An Environment for Named Entity Recognition and Translation

Filip Graliński
Adam Mickiewicz University
filipg@amu.edu.pl

Krzysztof Jassem
Adam Mickiewicz University
jassem@amu.edu.pl

Michał Marcińczuk
Wrocław University of Technology
marcinczuk@gmail.com

Abstract

We present an environment for the recognition and translation of Named Entities (NEs). The environment consists of a new formalism for the Named Entity Recognition and Translation (NERT), a parsing mechanism that reads the rules, recognizes Named Entities in given texts and suggests their translation, as well as a set of tools for the evaluation. We suggest a method for the evaluation of (sets of) NERT rules that uses raw (not annotated) bilingual corpora.

1 Introduction

The practical goal of our studies has been to develop a mechanism for correct processing of Named Entities in Machine Translation (MT) systems. Vilar et al (2006) claim that incorrect recognition of NE is responsible for approximately 10% of errors made by MT programs. The authors' experience in the field of MT (e.g. Jassem, 2004; Junczys-Dowmut and Graliński, 2007) tells that incorrect treatment of Named Entities is responsible for most serious errors made by MT programs (i.e. errors that make output incomprehensible to human readers). The research aims at improving the quality of Machine Translation by finding a robust solution for processing NEs in MT systems.

The paper is organized as follows:

In Section 2 we describe the issue of Named Entity Recognition (NER). In Section 3 we show the importance of robust NER solutions for Machine Translation. In Section 4 we present a formalism for the description of NERT rules. In Section 5 we show some examples of rules compatible with the grammar. In Section 6 we discuss the problem of NERT evaluation and suggest an evaluation method that does not require a bilingual corpus to be annotated. We end with the reference to future work in Section 7.

2 Named Entity Recognition

Named Entity Recognition consists in automatic determination of continuous fragments of texts (called Named Entities) which refer to information units such as persons, geographical locations, names of organizations, dates, percentages, amounts of money, locations in texts. A NER module is usually expected to provide a markup on boundaries and types of included NEs.

Here is an example of such a markup from Mikheev (1999), cited also in Nadeau (2007):

```
On <Date>Jan 13th</Date>,
<Person>John Briggs Jr</Person>
contacted <Organization>Wonderful
Stockbrockers Inc</Organization>
in <Location>New York</Location>
and instructed them to sell all
his shares in
<Organization>Acme</Organization>.
```

NER is recognized as a field of Natural Language Processing since 1995. The sixth Message Understanding Conference (Grishman and Sundheim, 1996) is usually considered a starting point of the NER history.

First attempts in the field consisted in creation of handcrafted rules (Rau, 1991; Ravin and Wacholder, 1996). In recent years, this idea has been driven out by machine learning techniques. They include:

supervised learning – NER process is learned automatically on large text corpora and then supervised by a human (Asahara and Matsumoto, 2003; McCallum and Li, 2003)

unsupervised learning – NER process is not supervised; instead, existing semantic lexical databases such as WordNet are consulted automatically (Alfonseca and Manandhar, 2002).

semisupervised learning – this "involves a small degree of supervision, such as a set of seeds, for starting the learning process" (Nadeau, 2007).

The survey of NER solutions is well presented by Nadeau and Sekine (2007).

Some research in the field of NER has been done for the Polish language by Piskorski (2005). The author developed a rule-based formalism for the recognition of named entities in Polish texts and handcrafted a set of NER rules for Polish.

3 Named Entity Recognition in MT

Vilar et al (2006) classify errors made by MT systems. The most general classes of errors are: Missing Words, Word Order, Incorrect Words, Unknown Words, Punctuation. The classification does not include the class (or subclass) named Wrong NE Translation. This is probably due to the fact that errors of this type are hard to classify, which in turn is a result of the fact that incorrect NE translation may cause any of the following errors: Word Order, Incorrect Words, Unknown Words. On the other hand, while examining the percentage of error types in various documents, later in the same paper, the authors introduce the error class "Named Entity" and claim that approximately 10% of MT errors may be classified as belonging to the class.

Our experience with the MT system Translatica (www.poleng.pl, www.translatica.pl) shows an even stronger need for correct recognition and translation of NEs. This is particularly important for free-order languages (like those of the Slavonic origin). Incorrect recognition of NE boundaries results in incorrect syntactic analysis of the sentence, which may be shown by the following example:

```
Podała rękę <Person-dative> Pani
Prezes Justynie Kowalskiej</Person-
dative>.
```

The above correct NE recognition leads to the correct translation:

```
She gave a hand to Mrs. Justyna
Kowalska, Chairperson.
```

Suppose that the same source sentence is erroneously processed by an imperfect NER module as:

```
Podała rękę <Person-nominative>Pani
Prezes</Person-nominative> <Person-
dative>Justynie Kowalskiej</Person-
dative>.
```

The above incorrect recognition would lead to the incorrect translation:

```
A chairperson gave a hand to Justy-
na Kowalska.
```

(It is worth noting that NER results for synthetic languages should contain linguistic information, such as case (e.g. Person-dative) to allow correct syntactical parsing.)

Basic research on Named Entity Recognition and Translation has focused on the paradigm of Statistical MT. The ideas presented by Huang (2005), Huang et al (2005) and Al-Onaizan & Knight (2002) aim at statistical methods to collect bilingual lexicons of Named Entities.

We are of the opinion that the purely statistical approach does not solve the NERT robustly because, unlike ordinary words, Named Entities are characterized by fewer repetitions. Instead, we propose the following approach:

1) NERT rules are first handcrafted according to a given formalism;
2) A testing environment allows automatic evaluation of the impact of the rules on translation quality;
3) The rules are enhanced by means of semi-supervised learning.

Babych and Hartley (2003) put forward a hypothesis that MT quality could be significantly improved if NER results were incorporated into MT systems. They carried out an experiment that consisted in incorporating the results of the GALE project into existing commercial MT systems (Systran, Reverso, ProMT). The results of NER tools were manually included into the MT system as Do-Not-Translate lists. The authors reported improvement in the quality of translation.

In 2004, we tried to follow this idea for our MT system, Translatica. Soon, we discovered that Do Not Translate idea is not sufficient for our needs. Thus, we extended the formalism for the rules so that it would allow for translation of (parts of) Named Entities. We also added types of recognized NEs, as they were needed for semantic analysis. Equipped with that, we tried to incorporate the NER rules into our translation system. The results were disappointing (improvement of translation quality in some areas was offset by deterioration in others) and we decided to give up the idea and wait for further development in the area of NER.

A paper by Piskorski (2005) gave some hope of attacking the problem again. However, a complex formalism suggested there in our opinion makes it difficult for linguists or machine learning algorithms to create the rules.

The Spejd formalism invented by Przepiórkowski (2008), intended basically for shallow parsing of a text (not necessarily for the needs of MT), gave us new hopes for handling the problem. Our formalism, presented in the Appendix and described in Section 4, is the extension of the Spejd notation. Our engine, intended for named entity recognition and translation (NERT), based on the formalism, was written from scratch.

4 NERT grammar

In this section, we discuss the components of the NERT grammar. Its detailed description is given in Appendix.

4.1 NERT definitions

NERT definitions aim at simplifying rules by using labels (in curly brackets) instead of longish expressions, e.g.:

```
UpperPL=[A-ZĄĆĘŁŃÓŚŹŻ]
LowerPL=[a-ząćęłńóśźż]

# Polish word starting with
# a upper-case letter:
ProperPL={UpperPL}{LowerPL}*

# Polish first name:
FirstNamePL
=<{ProperPL};sem=first_name>

# Sequence of any number of first
# names and a ProperPL
PersonPL={FirstNamePL}+ <{ProperPL}>
```

4.2 Match part of the rule

Any NERT rule consists of the match part and the action part. The match part consists of the main matching pattern and some optional context patterns:

`Before: pattern`

Imposes the conditions on the context preceding the match in the same sentence – directly or indirectly.

`Left: pattern`

Imposes the conditions on the context preceding the match directly, in the same sentence.

`Match: pattern`

Imposes the conditions on the matching pattern.

`Right: pattern`

Imposes the conditions on the context following the match directly, in the same sentence.

`After: pattern`

Imposes the conditions on the context following the match in the same sentence – directly or indirectly.

`Exists: pattern`

Imposes the conditions on the context occurring anywhere in the same sentence.

4.3 Action part of the rule

The action part of the rule creates the translation for the recognized NE. The translation is executed by copying or modifying groups of the NE, or adding new texts to the equivalents. There are two types of actions in the NERT formalism:

prepend adds "sure" translation of the recognized entity;

append adds "unsure" translation of the recognized entity.

The need for distinguishing between ***append*** and ***prepend*** is that some NEs might be alternatively processed by other translation modules. In such a case ***prepend*** gives priority to the NER module, whereas ***append*** leaves priority to other modules.

4.4 Group Ordering

Each group that occurs in the match part of the rule is assigned an ordering consecutive integer.

Suppose that the analyzed text contains a string *pani Prezes Justynie Marii Kowalskiej* (dat. *Mrs. Justyna Maria Kowalska, Chairperson*). The match part of the rule for such NEs may have the following form:

```
Match: <base~pani> <{ProperPL}>
{FirstNamePL}+ <{ProperPL}>
```

The recognized groups are then ordered as follows: *pani* – 1, *Prezes* – 2, *Justynie Marii* – 3, *Kowalskiej* – 4.

Group ordering integers are referred to in the action part of rules.

4.5 Modifiers

Modifiers operate on the recognized groups:

t – translate the group (use the lexicon)

nom, gen, dat, acc, instr, loc – replace the group with its appropriate inflected case

s[[(**+**|**-**)*Num*][,][(**+**|**-**)*Num*]**]** – cut characters from the given range of the group, e.g. **s[-1**] cuts the last character

u – uppercase the first letter of every token in the group

The ordering integers for recognized groups are preceded by '\', e.g.:

\1:t – translate the first group of the recognized entity (use lexicon)

\3:nom – replace the third group of the recognized entity with its nominative case.

For instance, the following action translates the entity *pani Prezes Justynie Kowalskiej* (assuming that each word matches one group) into *Mrs. Justyna Kowalska, Chairperson*:

```
prepend(Mrs. \3:nom \4:nom, \2:t)
```

The following action translates *2008r* (*r* stands for *rok = year*) into *2008*:

```
prepend(\1:s[-1])
```

4.6 Commands

Commands set the values of attributes of the translated NE. For example, for setting the semantic class of a recognized NE `sem=` command should be used:

```
  prepend(Mrs. \3:nom \4:nom, \2:t;
       sem=person)
```

5 Examples of NERT rules

5.1 Corporation recognition rules

Some named entities denoting corporations may be recognized by their specific endings, such as “S.A” (English: “jsc”). A simple rule may look like this:

```
Match: <{ProperPL}>+ <S.A.>
Action: prepend(\1 \2)
```

This would suffice for correct recognition and translation of the following texts:

```
Indykpol S.A.
Bank Handlowy S.A.
```

However, the above rule would not translate correctly the following text:

```
akcje Banku Handlowego S.A.
(= shares of Bank Handlowy S.A.)
```

Here, the named entity (in bold) should not be just copied. Instead, it should be transformed into the nominative case (*Bank Handlowy S.A.*).

The rule needs adjustment:

```
Match: <{ProperPL}>+ <S.A.>
Action: prepend(\1:nom \2)
```

This solution will still leave open the problem of words starting with an upper-case letter that precede named entities, as in the following two texts:

```
Wiceprezes Zarządu Banku PKO SA;
Zwyczajnego Walnego Zgromadzenia 
Akcjonariuszy INDYKPOL S.A.,
```

The underlined fragments lie beyond the scope of the named entities.

An exemplary rule may look like this:

```
CORP_AFFIX=wiceprezes|akcjonariusz|
zarząd|other words used in terms
denoting (members of) company bod-
ies
CORP_NAME=<{ProperPL};base!~{CORP_A
FFIX}>+
CORP_SUFFIX=S.A.
Match:{CORP_NAME} <{CORP_SUFFIX}>
Action: prepend(\1:nom \2; sem=or-
ganization)
```

The name of the organization may include a name of a city:

```
Bank Przemysłowo-Handlowy w
Krakowie SA.
```

An appropriate NERT rule looks like this:

```
Match: {CORP_NAME} <w> <sem=city>
<{CORP_SUFFIX}>
Action: prepend(\1:nom w \3 \4;
sem=organization)
```

5.2 Temporal expressions

The presented NERT mechanism allows for recognition and translation of temporal expressions. Here are some examples:

```
Match: <1> <base~kwartał> <[0-9]{4}r\.>
Action: prepend(1st quarter of \3:s[-2];
sem=time_period)
```

Example: *1 kwartale 2008r. = 1st quarter of 2008*

```
Match: <4> <kw> <[0-9]{4}>
Action: prepend(4th quarter of \3;
sem=time_period)
```

Example: *4 kw 2010 = 4th quarter of 2010*

```
Match: <base~{MonthPL}> <[0-9]{4}r\.>
Action: prepend(\1:t \2:s[-2];
sem=month)
```

Example: *lutego 1986r.* (gen.) = *February 1986*

```
Match: <[0-9]{1,2}> <base~{MonthPL}>
<[0-9]{4}> <r\.>
Action: prepend(\2:t \1, \3; sem=date)
```

Example: *1 czerwca 2007 r. = June 1, 2007*

5.3 Legal terms

In the machine translation of legal texts, one of the particular problems is the processing of references to act articles, e.g.

```
Original text: Podstawa prawna:
Art. 56 ust. 1 pkt 1 Ustawy z dnia
29 lipca 2005
Expected translation: Legal
grounds: Art. 56.1.1 of the Act of
29 July 2005
```

A NERT rule that processes the above Named Entity (reference to a location in a document) looks like this:

```
Match: <Art\.> <{NUM}> <ust\.>
{NUM} <pkt> <{NUM}> <[Uu]stawy> <z>
<dnia> <[0-9]{1,2}>
<base~{MonthPL}> <[0-9]{4}>
Action: prepend(Art. \2.\4.\6 of
the Act of \10 \11:t \12; sem=docu-
ment)
```

6 Evaluation

In the evaluation of NER systems two measures are referred to most often: precision and recall (sometimes they are merged in one measure, e.g. F-score). Precision is the ratio of the correct guesses to the number of all guesses, recall is the ratio of the correct guesses to the actual number of NEs in the text.

The question is how to treat the partial guesses, for instance the correct recognition of the NE type and the incorrect recognition of the NE boundaries.

There exist two approaches: one approach assigns a point for each correct type recognition

(TYPE) and each correct boundaries recognition (TEXT):

```
correct TYPE incorrect TEXT – 1
point
incorrect TYPE correct TEXT – 1
point
correct TYPE and correct TEXT – 2
points
```

(To calculate the recall, the actual number of NEs is multiplied by two).

In the other approach only guesses that are correct both in TYPE and TEXT are assigned a point:

```
TEXT and TYPE – 1 point
Otherwise – 0 points
```

See Nadeau (2007) for a more detailed discussion on NER evaluation.

In our opinion, it is crucial for MT goals that the boundaries are recognized correctly. Moreover, we need an additional parameter for the correct translation of NE. Therefore we suggest the following method of scoring for the evaluation of NERT:

```
incorrect TEXT – 0
correct TEXT  correct TYPE incor-
rect TRANSLATION – 1 point
correct TEXT incorrect TYPE correct
TRANSLATION – 1 point
correct TEXT correct TYPE correct
TRANSLATION – 2 points
```

Here is an example of how the suggested NERT evaluation may work:

Podała rękę Pani Prezes Justynie Kowalskiej.

Possible NERT recognitions:

```
1) Podała rękę Pani Prezes <TYPE:
PERSON; TRANSLATION: Justynie
Kowalskiej>Justynie
Kowalskiej</PERSON>
```
Score – 0 (correct TYPE, incorrect TEXT, incorrect TRANSLATION)
Recall – 0
Precision – 0

```
2) Podała rękę Pani Prezes <TYPE:
PERSON; TRANSLATION: Justyna Kowal-
ska>Justynie Kowalskiej</PERSON>
```
Score – 0 (correct TYPE, incorrect TEXT, correct TRANSLATION)
Recall – 0
Precision – 0

```
3) Podała rękę <TYPE: PERSON;
TRANSLATION: Mrs. Chairperson
Justyna Kowalska> Pani Prezes
Justynie Kowalskiej</PERSON>
```
Score – 1 (correct TYPE, correct TEXT, incorrect TRANSLATION)
Recall – 0,5
Precision – 0,5

```
4) Podała rękę <TYPE: PERSON;
TRANSLATION: Mrs. Justyna Kowalska,
Chairperson>Pani Prezes Justynie
Kowalskiej</PERSON>
```
Score – 2
Recall – 1
Precision – 1

In order to provide such an evaluation of a NERT module one needs to have access to an appropriately annotated corpus.

We have calculated the Precision of our methods in the following way:

1) Handcraft an initial set of NERT rules;
2) Run the NERT mechanism consistent with the rules against a set of approximately 10 000 Polish sentences from legal documents;
3) Select sentences which contain recognized NEs;
4) Divide the resulting set into two equal parts;
5) Evaluate the set of rules against the first half;
6) Adjust the rules;
7) Evaluate the set of rules against the remaining half.

To evaluate the results, three translators have been requested to verify the translation of all entities recognized by the modules. For each of 3160 entities the translators scored their TEXT, TYPE or TRANSLATION by either 1 point (correct) or 0 points (incorrect).

Table 1. shows the Precision calculated in the strict approach: set 1 point for the named entity with all of TYPE, TEXT and TRANSLATION values equal to 1, set 0 otherwise:

#NE	Max score	Actual score	Precision
3160	3160	2413	76,36%

Table 1.

Table 2 shows Precision, which allows for partial scores.

#NE	Text	Type	Trans lation	Max score	Actual score	Prec.
3160	2853	3002	2515	9480	8370	88,29

Table 2.

Table 3 shows Precision calculated in the method suggested in the paper:

#NE	Max score	Actual score	Precision
3160	6320	5132	81,20%

Table 3.

The above-mentioned method of NERT evaluation has required plenty of human work (it took 6 translators' workdays to estimate our results). We therefore suggest another method for the NERT evaluation – using the METEOR metrics. METEOR (Banerjee, 2005) is the metrics intended for the evaluation of MT algorithms – by comparing their output to reference texts, translated by humans. METEOR is based on BLEU

(Papineni, 2002) but it emphasizes more recall than precision.
The idea has the following merits:
(1) No annotated corpora are needed;
(2) The evaluation may be executed automatically for any selected subset of the NERT rules (including a single rule):

```
1) Take a bilingual "golden standard" corpus of manually translated texts (S | T), the set of all rules ALL, and the set of selected rules SELECTED
2) Translate all sentences from the corpus S:
      2.1 using rules from ALL, obtaining translation T1
      2.2 using rules from the difference: ALL – SELECTED, obtaining translation T2
3. Using METEOR metrics:
     3.1. Compare T1 to T, obtaining METEOR(T1)
     3.2. Compare T2 to T, obtaining METEOR(T2)
4. If METEOR(T1) – METEOR(T2) > F1 (positive threshold)
      then assume SELECTED as useful
   If METEOR(T2) – METEOR(T1) > F2 (negative threshold)
      then assume SELECTED as undesirable
   Otherwise assume SELECTED as unreliable
```

The METEOR evaluation of our preliminary efforts for the whole set of handcrafted Polish-to-English rules are shown in Table 4.

#sentences	avg. score without NERT	#sentences changed with NERT	avg. score with NERT
9794	0.577	1461	0.581

Table 4.

7 Future work

As reported in this paper, the first step of the research has been to create the NERT mechanism and incorporate it into an existing MT system.

The next step would be to create a testing environment, which will allow for the following supervision functionalities:
Rule edition: Edit a rule; Erase a rule; Create an inverted language direction rule.
Rule evaluation: Select a testing text corpus; Use the complete set of rules to test against the golden standard; Use an incomplete set of rules to test against the golden standard; Compare the tests; Use regressive tests.

We will develop the rules for 5 language pairs: Polish-English/French/German/Russian/Spanish. The seed sets of rules will be hand-crafted. Then the rules will be refined statistically. The testing environment will allow for supervision.

7.1 Statistical rule acquisition

We claim that human translation of NE between languages that use the same alphabet is reliable and therefore we want to use human expertise while creating NERT rules. On the other hand, we would like to benefit from existing bilingual corpora. Therefore we intend to develop statistical methods for the acquisition of NERT rules. These rules will be automatically evaluated against a bilingual corpus (see Section 6) and finally verified by humans.

Our method is similar to the semi-supervised learning used by Nadeau (2007). There, the author manually creates seeds of NE, on which the system learns new Named Entities. We shall create the seed rules. The system will learn new rules statistically.

To clarify the intended algorithm we show how it should work on exemplary definitions and a rule R.

```
CORP_AFFIX=<base~(prezes|akcjonariusz|zarząd)>
CORP_SUFFIX=S.A.
R:
Left: <{CORP_AFFIX}>
Match: {CORP_NAME} <{CORP_SUFFIX}>
```

The algorithm is to identify other, so far unknown, affixes that can occur directly before a company name. A new NERT rule (meta-rule) M is designed, where the affix is replaced by a wild character:

```
M:
 Left: <base~.*>
Match: {CORP_NAME} <{CORP_SUFFIX}>
```

Rule R is run against a corpus. Suppose R finds the following matches:

```
          Prezes Polmos S.A.
          Zarząd Citronex S.A
```

The following rules are derived from the meta-rule M and the set of found matches:

```
  M1:    Left: <base~.*>
         Match: <Polmos> <S.A.>
  M2:    Left: <base~.*>
         Match: <Citronex> <S.A.>
```

Now, M1, M2 are run against the corpus, resulting in new matches, e.g.:

```
  Wiceprezes Polmos S.A.
  Sekretariat Citronex S.A.
```

This, in turn, results in new NER rules:

```
  R1: CORP_AFFIX=<base~(wiceprezes)>
         CORP_SUFFIX=S.A.
  R2: CORP_AFFIX=<base~(sekretariat)>
         CORP_SUFFIX=S.A.
```

The ACTION part of the rules is copied from rules prepared by humans.

Acknowledgment

The paper is based on research funded by the Polish Ministry of Science and Higher Education (Grant No 003/R/T00/2008/05).

References

Alfonseca, Enrique and S. Manandhar (2002), Unsupervised Method for General Named Entity Recognition and Automated Concept Discovery. *Proc. Intl. Conference on General WordNet*.

Asahara, Masayuki and Y. Matsumoto (2003), Japanese Named Entity Extraction with Redundant Morphological Analysis. *Proc. Human Language Technology conference - North American chapter of the Association for Computational Linguistics.*

Babych, B., and A. Hartley (2003), Improving Machine Translation quality with automatic Named Entity recognition. *Paper presented at the 7th International EAMT workshop on MT and other language technology tools at the 10th Conference of the European Chapter of the Association for Computational Linguistics EACL 2003, Budapest.*

Banerjee, Satanjeev and Alon Lavie (2005), METEOR: An Automatic Metric For MT Evaluation With Improved Correlation With Human Judgments. Workshop: On Intrinsic And Extrinsic Evaluation Measures For Machine Translation And/or Summarization

Grishman, Ralph and Beth Sundheim (1996), Message Understanding Conference - 6: A Brief History. In: Proceedings of the 16th International Conference on Computational Linguistics, I, 466–471.

Huang F. (2005), Multilingual Named Entity Extraction and Translation from Text and Speech, Ph.D. Thesis. Carnegie Mellon University.

Huang F., Y. Zhang, and S. Vogel (2005), Mining Key Phrase Translations from Web Corpora. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, 483–490.

Jassem K., (2004), Applying Oxford-PWN English-Polish dictionary to Machine Translation, Proceedings of 9th EAMT Workshop, "Broadening horizons of machine translation and its applications", Malta, 26-27 April 2004

Junczys-Dowmut M. and F. Graliński (2007), Using a Treebank Grammar for the Syntactical Annotation of German Lexical Phrases, *Proceedings of 3rd L&T Conference*, Poznań 2007

McCallum, Andrew; Li, W. (2003), Early Results for Named Entity Recognition with Conditional Random Fields, Features Induction and Web-Enhanced Lexicons. In *Proc. Conference on Computational Natural Language Learning*.

Mikheev, Andrei, (1999), A Knowledge-free Method for Capitalized Word Disambiguation. In: *Proc. Conference of Association for Computational Linguistics*.

Nadeau, D. and S. Sekine (2007), A Survey of Named Entity Recognition and Classification. In: Sekine, S. and Ranchhod, E. *Named Entities: Recognition, classification and use*. Special issue of Lingvisticæ Investigationes.

Nadeau, D. (2007), Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision, PhD thesis, University of Ottawa

Al-Onaizan Y. and Knight K. (2002), Machine transliteration of names in Arabic text Full text , *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, Philadelphia, Pennsylvania

Papineni, K., Roukos S., Ward T. and Zhu W.-J. (2002), BLEU: a method for automatic Evaluation of Machine Translation, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02), pp. 311-318*

Piskorski, (2005), Named-Entity Recognition for Polish with SProUT. in: Leonard Bolc, Zbigniew Michalewicz, Toyoaki Nishida (eds.): *Lecture Notes in Computer Science Vol 3490 / 2005: Intelligent Media Technology for Communicative Intelligence: Second International Workshop, IMTCI 2004, Warsaw, September 13-14. Revised Selected Papers, Pages 122-, Springer-Verlag 10/2005,*

Przepiórkowski A. (2008). *Powierzchniowe przetwarzanie języka polskiego*. Warszawa: Akademicka Oficyna Wydawnicza EXIT.

Rau, Lisa F. (1991), Extracting Company Names from Text. In *Proc. Conference on Artificial Intelligence Applications of IEEE*.

Ravin, Yael and N. Wacholder (1996), *Extracting Names from Natural-Language Text*. IBM Research Report RC 2033

Vilar, D., J. Xu, L.F. D'Haro and H. Ney (2006), Error Analysis of Statistical Machine Translation Output. *Proc. Language Resources and Evaluation conference*

Appendix A. NERT Grammar

```
::nert_file::        (definition)*        # set of definitions
                     (rule)+              # non-empty set of rules

::definition::       Name =pattern        # definition of a pattern

::rule::     Rule(Name)                   # descriptive name of the rule
             [Before: pattern]            # pattern preceding the match in the same sentence (optional)
             [Left: pattern]              # left context of the match (optional)
             Match: pattern               # matching text
             [Right: pattern]             # right context of the match (optional)
             [After: pattern]             # pattern following the match in the same sentence (optional)
             [Exists: pattern]            # pattern in the same sentence as the match (optional)
             Action: action_list          # action evoked if the match is found in the specified context

::pattern::          group( group)*       # group is a sequence of tokens that meet the same conditions

::group::            NertRegExp           # a regular expression that may use a NERT definition in brackets

::group:: <condition(;condition)*>        # set of conditions for the pattern to satisfy
          ( *|+|?|{Num(,Num)})?           # number of consecutive tokens that should satisfy the conditions

::condition::        orth|                # orthographical form of the pattern or
                     base                 # canonical form of the pattern being a word or a word phrase
                     (~|!~)NertRegExp     # matches (or not) a NERT regular expression

::condition::        (pos|                # part of speech of the pattern being a word or a word phrase
                     case|                # case
                     num|                 # number
                     gen|                 # gender
                     deg|                 # degree
                     per|                 # person
                     sem                  # semantic class
                     )= Value

::action_list::      do(, do)*            # list of actions which transform source text into target text

::do::       prepend(newText [;Num [; Num ]][; command_list]) |   # add “sure” translation
             append(newText [;Num [; Num ]][; command_list])      # add “unsure” translation

::newText::          expression( expression)*

::expression::       Text |               # new text in the translation output
                     derived              # text derived from source match

::derived::          \Num(:modifier+)?           # copy or modify Numth element of the match

::modifier::| nom | gen | dat | acc | instr | loc |        # set the appropriate case
             t |                                            # translate (use lexicon)
             s[[(+|-)Num][,][(+|-)Num]]                     # cut characters from the text
             u                                              # uppercase the first letter

::command_list::  command (, command)*

::command:: (pos | case | num | gen | deg | per | sem) = (@Num | Value)
                              # copy the attribute value from Numth element of the match or set given value

::command:: all = @Num                                      # copy all attributes values from Numth element
```

Name, Text, Value – any text strings
Num – any number
NertRegExp – a regular expression that may use NERT definitions in brackets.