages as well.

REFERENCES

1. Cohen, S.: Entity extraction enables "discovery". Technical report, Basis Technology (2006)
2. Technologies, B.: BBN IdentiFinder Text Suite [Online; accessed 22-April-2010].
3. COLTEC: Anee: Arabic named entity extraction. Technical report, Computer & Language Technology (2007)
4. Koen, D.B., Bender, W.: Time frames: temporal augmentation of the news. IBM Systems Journal **39** (July 2000) 597–616
5. Llidó, D., Berlanga, R., Aramburu, M.J.: Extracting temporal references to assign document event-time periods. In: Proceedings of the 12th International Conference on Database and Expert Systems Applications, Springer Verlag (2001)
6. Setzer, A.: Temporal Information in Newswire Articles: An Annotation Scheme and Corpus Study. PhD thesis, University of Sheffield (2001)
7. Setzer, A., Gaizauskas, R.: On the Importance of Annotating Temporal Event-Event Relations in Text. In: Proceedings of 3rd International Conference on Language Resources and Evaluation (LREC2002) Workshop on Annotation Standards for Temporal Information in Natural Language. (2002)
8. Chase, G., Das, J., Davis, S.: Towards developing effective fact extractors. Technical Report DSTO-TR-1729, Australian Goverment, Department of Defence (June 2005)
9. Han, B., Gates, D., Levin, L.: Understanding temporal expressions in emails. In: Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics. HLT-NAACL '06, Stroudsburg, PA, USA, Association for Computational Linguistics (2006) 136–143
10. Neumann, G., Backofen, R., Baur, J., Becker, M., Braun, C.: An information extraction core system for real world german text processing. In: Proceedings of 5th conference on Applied natural language processing. ANLC '97, Stroudsburg, PA, USA, Association for Computational Linguistics (1997) 209–216
11. Li, H., Hu, Y., Gao, G., Shnitko, Y., Meyerzon, D., Mowatt, David: Techniques for extracting authorship dates of documents (December 2009)
12. Busemann, S., Declerck, T., Diagne, A.K., Dini, L., Klein, J., Schmeier, S.: Natural language dialogue service for appointment scheduling agents. In: Proceedings of 5th Conference on Applied Natural Language Processing. (1997) 25–32
13. Berlanga, R., Aramburu, M., Barber, F.: Discovering temporal relationships in databases of newspapers. In: Tasks and Methods in Applied Artificial Intelligence. Volume 1416. Springer Berlin / Heidelberg (1998) 36–45
14. Swan, R., Allan, J.: Extracting significant time varying features from text. In: Proceedings of the 8th International conference on Information and knowledge management. CIKM '99, New York, NY, USA, ACM (1999) 38–45

15. Allan, J., Papka, R., Lavrenko, V.: On-line new event detection and tracking. In: ACM SIGIR conference on Research and development in information retrieval. SIGIR '98, ACM (1998) 37–45
16. Traboulsi, H.: Arabic named entity extraction: A local grammar-based approach. In: Computer Science and Information Technology. (October 2009) 139 –143
17. Benajiba, Y., Zitouni, I., Diab, M., Rosso, P.: Arabic named entity recognition: using features extracted from noisy data. In: Proceedings of the ACL 2010 Conference Short Papers. ACLShort '10, Stroudsburg, PA, USA, Association for Computational Linguistics (2010) 281–285
18. Mohri, M.: Local grammar algorithms. In: Inquiries into Words, Constraints, and Contexts. CSLI Publications, Stanford University (2005) 84–93
19. Constant, M.: Grammaires locales pour l'analyse automatique de textes: Méthodes de construction et outils de gestion. PhD thesis, Université de Marne-la-Vallée (september 2003) (252 pp.).
20. Beesley, K.R.: Finite-state morphological analysis and generation of Arabic at xerox research: Status and plans. In: Workshop Proceedings on Arabic Language Processing: Status and Prospects, Toulouse, France (2001) 1–8
21. Gross, M.: Local grammars and their representation by finite automata. In Hoey, M., ed.: Data, Description, Discourse. Papers on the English Language in honour of John McH Sinclair. Harper-Collins (1993) 26–38
22. Chanier, T., Colmerauer, C., Fouqueré, C., Abeillé, A., Picard, F., Zock, M.: Modelling lexical phrases acquisition in l2. Second Language Acquisition Research: The state of the art (1992)
23. Chase, G., Das, J., Davis, S.: Fact extractor system processing engine
24. Han, B., Gates, D., Levin, L.: Understanding temporal expressions in emails. In: Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics. HLT-NAACL '06, Stroudsburg, PA, USA, Association for Computational Linguistics (2006) 136–143

FADI ZARAKET
AMERICAN UNIVERSITY OF BEIRUT,
LEBANON
E-MAIL: <FZ11@AUB.EDU.LB>

JAD MAKHLOUTA
AMERICAN UNIVERSITY OF BEIRUT,
LEBANON
E-MAIL: <JEM04@AUB.EDU.LB>

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 137–150
RECEIVED 28/10/11 ACCEPTED 09/12/11 FINAL 15/06/12

Subsymbolic Semantic Named Entity Recognition

RONALD WINNEMÖLLER

University of Hamburg, Germany

ABSTRACT

We present a novel application of our subsymbolic semantic TSR approach to named entity recognition and classification (NERC) in order to demonstrate the generic utility of the TSR approach beyond word sense disambiguation and language identification. Experimental results support our hypothesis that TSR techniques can successfully recognize named entities of different types in several languages. These experiments were based on a common framework infrastructure using different sets of features on two German and two English corpora.

1 INTRODUCTION

Named Entity Recognition and Classification (NERC) is an important topic: not only as subtask for natural language processing in terms of scientific research, but also for production environment applications such as web or intranet search, text mining, or content management software. Specifically, we found by analyzing some of our university's web servers log files that approx. 60% of all search queries are related to named entities of a particular kind (mainly personal or organizational data).

For several years, NERC tasks achieve good performance, especially for resource-rich languages. Still there is room for improvement regarding NERC for low-resource languages in named entity disambiguation, named entity fine-grained annotation and domain adaption. In the past, we have shown that the Text Sense Representations (TSR) approach described in Section 2 is well-suited for fine-grained word sense disambiguation. In this paper, we demonstrate that basic NERC also is a sensible topic for applying the TSR methodology thus paving the road for

TSR-based fine grained named entity disambiguation and classification (which, however, is beyond the scope of this paper).

Most current methods in the NERC field follow the statistical analysis paradigm: usually, a handful of selected lexico-syntactic features (e.g. bigram statistics, etc.) are fed into a machine learning algorithm trained on pre-annotated data. Sometimes, this process is augmented by applying handcrafted recognition rules.

In this paper, we will in principle also follow this path but instead of using lexico-syntactic features or applying explicit semantic databases such as wordnet, etc., we will introduce our own methodology of TSR-based subsymbolic semantics and pragmatics to the NERC field (cf. [1], [2]). From David Nadeaus great survey on NERC research and issues (cf. [3]) as well as our own literature work we derive that this indeed is a novel approach to NERC.

In the remainder of this paper, we will briefly describe the TSR approach and its application to NERC. We will then put our work in scientific context by relating to other researchers work in the field. Afterwards, we present some experimental results and conclude with a discussion about our findings.

2 About Text Sense Representations

In traditional Computational Linguistics, text meaning is usually encoded in a specific logical form, thus enabling semantic inferencing (cf. Allen [4]). Pragmatic aspects are often encoded as "heuristics" within a particular algorithm or even not handled at all. An alternative view, often attributed to the field of Text Engineering, uses implicit text meaning representation based on vector space models and accessible by specific algorithms, such as LSA (cf. Landauer *et al* [5]) and others. Other approaches associate lexical items to entities within a dictionary or an ontology in order to create a meaning oriented text knowledge representation, e.g. in the Microkosmos Machine Translation System (cf. Mahesh and Nirenburg [6]).

Our view of Text Meaning is more oriented toward pragmatics and general world knowledge and thus somewhat related to the field of prototype semantics (cf. Baerenfaenger [7], Meinhardt [8], Overberg[9] and others). For understanding "meaning", we rely on Wittgenstein's intuitive *family resemblance* notion of the *use* of language, particularly the usage of a word or text fragment in its context (cf. Wittgenstein's "Investigations" [10]). According to this theory, a human is - for example - able to recognize particular activities as “playing games” even though no single sharp common feature exists that is shared by every possible “game” (e.g. not every game is about winning; some, but not all games require teams, etc.). For a better understanding a visual interpretation of the semantic space of “game” is provided in figure 1.

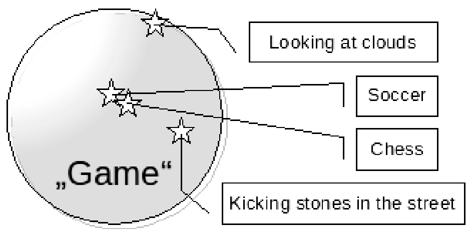


Fig. 1. Visual representation of a prototype for “game”

“Semantic spaces” can be regarded as prototypes in the sense described by Baerenfaenger [7] with some “typical”, i.e. central concepts located around a nucleus and others located at the periphery. These semantic spaces may also interfere with each other so that one concept might be close to the nucleus of one space and peripheral to other spaces.

Thus our notion of text meaning covers the ability of language to assign many pragmatic aspects of meaning to particular words or text fragments, some of which might be obvious, some might be rather unusual.

How words are used in language is analyzed by using the a.m. hierarchical system of *categories* so that a unique set of categories is associated with each particular word or text fragment. It is important to note that we are not assigning predefined “sense definitions” such as WordNet synsets (cf. Miller *et al* [11]) to particular word uses but rather associate data structures to word instances that contain hierarchical views of many possible uses of those words.

This hierarchical view is the basis of what we call a *Text Sense Representation* (TSR) and the foundation of our implementation of a text meaning representation (cf. [1]).

TSRs provide a methodology to represent semantic spaces in a uniform and fairly generally applicable way while being constructed in a fully automatic fashion prior to their use. The basic underlying data structures are tree hierarchies of labeled and weighted nodes, constructed from a web directory such as the Open Directory Project (ODP, cf. [12]).

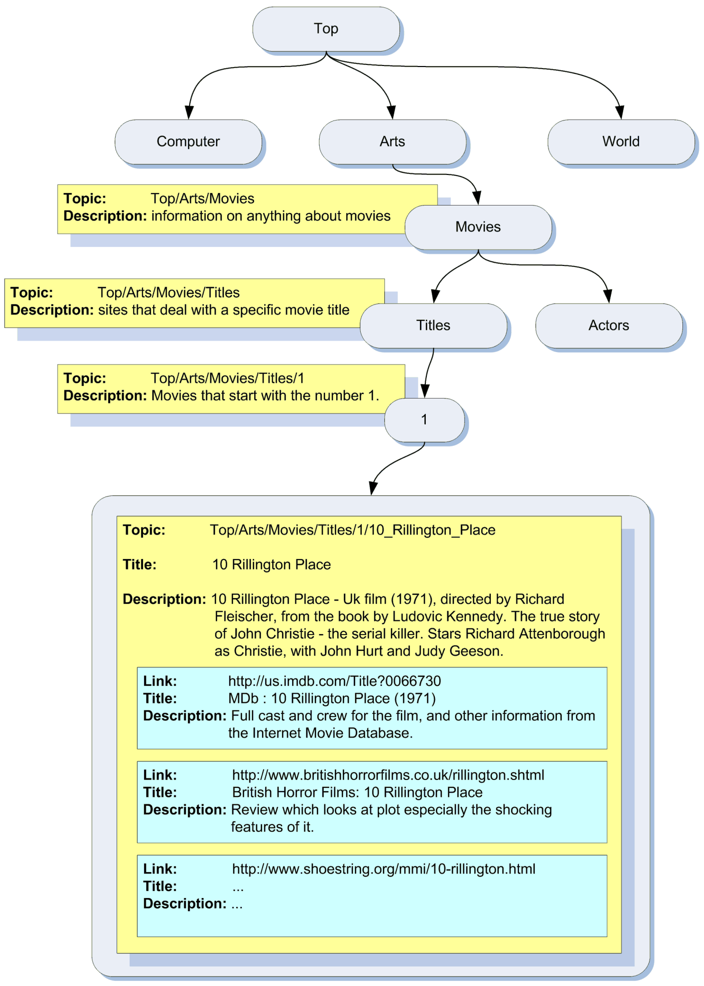
Even though TSRs were described in greater detail by Winnemöller in [1] and [13], we will briefly and informally introduce the fundamentals of TSR tree construction and some operations on TSRs:

1. **Web directory data acquisition** Before building TSR Trees it is necessary to retrieve the underlying web directory data. In our case, we used the ODP RDF-like data dumps, but in principle other data sources like Yahoo [14] or Internet newsgroups data can be used as well. Our implementation uses the ODP directory data in order to create a TSR tree structure for each term that is contained in the ODP data (a term being a sequence of alphanumeric characters). The TSRs are constructed from the nodes of the web directory—a schematic excerpt of which is shown in Figure 2 on the next page.
2. **Sense node construction** Every text node of the web directory is analyzed into distinctive terms (usually stemmed words). Each term is then associated with the full entry path. For example, if a term *account* is found in the directory node */top/business/finance/banking*, then a respective TSR tree node *[account – /top/business/finance/banking]* is created. Each node can be interpreted as "single sense" associated with the extracted term. Each node within a particular tree is therefore associated to a specific category of the ODP structure.
3. **TSR Tree construction** from sense nodes All nodes associated with a term are merged into a single TSR tree structure: every path is weighted by a $TFxIDF$-like formula calculating the occurrences of the TSR term against the overall number of terms within that ODP category.

A very simplistic example of a TSR is presented in Figure 3[1]. One might notice that the leaf nodes do not necessarily add up to 1 - that is because there is also some knowledge encoded in the branch nodes.

Furthermore, we have defined a number of basic TSR operations: a TSR relatedness measure can be used in order to compare two TSRs, a so-called *OR* operation will combine several TSRs into one by creating a TSR that consists of the union set of all input nodes (using the respective maximum weight) while a *AND* operation will create a merged TSR that consists of the intersection set of all input nodes (using the respective minimum weight).

[1] In a subsequent step, the ODP labels are exchanged for a node numbering for technical reasons.



Topic: Top/Arts/Movies
Description: information on anything about movies

Topic: Top/Arts/Movies/Titles
Description: sites that deal with a specific movie title

Topic: Top/Arts/Movies/Titles/1
Description: Movies that start with the number 1.

Topic: Top/Arts/Movies/Titles/1/10_Rillington_Place

Title: 10 Rillington Place

Description: 10 Rillington Place - Uk film (1971), directed by Richard Fleischer, from the book by Ludovic Kennedy. The true story of John Christie - the serial killer. Stars Richard Attenborough as Christie, with John Hurt and Judy Geeson.

Link: http://us.imdb.com/Title?0066730
Title: MDb : 10 Rillington Place (1971)
Description: Full cast and crew for the film, and other information from the Internet Movie Database.

Link: http://www.britishhorrorfilms.co.uk/rillington.shtml
Title: British Horror Films: 10 Rillington Place
Description: Review which looks at plot especially the shocking features of it.

Link: http://www.shoestring.org/mmi/10-rillington.html
Title: ...
Description: ...

Fig. 2. ODP excerpt (schematic)

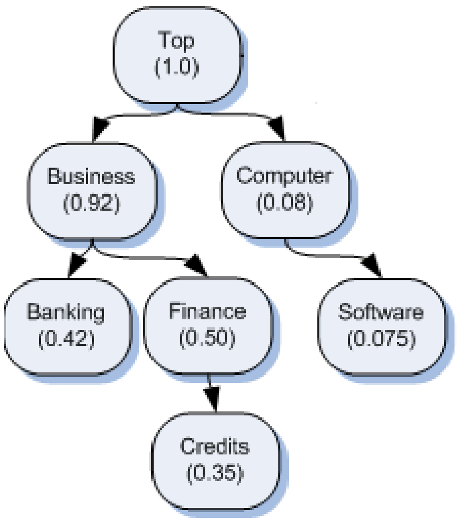


Fig. 3. TSR for the word "account"

In order to be able to compare TSRs with each other, we use a cosine function which consists of applying the well-known vector based cosine function onto the (vectorized) TSR tree node values.

Other operations include the ability of feature selection, as described by the author in [13] and tree minimization.

In summary, TSRs represent pragmatic aspects of text meaning via a combination of categorical text data from existing web directories and several semantic operations.

3 TSR BASED NERC

In the past, TSR vectors were used for unsupervised word disambiguation, language identification, categorizing emails and enriching the University of Hamburg telephone directory by web data (the latter two examples were internal projects and not published). The notion of semantic similarity in these cases was based on computing the cosine between vectors of serialized TSR nodes, i.e. TSR vectors.

Further work on TSRs derived the hypothesis that TSR vectors could be used as input data for machine learning algorithms just like term occurrences, n-grams or other term-related vector data. For this reason, we created a prototype system that reads raw text data, retrieves TSR vectors for the terms within those texts and submits those vectors (plus some TSR metadata such as *width*, *depth* and *size*) as feature vectors to some machine learning system.

Because the "meaning" of the textual terms is not represented by the term vectors themselves but rather by a complex data structure, i.e. a possibly large number of anonymous numerical elements which all together form the semantics of a particular term, we regard this approach "subsymbolic".

For this paper, we applied this methodology onto the problem of supervised NERC, being one important problem that can – to a certain extend – successfully be addressed by exchanging conventional term vector data input with TSR based vector data when training and testing a machine learning system.

4 RELATED WORK

Much work in the NERC field is based on lexico-syntactic features and handcrafted syntactic rules but some approaches also address semantics-based NERC. In this section, we present a small subset of those approaches that we find representative in respect to this paper.

A good example for augmenting the conventional methodology by "semantic rules" was evaluated by Maynard *et al* – they used several lexico-syntactic features plus manually designed "semantic" rules in order to detect named entities (cf. [15]), reporting an F-score of about 60–65% for their experiments.

Even earlier, Cucchiarelli and Velardi report on a multi-stage unsupervised approach which consists of using several third party NERC taggers for creating a robust initial training corpus, then creating several syntactic context features off this corpus and subsequently apply a kind of wordnet-based contextualization (cf. [16]). An evaluation procedure results in about 88% precision and 84% recall on an italian corpus and about 94% precision and 90% recall on an english corpus derived from the Wall Street Journal. Here, an understanding of attribute-based semantics enter the process during the wordnet-related stage.

In addition to this kind of conceptualization technique, Boufaden *et al* employ a semantic NERC algorithm based on a handrafted ontology and similarity scores, derived by using the Lesk algorithm on the ontology and the wordsmyth thesaurus (cf. [17]). They report an F-score of about 86% on transcribed conversations from the domain of maritime search an rescue logs, despite the rather unusual genre (It might be argued, that the underlying knowledge domain is closed and quite restricted, though).

Using large volumes of web-based data, Gentile *et al* present a conceptual graph matching algorithm based on Wikipedia concepts (cf. [18]). Semantical relatedness measures are obtained by applying a random walk algorithm between concept nodes. Accurracy is reported to be about 94% which is comparable to state-of-the-art results. In this case, the system also was avaluated on a closed domain and genre corpus (in the english part, texts from the Wall Street Journal were used). Interestingly, the authors also employed a corpus bootstrapping method vaguely alike the one we used for this paper.

A similar approach was used by Richman and Schone, but they also researched the effect of using Wikipedia NERC on several different languages (cf. [19]). Because Wikipedia and the ODP share the property of addressing many languages, we recognize this work as supporting our notion of multi-language orientation.

5 Experiments

5.1 *Systems Setup*

For the experiments conducted for this paper, we used the OpenNLP framework as common basis for all experiments : *"OpenNLP is a machine learning based toolkit for the processing of natural language text. It supports the most common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and coreference resolution.* "[2] OpenNLP was used successfully by other researchers in the NERC field, e.g. recently by Abacha and Zweigenbaum (cf. [20]) because it has the advantages of being relatively simple, modular, open source and easy to extend.

In principle, we built three system setups using different types of feature sets:

1. The Baseline system setup – the "full feature" set[3]:
   (a) Window features of tokens : previous and next two tokens in context. According to the OpenNLP documentation, the current token is included unchanged while previous p- and next n-tokens are prefixed with p-distance and n-distance respectively

[2] Taken literally from the website `http://incubator.apache.org/opennlp/documentation/manual/opennlp.html`

[3] These are the features originally deployed with the OpenNLP NERC module

   (b) Window features of token classes. This feature behaves just like the token window feature except that it does not include the tokens themselves but the token "classes" (token classes are: lowercase-word, uppercase-word, alphanumeric-word, etc.)
   (c) Bigram feature: bigram tokens
   (d) Sentence feature: sentence begin and end tokens
   (e) Previous mapping feature: features using the outcome of previously occuring words[4]
2. The Baseline system setup – the so-called "realistic" feature set: this setup excludes all features from the above that make use of previous outcomes and in-sentence positioning – TSRs are functionally oriented towards subsymbolic text fragments rather than to words or sentences. It is more realistic than the full feature set setup because the TSR system setup cannot (yet) use such features in a similar way. In order to achieve comparable results, these features ((d) and (e)) have to be abandoned therefore.
3. The TSR system setup: this setup includes TSR related Features only: TSR width, size, depth plus the key indices and values of the TSR vector itself (the TSR vector was pruned by the TOP-algorithm leaving only the top-rated 100 elements, cf. [13]).
4. A combined Baseline+TSR system setup. This setup can not be used to evaluate the baseline versus the TSR based approach but it superficially shows the effect of combining both paradigms.

It is important to notice that only the setups (2) and (3) are fundamental for the cause of this paper (the others are interesting, but they are not suited to prove our point for the reasons given above).

Because we want to demonstrate the effectiveness of the TSR approach augmenting a given system rather than determining the "best" NERC system, we did not evaluate the different setups described here against other approaches.

5.2 *Corpus preparation*

Four corpora were prepared from randomly chosen wikipedia articles for experimenting:

1. *de-1K corpus* : german language; about 1 K sentences; 22 K words; 182 K bytes

[4] The OpenNLP code contains one more feature but this seems equivalent to this one

2. *en-1K corpus* : english language; about 1 K sentences; 30 K words; 214 K bytes
3. *de-10K corpus* : german language; about 10 K sentences; 222 K words; 1.738 K bytes
4. *en-10K corpus* : english language; about 10 K sentences; 294 K words; 2.073 K bytes

The corpora were annotated following roughly the approach described by Cucchiaralli and Velardi (cf. [16]) using a third party NERC tagger – in our case the balie baseline information extraction system (cf. [21]) – and afterwards manually corrected for obvious errors (e.g. one-and-two-character terms as well as non-alphanumeric terms were de-annotated when appropriate). Still, the corpora can be regarded as of relatively low-quality, but on the other hand there was little choice because we simply had no access to publicly licensed cost-free NERC-tagged english and german corpora.

5.3 *Results*

The results of the above explained experiments in terms of precision, recall and combined F-score are presented in tables 1, 2 on the next page and 3 on the facing page (respective best scores are set in boldface).

Table 1. Experimental Results: **Precision**

	german	**english**
1K Baseline System – full feature set	*0.89*	*0.86*
1K Baseline System – “realistic” feature set	0.74	0.75
1K TSR System	**0.88**	**0.91**
10K Baseline System – “realistic” feature set	0.85	0.85
10K TSR System	**0.92**	**0.91**
1K TSR + Baseline System	**0.96**	**0.94**

The results show that the TSR system performs better than the baseline system in terms of precision (here, the best score is actually achieved by the combination of TSR and conventional features) and recall on the german corpus (on the english corpus, the 10 K baseline is better, but only by a small percentage) and also by the combined F-score (TSR performs better than baseline in all experiments).

Table 2. Experimental Results: **Recall**

	german	**english**
1K Baseline System – full feature set	*0.69*	*0.75*
1K Baseline System – “realistic” feature set	0.44	0.58
1K TSR System	**0.51**	**0.60**
10K Baseline System – “realistic” feature set	0.58	**0.69**
10K TSR System	**0.66**	0.67
1K TSR + Baseline System	0.44	0.54

Table 3. Experimental Results: **F-score**

	german	**english**
1K Baseline System – full feature set	*0.78*	*0.80*
1K Baseline System – “realistic” feature set	0.55	0.65
1K TSR System	**0.65**	**0.72**
10K Baseline System – “realistic” feature set	0.69	0.76
10K TSR System	**0.77**	**0.77**
1K TSR + Baseline System	0.61	0.68

It is also worth mentioning that the baseline system using the full feature set would perform best in terms of recall and f-score but not for precision – but as we argued previously, this also means taking in account previous outcomes in context which is not (yet) available for TSR processing.

5.4 *Discussion*

Apart from the explicit outcome, there are a number of interesting points to note:

– High Precision in the case of a TSR based system indicates that the NEs that were detected usually we correct – only a few misclassifications were reported. This is a promising result that points future work to increasing recall rather than precision.

– Low Recall in the case of a TSR based system means that many NEs were missed. This is due to the fact that the intrinsic TSR methodology of learning new terms is not used within these experiments. Therefore only terms that occur within the word space of the ODP can be recognized. This situation can supposedly be remedied by adding terms to the TSR index database through online TSR learning (using words-in-context for creating a corpus on-the-fly – these words can be taken from the original experiment corpus but can also be retrieved from the web using a common search engine) or offline TSR learning (using term features from wikipedia for creating words-in-context. These term features can be similar to the ones mentioned by Gentile *et al*, cf. [18]). Unfortunately, activating on- or offline learning of TSRs remains is out-of-scope of this paper and must remain topic of future research.

- The "realistic" baseline experiments tend to show a much better performance on english than on german data. Since the size of each training corpus within the experiment is approximately equal, it can be argued that the original tagging of the training corpus must be better in the english language case. This effect seems to vanish when using the TSR system on the larger 10K corpora which presumably relies on the fact that german is well represented in the TSR database.
- The problem of named entity ambiguity (cf. [22]) is not yet recognized; TSRs are in principle related to the surface form of words rather than to underlying attribute based semantics. Nonetheless, because of their rich internal hierarchical structure, they can be used to distinguish word senses on different levels of granularity and for many different domains.

6 Conclusions

We presented the novel application of TSR based techniques onto the research field of named entity recognition and classification, using a bootstrapped corpus for evaluating the (supervised) TSR approach against the conventional term based method.

In general, the TSR approach seems well suited to tackle NERC issues - either in a standalone fashion or in combination with conventional methods and features thus once more demonstrating the quite generic nature of the TSR methodology and concept.

Our findings include that the TSR setup led to good results in terms of precision but somewhat less desirable results in terms of recall on the account of several terms being not recognizable by the TSR system. Appropriate future work should therefore involve applying the different TSR learning techniques in order to increase recall scores.

We also showed that performance does not seem to degrade between languages (in our case: german and english) when using a large enough corpus of training data.

REFERENCES

1. Winnemöller, R.: Constructing text sense representations. In Hirst, G., Nirenburg, S., eds.: ACL 2004: Second Workshop on Text Meaning and Interpretation, Barcelona, Spain, Association for Computational Linguistics (July 2004) 17–24
2. Winnemöller, R.: Using meaning aspects for word sense disambiguation. In: 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing), Haifa, Israel (February 2008)
3. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Journal of Linguisticae Investigationes **30(1)** (2007) 3–26
4. Allen, J.: Natural Language Understanding. 2 edn. Benjaming/Cummings Publish. Corp, CA (1995)
5. Landauer, T.K., Foltz, P.W., Laham, D.: Introduction to latent semantic analysis. Discourse Processes **25** (1998) 259–284
6. Mahesh, K., Nirenburg, S.: A situated ontology for practical nlp. In: Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada (Aug 1995)
7. Bärenfänger, O.: Merkmals- und prototypensemantik: Einige grundsätzliche überlegungen. Linguistik online **12** (Mar 2002)
8. Meinhardt, H.J.: Invariante, variante und prototypische merkmale der wortbedeutung. Zeitschrift für Germanistik **5**(1) (1984) 60–69
9. Overberg, P.: Merkmalssemantik vs. prototypensemantik—anspruch und leistung zweier grundkonzepte der lexikalischen semantik. Master's thesis, Universität Münster (feb 1999)
10. Wittgenstein, L.: Philosophische Untersuchungen. In: Werkausgabe Bd. I. Suhrkamp Verlag, Frankfurt am Main (1984)
11. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database. The MIT Press (1998)
12. Netscape Communications Corporation: Open directory project. dmoz.org (2004)
13. Winnemöller, R.: Knowledge based feature engineering using text sense representation trees. In: International Conference RANLP-2005, Borovets, Bulgaria (Sept 2005)
14. Yahoo Inc.: Yahoo. http://www.yahoo.com (01 2004)
15. Maynard, D., Bontcheva, K., Cunningham, H.: Towards a semantic extraction of named entities. In: In Recent Advances in Natural Language Processing. (2003)

16. Cucchiarelli, A., Velardi, P.: Unsupervised named entity recognition using syntactic and semantic contextual evidence. Computational Linguistics **27**(1) (2001) 123–131
17. Boufaden, N., Lapalme, G., Bengio, Y.: Extended semantic tagging for entity extraction. In: Beyond Named Entity Recognition Semantic labeling for NLP tasks Workshop held Jointly with LREC 2004, Lisbon, Portugal (may 2004) 49–54
18. Gentile, A.L., Zhang, Z., Xia, L., Iria, J.: Graph-based semantic relatedness for named entity disambiguation. Proceedings of S3T (2009) 13–20
19. Richman, A., Schone, P.: Mining Wiki Resources for Multilingual Named Entity Recognition. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. (2008) 1–9
20. Abacha, A.B., Zweigenbaum, P.: Medical entity recognition: A comparison of semantic and statistical methods. In: BioNLP 2011 Workshop of the 49th Annual Meeting of the ACL: Human Language Technologies, Portland, Oregon, USA, ACL (June 2011)
21. Nadeau, D.: Balie—baseline information extraction. Technical report, School of Information Technology & Engineering, University of Ottawa, Ottawa, Canada (January 2005)
22. Cucerzan, S.: Large-scale named entity disambiguation based on wikipedia data. In: EMNLP-CoNLL. (2007) 708–716

RONALD WINNEMÖLLER
REGIONAL COMPUTER CENTRE,
UNIVERSITY OF HAMBURG,
GERMANY
E-MAIL: <RONALD.WINNEMOELLER@RRZ.UNI-HAMBURG.DE>

Translation

IJCLA VOL. 3, NO. 1, JAN-JUN 2012, PP. 153–162
RECEIVED 25/10/11 ACCEPTED 09/12/11 FINAL 25/06/12

Translog-II: A Program for Recording User Activity Data for Empirical Translation Process Research

MICHAEL CARL

Copenhagen Business School, Denmark

ABSTRACT

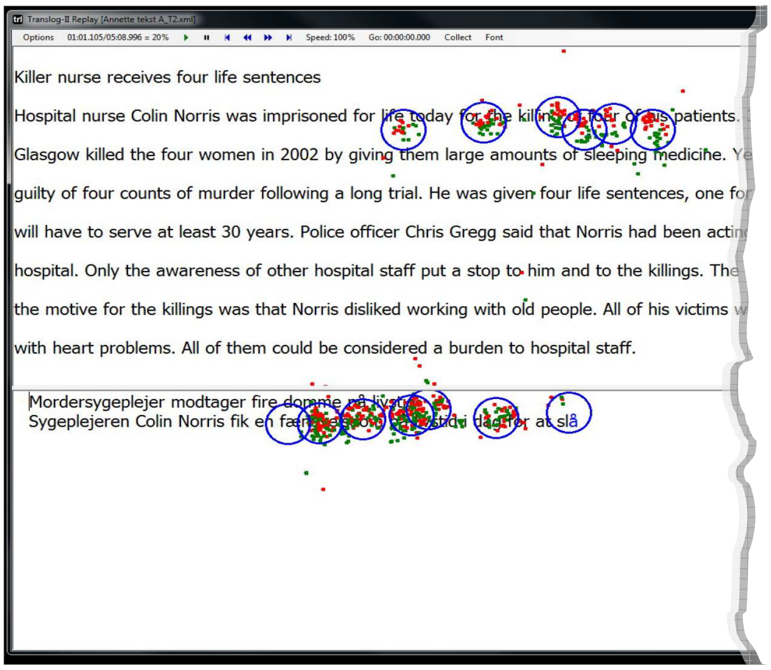
This paper presents a novel implementation of Translog-II. *Translog-II is a Windows-oriented program to record and study reading and writing processes on a computer. In our research, it is an instrument to acquire objective, digital data of human translation processes. As their predecessors, Translog 2000, Translog 2006, also Translog-II consists of two main components:* Translog-II Supervisor *and* Translog-II User. *The two components are interdependent: Translog-II User serves to record experimental translation sessions, while in Translog-II Supervisor project files can be generated, log files replayed and analysed. This paper gives an overview of the tool and its data visualization options.*

KEYWORDS: *Data acquisition software, Multi-modal data acquisition, Human translation process research, User-activity data.*

1 INTRODUCTION

Human translation process research analyses the translation behaviour of translators such as properties of reading and text production rhythms, mental memory and search strategies, types of textual units that translators focus on, etc. It investigates the temporal and contextual structure of those activities and describes inter- and intra personal variation in terms of translation competence and translation performance.

In order to acquire objective data about human translation processes, the program Translog has been designed. Translog can be used to study translation processes, hence the name Translog, but it can be equally used for other kinds of computer-based reading or writing. Since its first conception in 1995, Translog has gone through several re-implementations. Right from its beginnings, Translog had two main components, originally called *Writelog* and *Translog*, (Schou *et al.* 2010) the former component was designed for recording writing



Killer nurse receives four life sentences

Hospital nurse Colin Norris was imprisoned for life today for the killing of four of his patients.
Glasgow killed the four women in 2002 by giving them large amounts of sleeping medicine. Ye
guilty of four counts of murder following a long trial. He was given four life sentences, one fo
will have to serve at least 30 years. Police officer Chris Gregg said that Norris had been actin
hospital. Only the awareness of other hospital staff put a stop to him and to the killings. The
the motive for the killings was that Norris disliked working with old people. All of his victims w
with heart problems. All of them could be considered a burden to hospital staff.

Mordersygeplejer modtager fire domme på livstid
Sygeplejeren Colin Norris fik en fængselsdom på livstid i dag for at slå

Fig. 1. The screenshot shows the source text (top) and the beginning of a translation (bottom) in the target text window. It plots the accumulated gaze movement during the translation of the 8 words: “was imprisoned for life today for the killings” into Danish, which lasted approximately 10 seconds. Red and green dots are gaze-sample points (sampling rate 60Hz.) for the right and left eye respectively, and the blue circles represent fixations. Much more gaze activity is measured on the target window during translation.

processes in time, while the latter component served for playback. These components are now referred to as the *Translog-User* and the *Translog-Supervisor* which are two interdependent programs. A major extension was introduced in the context of the EU project Eye-to-IT in 2006 when a new version *Translog-2006* could connect to eye-tracker through the GWM module (Sparkov, 2008) so as to record both, keyboard and gaze behaviour in time. Translog-2006 was a complete re-implementation in C#, supporting Unicode and XML. However, the communication with the eye tracker through GWM[1] turned out to be too inflexible and so a further development of Translog-II now communicates directly with the eyetracker. This paper describes the purpose and usage of the Translog-II software.

Similar programs such as ScriptLog (www.scriptlog.net/demo.asp), and InpuLog (www.inputlog.net/download.html) are mainly intended for logging and analyzing writing processes, while Translog is specially designed for the acquisition of data for translation process research, and is widely used in the translation process research community. Schou et al (2009) count more than 80 publications making use of Translog, for translation process research of linguistic phenomena, (e.g. the translation of metaphors, cognates, idioms, etc.) as well as translator behaviour and cognitive processes (e.g. translator's awareness, memory constraints, (self)revision etc.), translation expertise, translation under time pressure, and machine translation post-editing. Translog is also used for translator training, teaching and learning purposes.

Translog-II records user activity data (UAD) all the keystrokes and gaze movements (if an eye-tracker is connected). It classifies the keystroke data as 1) insertion, 2) deletion (delete and backspace), 3) navigation (cursor movements), 4) copy/cut-and-paste, 5) return key or 6) mouse operations. Since the keylogger runs in the background, the recording does not interfere with the writing or translation process. Translog-II logs the exact time at which each keystroke operation is made. If connected to an eye-tracker,[2] Translog-II also records 7) gaze-sample points, 8) computes fixations (i.e. clusters of gaze-samples) and 9) mappings of fixations to the closest character on the screen. This latter operation performs a mapping from the spatial location of the gaze on the screen to a character offset in the text. That is, an X/Y

1 These were Borland C++ implementations communicating with Translog through COM interface.

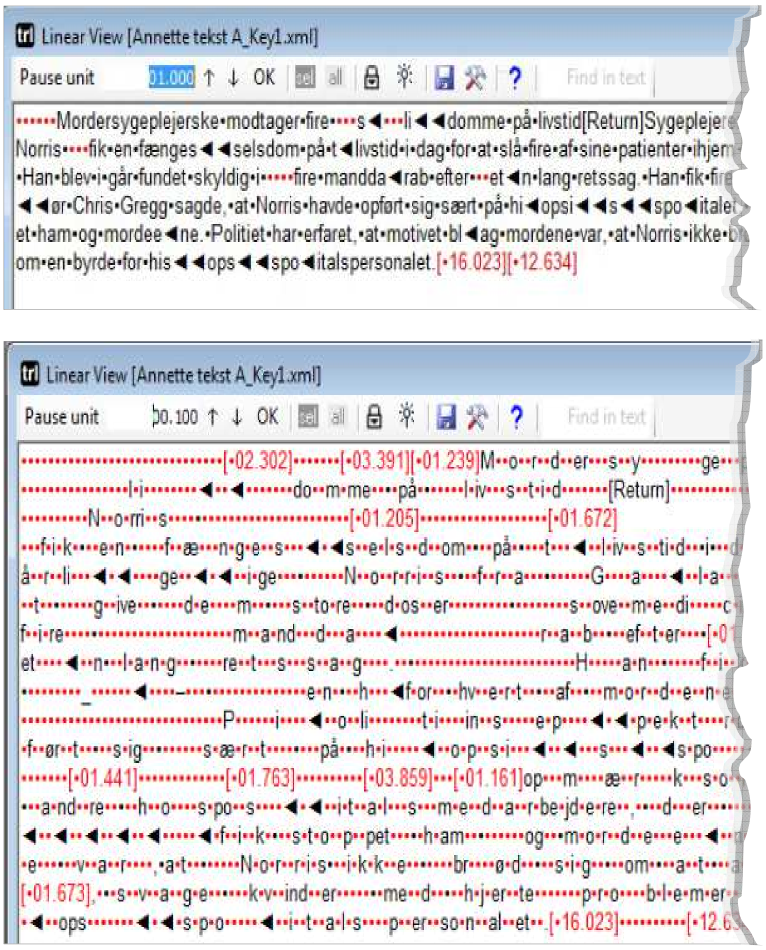
2 Currently connection to Tobii eye tracker is supported, but other interfaces are planned.

coordinate of a fixation center is mapped onto a character position of the text that is being looked at. Since there is some noise in the recordings of gaze-sample points, the representation in the log file is such that fixations and to a certain extent also mappings can be re-computed offline. The gaze and the keystroke information can then be correlated, as they both refer to textual positions. The information is stored in an XML format and can be replayed or analyzed with Translog-II or analyzed in external tool. Carl and Müller (2011) and Carl and Jakobsen (2009) give more information on the XML representation. Here we describe the data acquisition software *Translog-II.*

2 FUNCTIONS OF TRANSLOG-II

Translog-II has three main functions:

1. Create a project file:
   – Determine the size and orientation of a source and a target window on the screen for reading and writing permission respectively.
   – Produce texts for the source and/or the target window, their layout, text font, size, color, line spacing etc.
   – Determine which data are to be logged, keyboard and eye-tracking,
2. Run and record a Translog-II session:
   – Load a project file
   – Calibrate eye-tracker (if connected)
   – Record and log UAD
3. Replay and analyze a recorded log file:
   – Statistics: figures about text production/elimination/navigation events
   – User view: replays the translation session in time (Figure 1)
   – linear view: plots a textual representation of the UAD (Figure 2)
   – Pause plot: shows a 2D representation how the text emerge in time (Figure 3)



```
Linear View [Annette tekst A_Key1.xml]
Pause unit 01.000 ↑ ↓ OK sel all Find in text

••••••Mordersygeplejerske•modtager•fire••••s◄•••li◄◄domme•på•livstid[Return]Sygeplejer
Norris••••fik•en•fænges◄◄selsdom•på•t◄livstid•i•dag•for•at•slå•fire•af•sine•patienter•ihjem
•Han•blev•i•går•fundet•skyldig•i•••••fire•mandda◄rab•efter•••et◄n•lang•retssag.•Han•fik•fi
◄◄ør•Chris•Gregg•sagde,•at•Norris•havde•opført•sig•sært•på•hi◄opsi◄◄s◄◄spo◄italet
et•ham•og•mordee◄ne.•Politiet•har•erfaret,•at•motivet•bl◄ag•mordene•var,•at•Norris•ikke•b
om•en•byrde•for•his◄◄ops◄◄spo◄italspersonalet.[•16.023][•12.634]
```

```
Linear View [Annette tekst A_Key1.xml]
Pause unit 00.100 ↑ ↓ OK sel all Find in text

••••••••••••••••••••••••••••••[•02.302]•••••••[•03.391][•01.239]M••o••r••d••er•••s••y•••••••••ge•••
••••••••••••••••l•i•••••••••◄••◄•••••••do••m•me•••på••••••l•iv•••s••t•i•d•••••••[Return]••••••••••
••••••••••N••o•rri••s••••••••••••••••••••••••••••[•01.205]••••••••••••••••••••[•01.672]
•••f•i•k••••e•n••••••f••æ•••n•g•e••s•••◄••◄s••e•l•s••d••om••••på•••••t•••◄••l•iv••s••ti•d•••i•••d
å••r••li••••◄•◄••••ge•••◄••◄••i•ge••••••••••N••o••r•r•i••s•••••f••r••a••••••••••G••••a•••••◄••l•a••
••t••••••••g••ive•••••••d•e•••••m••••••s••to•re•••••d•os••er••••••••••••••••••••s••ove••m•e••di••••••c
f••i•re••••••••••••••••••••••••••••m••a•nd•••d•••a•••••◄••••••••••••••••••••••••r••a••b•••••ef••t•er•••••[•0
et•••••◄••n•••l•a•n•g••••••••re••t•••s•••s••a••g•••••.••••••••••••••••••••••••••••H•••••a•n••••••••f••i•
•••••••••_••••••◄•••••–••••••••••••••••••••e•n••••h••••◄f•or••••hv••e•r•t•••••af•••••m•o•r••d••e••n•e
••••••••••••••••••••••••••••••••P••••••j•••••◄••o••li•••••••••t•i•••in••s•••••••e•p•••••◄••◄••p•e•k••t••••••r
•f••ør••t•••••s•ig••••••••••s•æ•r••t••••••••••på••••h•i••••••◄••o•p••s•i••••◄•••◄•••s••••◄•••◄s•po•••••
•••••••[•01.441]••••••••••••••[•01.763]••••••••••••[•03.859]•••[•01.161]op•••m•••••æ••r•••••k•••s•o
•••a•nd••re•••••h••o••••s•po••s•••••◄••◄••i•t••a•l•••s•••m•e••d••a••r•be•jd•e•re••,••••d•••er••••••
◄•••◄•••◄•••◄•••◄••••••◄•f••i••k••••s•t•o••p••pet••••••h•am•••••••••og•••m•o•r••d••e•••e•••◄•••
•e••••••v••a••r••••,•a•t••••••••N•o•r••r•i•s••••i•k•k••e•••••••br••••ø•d••••s•i•g••••••om••••a••t•••a
[•01.673],•••s••v••a••g•e••••••k•v••ind••er••••••••me••d•••••h•j•er••te•••••••p•r•o•••b•l•e•m•er•
•◄••ops•••••••◄•◄•s•p•o•••••◄••i••t••a•l•s•••••p••er••so•n••al••et••.[•16.023]•••••••••••[•12.6
```

Fig. 2. Two linear view screen shots of the same text with different temporal resolution. Top: each dot represents 1 second pause. Bottom: a dot represents 0.1 seconds between successive keyboard activities.

The Translog-II Supervisor program implements the functions 1 (create a project file) and 3 (replay a log file), Translog-II User is only used to record a Translog session and to store the UAD in a log file. A Translog-II project file can be configured for a reading experiment, where only the “source window” which contains the text to

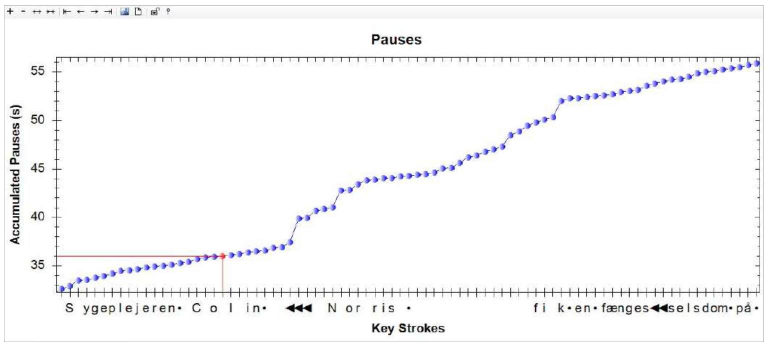


Fig. 3 Screen shot of the pause plot: Blue dots indicate the accumulation of pauses during a translation session (in seconds).

be read will be visible in the recording session, it can be configured for a writing experiment, where only the "target window" is visible, in which a text can be typed, or for a translation experiment, in which both windows are visible. In fact Translog-II also allows for post-editing texts, if a pre-defined text is entered in the target window. Translog-II allows the source and the target windows to be horizontally or vertically oriented and the source or target windows to be left or right, or bottom or top.

As in previous Translog versions, texts can be displayed in smaller portions, e.g. one sentence at a time. Each portion can be displayed for a certain number of pre-defined seconds, or the writer may decide to go on to the next portion of source text when ready to do so.

2.1 *Translog User*

The Translog-II User program is an interface for displaying and typing text and for logging UAD. To start a translation session, a project file must be loaded. According to the settings in a project file, the eye-tracker needs be calibrated, then Translog-II User opens a source and/or a target text window, plots the pre-defined texts in the source window, and waits for the translator to type a translation into the target window. As the size, orientation and rendering of the windows and the font is defined in the project file, it is not possible to re-size the windows in Translog-II User, to change the font. It is possible to use Translog-II User as a post-editing, by providing (machine) translation in the target window and to record text modifications during post-editing.

2.2 *Translog-II replay mode*

The most interesting feature in Translog is the replay mode. Translog-II Supervisor computes some statistical figures on the number of keystrokes, but more interesting are certainly the possibilities to replay the log file. As mentioned above, there are three different ways to visualize the UAD, the user view, the linear view, and the pause plot which are respectively presented in the figures 1, 2 and 3.

The user view (a screen shot is shown in Figure 1) replays the typing process in real-time, and radio buttons can be used to accelerated or decelerated, to pause the replay, rewind or forward it etc. In addition to the keystrokes, Translog-II also plots the gaze-sample points, fixations, and fixated words. In Figure 1, gaze sample points and fixations were collected over a period of approx. 30 seconds illustrating the gaze path and the coordination of reading and writing activities. It is possible to select whether gaze and fixation information should be plotted.

The linear view represents the UAD in a textual (linear) manner. Each key and mouse activity[3] has a representation in the linear view, and pauses are either indicated as asterisks, and/or numeric value indicating the duration between successive activities. The granularity of the pause display can be selected starting from 1ms up to any amount of time. This gives the possibility to get an overview over the coarse temporal structure of a translation session, reducing the temporal information to a minimum (Figure 2, top), or to zoom into a sequence to study pausing behaviour as small as a few hundreds of seconds (Figure 2, bottom).

The third Translog-II replay mode is the pause plot. A pause plot represents essentially the same information as the linear view does, this time in as 2D graph. Keyboard activities are indicated on the horizontal X-axis, while the vertical Y-axis shows the accumulation of time (pauses). Figure 3 shows a segment of a translation session.

It is possible to scroll through the pause plot, to zoom in or out. Translog-II also allows to synchronize all three visualization methods. That is, all three windows (user and linear view as well as pause plot) can be opened at the same time, and by clicking the synchronization item the cursor in all three windows will be positioned at the same time. The option in the user view would then trigger a synchronous replay in the three windows.

[3] It is also possible to visualize gaze-samples and fixations in the linear view, which is omitted here.

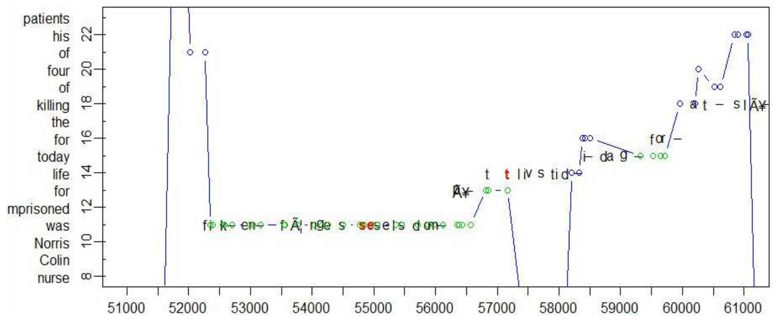
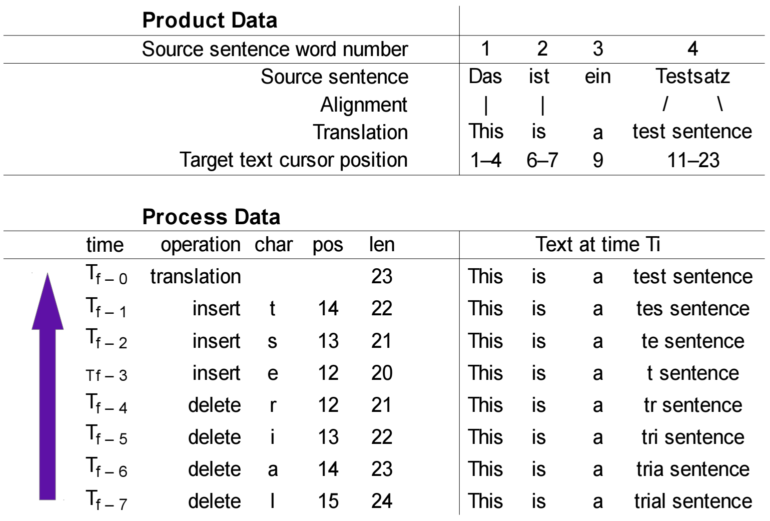


Fig. 4. The graph visualizes the translation progression of the 8 English words: "was imprisoned for life today for the killings" into Danish: "*fik en fængselsdom på livstid i dag for at slå*". This translation segment corresponds to the accumulated gaze movements in Figure 1, which lasted approximately 10 seconds. Blue dots represent fixations on the source text, green dots fixations on the target text, the characters are insertions and the red characters are deletions.

3 TRANSLATION PROGRESSION GRAPHS AND PRODUCT DATA ALIGNMENT

While the visualization options in Translog-II (Figures 1–3) trace how the *target text* emerges in time, we have also developed more powerful visualization possibilities that show how the *translation* evolves. Figure 4 plots the relation between the word positions in the source text (vertical axis) and the translation activity on the horizontal axis in time. The figure presents a time segment of ca. 10 seconds (from secs. 51 to 61) in which the Danish words: "*fik en fængselsdom på livstid i dag for at slå*" are produced. Each keystroke is mapped onto the source text segment to the translation of which it contributes. Thus, line 11 shows all activities that relate to the production of the translation for English "was imprisoned", line 13 for that of "for", line 14 of "life" etc. The graph also shows gaze activities in relation to source segments.

Translation progression graphs require additional alignment knowledge of the source and the target texts, and are therefore not supported inside Translog-II. A set of additional tools are used 1. to align the translation, 2. to compute the keystroke-to-source text mapping and 3. to visualize the graph. Carl and Jakobsen (2009) describe a rule formalism and a general method to map keystrokes on



Product Data

Source sentence word number	1	2	3	4
Source sentence	Das	ist	ein	Testsatz
Alignment	|	|		/ \
Translation	This	is	a	test sentence
Target text cursor position	1–4	6–7	9	11–23

Process Data

time	operation	char	pos	len	Text at time Ti			
T_{f-0}	translation			23	This	is	a	test sentence
T_{f-1}	insert	t	14	22	This	is	a	tes sentence
T_{f-2}	insert	s	13	21	This	is	a	te sentence
T_{f-3}	insert	e	12	20	This	is	a	t sentence
T_{f-4}	delete	r	12	21	This	is	a	tr sentence
T_{f-5}	delete	i	13	22	This	is	a	tri sentence
T_{f-6}	delete	a	14	23	This	is	a	tria sentence
T_{f-7}	delete	l	15	24	This	is	a	trial sentence

Fig. 5. The mapping algorithm traverses the process data against the time line, from the final translation product towards the start of the translation session. The arrow on the left indicates the time flow in the process data. While traversing the process data, all operations are associated with a target text position, and hence a source sentence number.

source text words. They iteratively retrieve all keystrokes which contribute to the creation of a target text segment T*i*. Given alignment information for the target text segment T*i* to a source segment S*j*, the retrieved keystrokes can be mapped via T*i* to source segment S*j*. With an exhaustive fragmentation of the target text into *n* non-overlapping text segments T*1...n*, and a complete alignment of the source and the target texts, every keystroke can be associated with a source segment S*j*. As illustrated in Figure 5, we have re-implemented this algorithm in a more efficient way: From the alignment of the translation product we know which source words are linked to which cursor positions in the target text. The text processing operations can then be mapped on the target text positions and further on source words. Figure 5 illustrates the correction of "trial sentence" into "test sentence" in the target text. These operations are mapped on source word 5 "Testsatz".

Figure 5 shows that the four last letters "rial" of the word "trial" were deleted and then substituted by "est". As all deletion and insertion activities take place between cursor positions 11 and 23, these keystrokes are part of the operations that lead to the translation of

"Testsatz". To compute these mappings, the algorithm looks backwards into the activity data. The last operation in Figure 5 was the insertion of the letter "t" at time T_{f-1}, which took place at cursor position 14. Accordingly, the text length was accordingly one character shorter and the translation of "Testsatz" consisted only of the characters 11 to 22. Starting at T_f, the time of the final translation, the loop ends with the first operation, where all keystrokes are mapped on a source text word.

A free version of Translog-II can be downloaded for academic use upon request to the author of this paper.

ACKNOWLEDGEMENT. The implementation of Translog-II would not have been possible without previous work on the various Translog versions in the development of which were involved Lasse Schou, Arnt Lykke Jakobsen, Morten Lemvigh and Jakob Elming.

References

1. Carl, M., Jakobsen, A.L.: Towards Statistical Modeling of Translators' Activity Data. In International Journal of Speech Technology, Volume 12, Number 4, (2009) 125–138
2. Michael Carl and Henrik Høeg Müller: CRITT NLP Resources for Translation Representation of User Activity Data in Translog-II. In Proceedings of LTC, Poznan (2011)
3. Lykke Jakobsen, A.: Logging target text production with Translog. In Hansen, G. (ed.), Probing the process in translation: methods and results, Copenhagen Studies in Language. Volume 24. Copenhagen: Samfundslitteratur. (1999) 9–20
4. Lasse Schou, Barbara Dragsted & Michael Carl: Ten years of Translog. Copenhagen Studies in Language (37), (2009) 37–51
5. Špakov, O.: GWM – the Gaze-to-Word Mapping Tool. (2007). Available online at: http://www.cs.uta.fi/~oleg/gwm.html (visited on June 21, 2011).

MICHAEL CARL
COPENHAGEN BUSINESS SCHOOL,
DENMARK
E-MAIL: <MC.ISV@CBS.DK>

Author Index

ADDITIONAL REFEREES FOR THIS VOLUME

Dipankar Das
Egoitz Laparra
Ekaterina Ovchinnikova
Enrique Amigó
Eugen Ignat
Fazel Keshtkar
Feiyu Xu
Francisco Jose Ribadas Pena
Frederik Vaassen
Gabriela Ferraro
Gabriela Ramirez De La Rosa
Gerold Schneider
Gorka Labaka
Guenter Neumann
Guillermo Garrido
H M Ishrar Hussain
Håkan Burden
Hendra Setiawan
Hiroya Takamura
Hiroyuki Shindo
Ingo Glöckner
Ionut Cristian Pistol
Irina Chugur
Irina Temnikova
Jae-Woong Choe
Janez Brank
Jirka Hana
Jirka Hana
Jordi Atserias
Julian Brooke
K. V. S. Prasad
Katsuhito Sudoh
Kishiko Ueno
Kostas Stefanidis
Kow Kuroda
Krasimir Angelov
Laritza Hernández
Le An Ha
Liliana Barrio-Alvers
Lorand Dali
Luis Otávio De Colla Furquim
Luz Rello
Maite Oronoz Anchordoqui
Maria Kissa
Mario Karlovcec
Martin Scaiano
Masaaki Nagata
Matthias Reimann
Maud Ehrmann
Maya Carrillo
Michael Piotrowski
Miguel Angel Rios Gaona
Miguel Ballesteros
Mihai Alex Moruz
Milagros Fernández Gavilanes
Milos Jakubicek
Miranda Chong
Mitja Trampus
Monica Macoveiciuc
Najeh Hajlaoui
Natalia Konstantinova
Nathan Michalov
Nattiya Kanhabua
Nenad Tomasev
Niyu Ge
Noushin Rezapour Asheghi
Oana Frunza
Oier Lopez De Lacalle
Olga Kolesnikova
Omar Alonso
Paolo Annesi
Peter Ljunglöf
Pinaki Bhaskar
Prokopis Prokopidis
Rainer Winnenburg
Ramona Enache
Raquel Martínez
Richard Forsyth
Robin Cooper
Rodrigo Agerri

Roser Morante
Ryo Otoguro
Samira Shaikh
Santanu Pal
Shamima Mithun
Sharon Small
Simon Mille
Simone Paolo Ponzetto
Siva Reddy
Somnath Banerjee
Tadej Štajner
Thierry Declerck
Ting Liu
Tom De Smedt
Tommaso Caselli
Tong Wang
Toshiyuki Kanamaru
Tsutomu Hirao
Ulf Hermjakob
Upendra Sapkota
Vanessa Murdock
Victor Darriba
Víctor Peinado
Vít Baisa
Vojtech Kovar
Wilker Aziz
Yulia Ledeneva
Yvonne Skalban
Zuzana Neverilova