SciVerse ScienceDirect

Procedia - Social and Behavioral Sciences 27 (2011) 178 – 187

Procedia Social and Behavioral Sciences

Pacific Association for Computational Linguistics (PACLING 2011)

Named Entity Recognition for Short Text Messages

Tobias Ek[a]*, Camilla Kirkegaard[a], Håkan Jonsson[b], Pierre Nugues[a]

[a]*Lund University, Department of Computer science, Box 118, S-221 00 Lund, Sweden*
[b]*Sony Ericsson, Nya vattentornet, S-221 88 Lund, Sweden*

Abstract

This paper describes a *named entity recognition* (NER) system for short text messages (SMS) running on a mobile platform. Most NER systems deal with text that is structured, formal, well written, with a good grammatical structure, and few spelling errors. SMS text messages lack these qualities and have instead a short-handed and mixed language studded with emoticons, which makes NER a challenge on this kind of material.
We implemented a system that recognizes named entities from SMSes written in Swedish and that runs on an Android cellular telephone. The entities extracted are *locations*, *names*, *dates*, *times*, and *telephone numbers* with the idea that extraction of these entities could be utilized by other applications running on the telephone. We started from a regular expression implementation that we complemented with classifiers using logistic regression. We optimized the recognition so that the incoming text messages could be processed on the telephone with a fast response time. We reached an F-score of 86 for strict matches and 89 for partial matches.

Keywords: Named entity recognition; Short text messages; SMS; Information extraction; Ensemble systems;

1. Introduction

Named entity recognition (NER) from short text messages (SMS) on handsets has until now been constrained by computing power and memory. Current implementations only detect telephone numbers or hyperlinks using regular expressions. Most implementations are optimized for a high recall and usually match any number as a telephone number.

Telephone numbers and hyperlinks are not the only valuable named entities within SMSes. Locations, names, telephone numbers, dates, and times are all potentially important information that the user may

* Corresponding author. Tel. 0046705973300
E-mail address: tobias.ek@telia.com.

doi:10.1016/j.sbspro.2011.10.596

want to use in other applications. For example, the user may want to perform a map search on a location name occurring in a SMS. Recognizing these entities is the first step to allow the user to pass the data on to other applications with minimal user interaction.

This paper describes a system that recognizes named entities from SMSes written in Swedish and that runs on an Android cellular telephone. Entities detected are telephone numbers, dates, times, locations, and person names. We gathered a corpus to evaluate the detection accuracy and we compared it with that of existing published systems. Although not exactly comparable, we report results that we believe are on a par with what could be expected from a mid or high-end computer. One of the essential features of our NER system is that we started from a regular expression implementation that we improved with corpus-trained classifiers using logistic regression.

2. Previous Work

Little work on named entity recognition in constrained environments has been published. Jiang et al. [7] investigated the use of hidden Markov models (HMM) to extract named entities related to events or activities from SMSes in Chinese. The execution was specifically targeted to be for handsets. While they achieved a lower F-score, the authors could reduce significantly memory consumption. However, they used a SMS corpus of 1,000 messages, which is relatively small compared with sizes common in the field. They combined it with a larger corpus using daily newspaper data.

Polifroni et al. [11] used logistic regression to recognize name, location, date, and time entities from spoken or typed messages. They built a corpus from transcribed utterances and English SMSes from real users in a laboratory setting. They reported F-scores for names and locations reaching 88 on an individual word basis. The end goal is to use the system in a mobile setting for automatic speech recognition, but they do not report on computational or memory resources required of their approach.

Hård af Segerstad [6] provides an extensive analysis of the linguistic characteristics of Swedish SMS texts and usage of SMS in Sweden. She found that SMSes contain unconventional and not yet established abbreviations based on Swedish as well as words from other languages, unconventional or spoken-like spelling, unconventional use of punctuation, and use of non-alphabetical graphical means, e.g. emoticons. For NER, this means that SMS is a particularly hard domain.

3. Corpus: Collection and Annotation

We built a corpus using incoming and outgoing messages from 11 participants. Messages were written mainly in Swedish, but sometimes mixed with English and German. This corresponds to a realistic SMS use in an international setting in Sweden. We collected this corpus in parallel with the development of the NER system and we reached a size of about 4,500 text messages consisting of 60,000 tokens.

Such a size seems to be at the lower limits in terms of data quantity needed to have a reliable evaluation [1]. A larger amount of data would of course guarantee a more accurate result, but at the expense of a more costly gathering procedure. Even if the figure of 60,000 tokens seems limited, the gathering task turned out to be surprisingly difficult as many users consider a SMS to be private, if not intimate. Most users were unwilling to share their data unless an option was offered to exclude certain private text messages.

We annotated the corpus with five categories of named entities potentially useful for applications; see Table 1. As markup language, we used the IOB2 format [12] with the tags: B (begin), I (inside), and O (outside). In our corpus, around 90% of the tokens are tagged as outside. Below is an example of a bracketed message:

Nu åker vi in till [$_{LOC}$ stan]. Ska vi ses [$_{TIM}$ kl 17.15] på [$_{LOC}$ max]? Puss
'We are driving into [$_{LOC}$ town] now. Should we meet at [$_{TIM}$ 17.15] at [$_{LOC}$ max]? xxx'

Table 1. Named entity types and examples

Entity	Tag	Examples	Entity	Tag	Examples
Date	DAT	*10-09-22*, *22/09/10*	Name	PER	*Tobias*, *Torsten Andersson*
Time	TIM	*12:34*, *klockan nio*	Location	LOC	*Lund*, *skolan*
Telephone no.	PHO	*073-123456*, *+464612345*			

We annotated all the tokens of the tokenized corpus with the five categories we wanted to extract and their corresponding IOB2 tag. Annotation is a time-consuming and costly process that requires a good deal of human effort. As an initial step, we *bootstrapped* this process by applying a set of manually written regular expressions to detect and label the entities. As a second step, we corrected the entity labels by hand to produce the final corpus.

In addition to being concise, regular expressions are very effective in finding numerical tokens such as dates, times, and telephone numbers that appear in recurring patterns even across disparate formatting styles. However, they are insufficient for entities that differentiate too much from these simple patterns.

During the development, we utilized a development set consisting of 2,000 tokens (214 text messages). This set was throughout used for regression testing, feature selection, system comparison, and general error analysis.

4. The Initial Regex-based System

We started the NER system with a regex-based implementation. Table 2 shows examples of regexes for numerical expressions. We evaluated this system with a script derived from the one used in CoNLL 2003 [13] and we reached a F-score close to 74 on the development set.

Table 2. Sample of regular expressions used to detect numerical entities

Regex	Matches
`den[ ](([1-3][0-9])|([1-9]))`	den 23
`kl(ockan)?[ ](\d{1,2}[:.]\d{2})`	kl 13:37
`(2[0-3])(([:.])[0-5]\d)\{1,2\}`	21.52
`[0-2]\d[0-5][0-9]`	0845
`((00|[+])\d{2}[-\s]?([\s]?\d){6,})`	+46-123456
`\d{2,}-\d+`	08-123456
`[+][+]?\d+[ ]?(\d{1,}|[ ()]){2,}`	++45 (404) 354 54

The system accuracy was acceptable for number-based tokens, mostly dates, times, and telephone numbers, as long as the entities were more or less well formed. Nonetheless, the regular expressions quickly reach their limits when users are using nonstandard formats. Although, it is possible to continue developing a regex set, any improvement comes with increasing complexity and reduced manageability.

Finding letter-based tokens, like names and locations with regexes limits the system to an already known datasets (i.e. lists) with the exception of suffixes for locations. One could gamble and try

expressions that look at word patterns such as “at the” but such guesswork is better handled by a classifier. There is also an ambiguity with names and locations that is not easily solved; locations, such as restaurants or coffee shops, are often named after a person.

5. Named Entity Detection using Classifiers

The architecture of the classifier-based system consists of a pipeline of components; see Figure 1. The training steps are carried out on a desktop computer and the recognition steps on a cellular telephone. The training procedure takes a SMS corpus as input, tokenizes it, and tags each token with its part of speech using tools from the OpenNLP toolkit [10]. Finally, we trained our recognition models using logistic regression from the LIBLINEAR package [4]. The recognition procedure on the cellular telephone uses a similar pipeline except that it utilizes the linear regression models produced in the training phase to mark up the named entities.

5.1. Part-of-Speech Tagging

We used two part-of-speech taggers: Granska [2], a high-performance tagger for Swedish written in C++ for the training phase, and the OpenNLP tagger written in Java to run on the Android platform. We had to use Granska first to annotate our corpus, as OpenNLP has no model for Swedish. We could not use Granska on Android, as it is not written in Java.

We also wanted to control the size of the models due to memory restrictions for the cellular telephone. An application on Android cannot allocate more than 16 MB of memory and with OpenNLP, we can adjust the model size using smaller amounts of training data. The price of a smaller POS tagger is unfortunately that it will be less accurate.

5.2. Design of a Feature Set

We trained the named entity classifier with a set of features that we extracted from the tokens. Finding efficient features can be a never-ending task. We first built a feature superset from sets described by [3] and [5]. We then created our own features. Throughout the development, we carried out regression tests using a forward greedy selection to verify that features had positive impacts on the performance and design an optimal set.

We used a set of about 30 features and Table 3 shows the major ones. As in [8], we extracted these features using a window of three tokens before and after the current token. We used the lexical value and part of speech of the tokens. Most of the features in Table 3 have a self-explanatory name. Some features reflect properties of the current token: digits, letters, prefixes, and suffixes; some consider the part-of-speech tags. The *contain features* use either regular expressions or lookups from lists. Finally, some features model the context by looking at surrounding words.

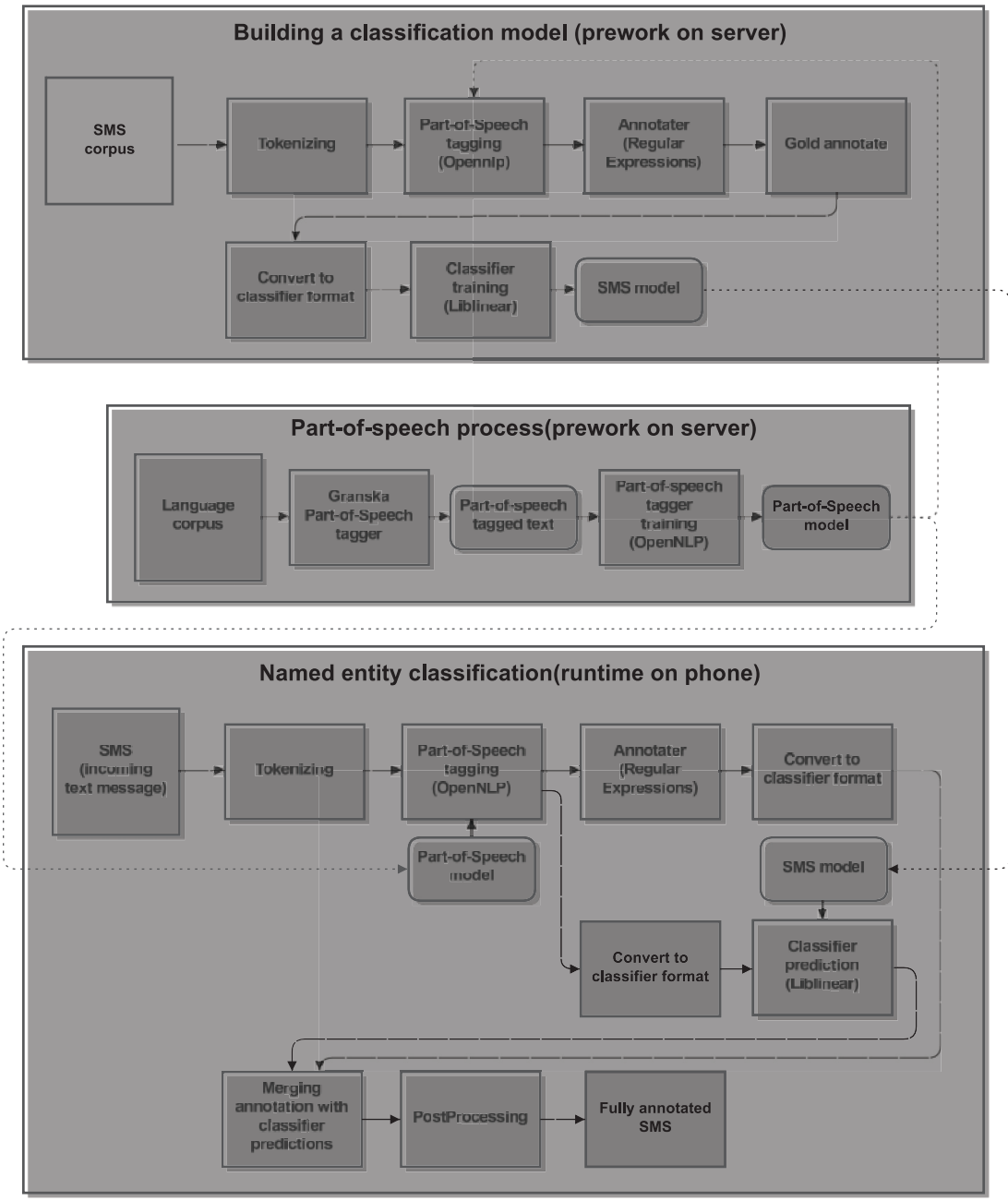

Figure 1 Overview of the major steps and components of the NER system

Table 3. The major features

Type	Regex	Matches	Type	Regex	Matches
Regex	Contains date	*Augusti, Måndag*	**List**	Contains derived word	
	Contains times	*12:00, klockan 11*		Common word	*bil* 'car', *hus* 'house'
	Contains telephone no.	*046-123456*		Contains common name	*Maria*, *Peter*
POS	Part-of-speech tags	noun, verb		Contains derived name	
	Verb	*gå* 'walk', *läsa* 'read'		Contains location	*Stockholm*, *Malmö*
	Specific verb	*träffas* 'meet', *äta* 'eat'		Contains derived location	
	Noun	*tåg* 'train', *bok* 'book'		Frequent unigrams	*södra* 'south'
	End of sentence	!.?		Frequent bigrams	*jag och* 'me and'
	Preposition	*till* 'to', *från* 'from'		Frequent trigrams	*Jag är på* 'I am at'
Other	Contains digit	12:34, 1234		Frequent POS unigrams	preposition, noun
	Contains denominator	, . - ; : /) (			
	Only digits	1234567890			
	All uppercase	SEMC, XML, NLP			
	Initial uppercase	*Orange*			
	Initial lowercase	*orange*			
	Prep + token + end	*till Lund*. 'to Lund.'			
	Only letters	*utansiffror* 'withoutdigits'			
	Length of the word				

5.3. Feature Performance

We analyzed the performance obtained by different categories of features on the development set described in Sect. 3.2. Table 4 shows the recognition results obtained using sets, where all the features belong to one category only. The main conclusions are that:

- Features based on the POS tags produce relatively evenly distributed figures across the named entity categories. However, the recognition performance of the system would be low if it only relied on POS tags.
- Regular expressions have a good performance, especially with numbers, even when being without support from other features. The classifier responds well on the clues given by the regexes and they are easy to combine with other sets of features.
- Lists of names and locations give a boost to locations and names. Lists containing unigrams, bigrams, and trigrams improve further the performance. N-gram lists need a *cut off* value to keep the system robust. Otherwise the system would overreact to hapaxes.
- Other features deal mostly with the different characteristics of a single token. For instance, one feature checks if the token only contains uppercase letters.

6. Lists

Using lists improved the performance for both systems. The classifier responded well on features using information based on the lists. There exists a risk to overgrow the lists and introduce noise, not to mention to slow down performance. We tried to avoid building large lists to mitigate the risk of false positive hits.

Selective smaller lists with relevant content should in most cases be the wiser choice. Mikheev et al. [9] have shown that their content is far more important than their size and a small gazetteer list with well-known entries is far more helpful than a large list listing relatively unknown names, which seldom appear in text.

We gathered lists of people names and locations from sources such as *Statistiska centralbyrån* 'Statistics Sweden' (SCB)[†]. From these lists, we pulled first names, last names, and locations. Table 5 shows our static lists.

In addition to the gazetteer information, we also used lists that we derived automatically from the training data. We extracted a list of frequent words occurring in the training corpus. We also used lists of frequent bigrams and trigrams. We built lists of words and parts of speech that precede a *named entity*. We assigned different cutoff frequencies for each list with the aim to keep the lists as small as possible without infringing on performance.

Table 4. F-score of different feature sets. The *All* column shows the results of the complete feature set. The other columns show the results of feature subsets consisting of only one category

Tag	POS	Regex	List	Other	All
TIM	43	67	43	22	87
PHO	43	64	67	72	87
DAT	31	72	50	0	94
LOC	17	57	48	3	72
PER	22	65	81	0	87
Mean	28	65	56	10	84

Table 5. Gazetteer information obtained from various Swedish sources

Category	Size	Example	Category	Size	Example
First names	200	*Adam*, *Maria*	International cities	150	*New York*, *Tokyo*
Last names	100	*Svensson*, *Lundberg*	Points of interest	15	*systemet*, *torget*
Family relation	14	*faster*, *mamma*	Months	32	*juli*, *dec*
National towns	2,000	*Stockholm*, *Lund*	Days	60	*julafton*, *torsdag*

7. Evaluation

We developed an evaluation program based on the CoNLL 2003 script [13] using the precision, recall, and harmonic mean of them: F-score. As users may accept a partially correct detection, we computed two F-scores:

- **A strict F-score** that uses the entire tag: A tag is counted as correct if both the prefix, *Begin* or *Inside*, and the entity category, TIM, PHO, DAT, etc., are correct.
- **A partial F-score** that sets aside the prefixes, *Begin* and *Inside*, and uses the entity category. It makes no difference between B-LOC and I-LOC, for instance.

The strict F-score reflects then the proportion of tags that are completely correct, while the partial F-score measures recognitions where the user has to adjust the boundaries of the named entities.

[†] http://www.scb.se

7.1. Comparison

We compared the performance of our initial regular expression system with that of the classifier-based one using the development set and our evaluation script. The system based on regular expressions reached a strict F-score of 76.76 and the classifier an F-score of 77.73. Both systems found mostly the same occurrences, but there were differences in their performance. Table 6 shows their strengths broken down by category.

Table 6. Classifier vs. regular expressions. The star (*) shows which system performed best with that particular IOB2 tag

NE	Classifier	Regex	Difference
B-TIM		*	2%
I-TIM		*	36%
B-PHO		*	5%
I-PHO		*	15%
B-DAT		*	9%
I-DAT		*	14%
B-LOC		*	4%
I-LOC	*		30%
B-PER	*		3%
I-PER	*		71%

7.2. Reconciling the Output

We used these differences to build the final ensemble recognition algorithm. It uses a voting procedure, where in case of disagreement the system the most efficient in the category wins. The algorithm is based on trial and error, and the combination that gave the best performance:

- If both systems agree on a tag, this tag is selected.
- If one of the tags is O and the other is a named entity tag, the entity tag is selected.
- If the classifier outputs a tag with the PER category, this tag is selected.
- If both tags are named entities and not of the PER category, the regex annotation is chosen.

8. Evaluation Setup and Results

We used a 10-fold *cross validation* to estimate the performance of the ensemble system: regular expressions combined with the classifier-based system. We divided the data set into a test set (1/10) and training set (9/10), where we assigned every tenth text message to the test set.

We broke down the F-scores per entity category to get a more detailed picture of the strengths and weaknesses of our system. Table 7 shows the confusion matrix per category. The O tag is excluded from most tables and F-score calculations, since it is not a named entity like the other tags. Table 8 shows the results for the strict and partial matches using cross validation. As final results, we obtained F-scores of 86.44 for the strict matches and 88.85 for the partial matches on the cross validation.

9. Comparison with other NER Systems

We compared the performance of our ensemble system with that of other published systems. This area seems to be new for cellular telephones. Most systems we reviewed are larger and run on desktop computers without specific constraints in memory, response time, or processing power. Most systems are also intended to process formal text. Ours had to handle both informal and formal text. In addition, few use Swedish and their corpora were not accessible to us.

Table 9 shows the key figures that compare our system with the works of [7] and [11]. We could not find any NER system for SMS that targets Swedish as language. [11] did not report the size of corpus, other than qualifying it as large. The SMS corpus was created by users in a laboratory setting and is used together with a corpus of transcribed voice notes. [7] used a small SMS corpus for testing and a large newspaper corpus for training. They extracted person names, location names, organization names and verbs, while [11] extracted person names and locations only.

Table 7. Confusion matrix of partial matches on the cross validation testing. Gold tags/columns and predicted tags/rows

Gold\Predicted	TIM	PHO	DAT	LOC	PER	OUT
TIM	**549**	1	4	1	3	37
PHO	0	**220**	0	0	0	8
DAT	17	0	**921**	1	0	74
LOC	0	1	1	**850**	12	216
PER	0	1	0	3	**840**	194
OUT	42	20	70	99	0	**51073**

Table 8. Strict matches on all labels (Left) and partial matches on the main labels (Right)

Strict matches				Partial matches	
Tag	Score	Tag	Score	Tag	Score
B-TIM	86.95	I-TIM	84.78	TIM	91.27
B-PHO	93.14	I-PHO	90.32	PHO	93.62
B-DAT	92.89	I-DAT	69.43	DAT	91.69
B-LOC	81.47	I-LOC	87.94	LOC	83.58
B-PER	86.24	I-PER	68.83	PER	88.79
O	99.19	**Total**	**86.44**	**Total**	**88.85**

Table 9. Comparison with other SMS NER systems

System	Language	Size	F-score
[7]	Chinese	1000	61
[11]	English	?	88
This paper	Swedish	4500	86

10. Conclusion and Future Work

We have presented a named entity recognition system based on the combination of regular expressions and corpus-driven classifiers. One of the major roadblocks we encountered to apply classical machine-

learning techniques lied in the collection a large SMS corpus. This proved very difficult due to the sensitive and personal nature of SMSes. The lack of SMS corpora limits the possibility of training a good model that is well rounded and can deal with most cases. In addition, there are inherent properties of SMSes that make the recognition complex:

- Our POS tagger is trained on newspaper text, whose style and language are quite different from those found in text messages. As a consequence, the tagger accuracy is significantly degraded.
- The brevity and lack of context in text messages impairs the ability to extract information at a high level.

Although we had also to cope with cellular telephone limitations on processing power, storage, and memory, we showed it was possible to reach accuracies competitive with those reported on other kinds of text. We also showed that it was possible to complement existing regex-based implementations with a machine-learning classifier and improve the overall system performance without severe penalties in terms of CPU or memory. This paves the way for a possible replacement of regular expressions with classifiers.

The brevity and lack of context is one of the major difficulties when applying NER on single SMS. We believe that there is a potential in exploring sequences of SMS (conversations) between users as a source of context.

Other contextual information sources specific to cellular telephones are telephone books or call logs for number or name references. Location information can be used both to localize *point of interest* (POI) search as well as for the disambiguation of generic POIs.

References

[1] Bikel D. M., Miller S., Schwartz R., & Weischedel R. (1997). Nymble: High-performance learning name-finder. In *Proceedings of the Fifth ANLP Conference*, pages 194–201.

[2] Carlberger J., & Kann V. (1999). Implementing an efficient part- of-speech tagger. *Software – Practice and Experience*, 29(2):815–832.

[3] Chieu H. L., & Ng H. T. (2003). Named entity recognition with a maximum entropy approach. In *Proceedings of CoNLL-2003*, pages 160–163.

[4] Fan R.-E., Chang K.-W., Hsieh C.-J., Wang X.-R., & Lin C.-J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

[5] Florian R., Ittycheriah A., Jing H., & Zhang T. (2003). Named entity recognition through classifier combination. In *Proceedings of CoNLL-2003*, pages 168–171.

[6] Hård af Segerstad Y. (2002). *Use and adaptation of written language to the conditions of computer-mediated communication*. Doctoral thesis, Göteborg University.

[7] Jiang H., Wang X., & Tian J. (2010). Second-order HMM for event extraction from short message. In *Proceedings of NLDB*, pages 149–156.

[8] Kudoh T., & Matsumoto Y. (2000). Use of support vector learning for chunk identification. In *Proceedings of CoNLL-2000 and LLL- 2000*, pages 142–144.

[9] Mikheev A., Moens M., & Grover C. (1999). Named entity recognition without gazetteers. In *Proceedings of EACL'99*.

[10] OpenNLP. (2004). A package of Java-based NLP tools. http: //opennlp.sourceforge.net/.

[11] Polifroni J., Kiss I., & Adler M. (2010). Bootstrapping named entity extraction for the creation of mobile services. In *Proceedings of LREC*.

[12] Tjong Kim Sang E. F. (2002). Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158.

[13] Tjong Kim Sang E. F., & De Meulder F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*, pages 142–147.