Named Entity Matching Method Based on the Context-Free Morphological Generator

Jan Kocoń and Maciej Piasecki

Institute of Informatics, Wrocław University of Technology,
Wybrzeże Wyspiańskiego 27, Wrocław, Poland
{jan.kocon,maciej.piasecki}@pwr.edu.pl

Abstract Polish named entities are mostly out-of-vocabulary words, i.e. they are not described in morphological lexicons, and their proper analysis by Polish morphological analysers is difficult.The existing approaches to guessing unknown word lemmas and descriptions do not provide results on a satisfactory level. Moreover, lemmatisation of multi-word named entities cannot be solved by word-by-word lemmatisation in Polish. Multi-word named entity lemmas (e.g. included in gazetteers) often contain word forms that differ from lemmas of their constituents. Such multi-word lemmas can be produced only by tagger- or parser-based lemmatisation. Polish is a language with rich inflection (rich variety of word forms), therefore comparing two words (even these which share the same lemma) is a difficult task. Instead of calculating the value of form-based similarity function between the text words and gazetteer entries, we propose a method which uses a context-free morphological generator, built on the top of the morphological lexicon and encoded as a set of inflection rules. The proposed solution outperforms several state-of-the-art methods that are based on word-to-word similarity functions.

Keywords: Morphological generator, similarity of proper names, word similarity metric, Named Entity Recognition, Information Extraction.

1 Introduction

Lexicons of proper names (henceforth, PNs) are valuable resources for many natural language processing tasks, especially for Named Entity Recognition (henceforth, NER). Most PN inflected forms in text cannot be straightforwardly matched in lexicon. In the case of inflectional langauges, including Polish, *basic morphological forms* (called here shortly lemmas) of PNs are used as the entry forms in PN lexicons. For instance:

1. Inflected PN: $[Lidze_{\text{lemma=0}}\ Polskich_{\text{lemma=0}}\ Rodzin_{\text{lemma=0}}]_{\text{PNlemma=0}}$
2. Lemma: $[Liga_{\text{lemma=1}}\ Polskich_{\text{lemma=0}}\ Rodzin_{\text{lemma=0}}]_{\text{PNlemma=1}}$
3. Lemmatiser: $[Liga_{\text{lemma=1}}\ Polski_{\text{lemma=1}}\ Rodzina_{\text{lemma=1}}]_{\text{PNlemma=0}}$

In the case of multi-word PNs, the PN lemma (2) of the inflected PN form (1) is not identical to the lemma sequence (3) produced by a form lemmatised (e.g.

A. Przepiórkowski and M. Ogrodniczuk (Eds.): PolTAL 2014, LNAI 8686, pp. 34–44, 2014.

based on a tagger) for (1). PN lemmatisation is challenging task and for the Polish language (and for other Slavic languages too) that is not still solved by the existing language tools. This is a specific case of the general unknown word recognition problem, i.e. the recognition of words not covered by the existing morphological analyser [5,10]. Many of such words are infrequent domain-specific PNs. Language resources providing extensive coverage of the inflected PN forms are rare. Having them, comparison of PNs occurring in text with the lexicon PN entries would be easier task, but still discontinuous PNs must be taken into account. However, due to the huge number of PNs and their forms, building a language resource of that kind is laborious and expensive, even for domain-specific lexicons. Moreover, it is hardly possible to find and collect from text enough PN forms to build an extensive lexicon for many specific domains.

PN lexicons (e.g. gazetteers) are relatively large. This increases computational complexity of searching and matching. For the sake of wide applicability, we assume that the unknown word recognition is performed out of context, i.e. we do not use additional external knowledge sources and the information from the occurrence context of an unknown word form. There is also no information whether the token being processed is a true word or a non-word symbol. Polish is a language with rich inflection and each lemma corresponds to many morphological forms on average. Identification of a proper threshold for the similarity measures allowing for proper matching unknown words against the gazetteer entries can be difficult.

Our goal is to develop a method for effective recognition and classification of unknown word forms in Polish texts, in general, with a special focus given to the recognition of the unknown inflected PN forms that are included in a large PN lexicon. The method can be also used in more sophisticated NER tasks [6] e.g. recognition of the words composing multi-word PNs.

2 Related Works

The issue of word-to-word similarity measure has been intensively studied and many solutions were proposed in the literature, including methods dedicated to inflected languages such as Polish. Such metrics take two strings as the input and return a real number from the range $[0, 1]$. Evaluation of their performance is not straightforward – the direct interpretation of the similarity value between two words by the human is difficult, if possible at all. Another option is an indirect evaluation by application – the performance of a language tool utilising the metric in some text processing task is measured.

Several metrics applied to PN matching task were analysed in [1], like *Overlap coefficient*, *Soundex* or *Levensthein*. Recommendations concerning their suitability for different PN data sets were formulated, but the experimental results in [1] on different real data sets showed that there is no single best technique. In [7,8] several known and unique proposed metrics (e.g. *Common Prefix* δ) for Polish were evaluated in a task of assigning named entities (NEs) included in a gazetteer to text words. The best results in one-word NE matching were

obtained with *Common Prefix* δ, mainly due to favouring certain suffix pairs by the δ parameter [7].

In [3] it was shown that a combination of several different similarity metrics can improve the overall result. First, a combination of the selected metrics from the *SimMetrics* library[1] was considered, namely: *Cosine Similarity, Euclidean Distance, Jaro, Jaro-Winkler, Matching Coefficient, Overlap Coefficient, Q-grams Distance, Soundex. Jaro* and *Jaro-Winkler* metrics were also analysed in [7,8]. They also proposed a very efficient *Common Prefix* δ (CP_δ) metric, mentioned earlier, which is based on the longest common prefix of two strings, using simple rules that were derived from the analysis of similarity examples:

$$\mathrm{CP}_\delta(s,t) = \frac{(|\operatorname{lcp}(s,t)| + \delta)^2}{|s| \cdot |t|}$$

where $\operatorname{lcp}(s,t)$ is the longest common prefix of given strings: s & t, δ – a parameter, that equals 1 if one of the two given strings ends with *a*, and the second ends with one of the following: *o, y, ą, ę*, else 0.

The work of [3] is based on the idea of combining individual metrics into a complex one. It was noticed that the dependency of the overall result on the constituents can be very complicated. So, a classifier-based approach was proposed: a vector of individual single metric values is classified into two classes: *similar* and *non-similar*. A decision function value is produced as an additional description. On the basis of the experiments, *Logistic Regression (LR)* classifier was chosen. It provides binary classification (similar/not similar), but it is also possible to obtain decision function value [4], which can be used to describe word pair similarity strength. As a result, the complex word similarity function called NamEnSim[2] was constructed, associated with the initial selection of candidates (performed as a simple morphological filtering applied to compared words). Details of the solution and the previous evaluation results are presented in [3].

In the work presented here we decided to apply the same evaluation process as proposed in [3] in order to show that the solution proposed here outperforms the method combining several similarity functions and the single metric approaches too.

3 Context-Free Morphological Generator

The proposed method originated from the problem of matching NEs against a gazetteer, but it has been next generalized to match any pair of words which share the same lemma. It means that it is not necessary to have a dictionary (e.g. gazetteer) in which all constituents of one and multi-words (e.g. including words comprising NEs) are lemmas, but multi-words can be stored in their proper forms. Whereas the source dictionary remains not processed by lemmatiser or

[1] `http://sourceforge.net/projects/simmetrics`

[2] NamEnSim – Named Entity Similarity function.

stemmer, we try to generate all possible word forms of any unknown text word and match the result against the known set of constituents from gazetteers (e.g. PNs constituents).

Contrary to previous approaches, our method does not use a similarity metric or a preliminary candidate selection for possible similar words. The idea is to generate candidates on the basis of the *ending* (even if it is empty, see Sect. 3.2) of the input word w exclusively. The *ending* of the word w is the part of the word, which is formed by removing the longest common prefix from the set of all possible inflected forms of the word w (assuming that we know that set). Such a collection is called a *group* of the word w or set of words having the same lemma and morphological description consistent with the definition of similarity (see Sect. 4).

In the following, let:

s be an unknown word (string)
w – a word
k – a word ending
$N = \{s^1, s^2, ..., s^{|N|}\}$ – a PN dictionary comprised of a set of strings
$G = \{w^1, w^2, ..., w^{|G|}\}$ – a group – a set of words from the morphological dictionary such that they have the same lemma and morphological description which is consistent with the definition of similarity (see Sect. 4)
$S = \{G^1, G^2, ..., G^{|S|}\}$ – a set of groups
$K_G = \{k_G^1, k_G^2, ..., k_G^{|G|}\}$ – a set of word endings from group G
$C^2_{K_G} = \{\{k_G^1, k_G^2\}, \{k_G^1, k_G^3\}, ..., \{k_G^1, k_G^{|G|}\}, \{k_G^2, k_G^3\}, ..., \{k_G^{|G|-1}, k_G^{|G|}\}\}$ – – 2-combination of the set K_G
$A_S = \sum_{i=0}^{|S|} C^2_{K_{S^i}}$ – a set of all 2-element-combinations of word endings from all groups
$B_S = \{\{A_S^1, a^1\}, \{A_S^2, a^2\}, ..., \{A_S^{|A_S|}, a^{|A_S|}\}\}$ – a set of all 2-combinations of word endings from all groups in a dictionary with the global counter a which is a number of pairs of endings occurrences in all groups.

3.1 Inflection Rules

Consider the following *group* G, presented as a set of triples *word, lemma, tag*[3] (each triple in a separate line):

```
sprawom  sprawa  subst:pl:dat:f
sprawie  sprawa  subst:sg:loc:f
sprawą   sprawa  subst:sg:inst:f
sprawie  sprawa  subst:sg:dat:f
sprawę   sprawa  subst:sg:acc:f
```

[3] In the given example tags come from the National Corpus of Polish Tagset [9] and denote the following attributes (separated by the colon) – *grammatical category : number : case : gender*. For the details of the similarity definition, see Sect. 4.

```
sprawo   sprawa  subst:sg:voc:f
sprawy   sprawa  subst:pl:voc:f
sprawy   sprawa  subst:pl:nom:f
sprawy   sprawa  subst:pl:acc:f
sprawami sprawa  subst:pl:inst:f
sprawach sprawa  subst:pl:loc:f
spraw    sprawa  subst:pl:gen:f
sprawa   sprawa  subst:sg:nom:f
sprawy   sprawa  subst:sg:gen:f
```

The construction of the inflection rules starts from the initial identification of *groups* (see Sect.3) by aggregating words from the dictionary due to their morphological descriptions that are consistent with the definition of the similarity (see Sect. 4). As a result, the set S is built. Next, for each group G in S the longest common prefix lcp_G is determined (for all words belonging to G). For the given example lcp_G = 'spraw'. After that for each word w in G the word ending k is determined by removing lcp_G from the beginning of w. As a result, K_G set is created. For the given example, K_G = {'om','a','ie','ą','ę','o','y','ami','ach',ø}. On the basis of K_G we build a set of all 2-element-combinations of word endings from all groups, denoted as $C^2_{K_G}$. The part of this set from the given example looks as follows:

$C^2_{K_G}$ = {
{'om','a'},{'om','ie'}, {'om','ą'}, ...,
{'om',ø}, {'a','ie'}, {'a','ą'}, ...,
{'a',ø}, ..., {'ach',ø}
}

The last step is to create the set, which contains all possible pairs of endings from groups in S with global counter a (the number of pairs of endings occurrences in all groups), denoted as B_S. Assuming that the group given as an example is the only one:

B_S = {
{{'om','a'}, 1},{{'om','ie'}, 2}, {{'om','ą'}, 1}, ..., {{'om','y'}, 4}, ...,
{{'om',ø}, 1}, {{'a','ie'}, 2}, {{'a','ą'}, 1}, ...,{{'a','y'}, 4}, ...,
{{'a',ø}, 1}, ..., {{'ach',ø}, 1}
}

3.2 Inflected Forms Generator

The main method responsible for the identification of word candidates that are similar to the input string s is the *context-free morphological generator*. This method can be used as a part of a NER language tool. According to the introduced definitions, the algorithm consists of the following steps:

Input:

s – a *word* (unknown) for which a list of similar word candidates is generated,

S – a dictionary of the inflected word forms, on the basis of which the B_S set is created

Output:

$W = \{\{gs^1, as^1\}, \{gs^2, as^2\}, ...\}$ - a set of pairs, where gs is a word candidate similar to s, and as is the number of pairs of $\{gs, s\}$ endings occurrences in S

In the following, let:

$s = \{$'Warszawie'$\}$
$B_S = \{\{$'ie','a'$\}$, 4$\}$,$\{\{$'om','ie'$\}$, 2$\}$, $\{\{$'om','ą'$\}$, 1$\}$, $\{\{$'e',ø$\}$, 1$\}\}$

1. Select a subset BS_S of the set B_S of such elements, in which at least one of the endings is the ending k of word s and each of the endings has a non-zero length.
2. For each element BS_S such as $\{\{k, ks\}, a\}$ (e.g. BS_S =$\{\{\{$'ie','a'$\}$, 4$\}$, $\{\{$'om','ie'$\}$, 2$\}$, $\{\{$'e',ø$\}$, 1$\}\}$):
   (a) create a word gs by removing the ending k from the word s and adding the ending ks,
   (b) add the pair $\{gs, a\}$ to the set W, e.g. $W = \{\{$'Warszawa', 4$\}$, $\{$'Warszawom', 2$\}$, $\{$'Warszawi', 1$\}\}$.
3. Select a subset $BS2_S$ of the set B_S of such elements, in which at least one of the endings is zero length (e.g. $BS2_S = \{\{\{$'e',ø$\}$, 1$\}\}$).
4. For each element $BS2_S$ such as $\{\{\varnothing, ks\}, a\}$:
   (a) create the word gs by adding the ending ks to the word s,
   (b) add the pair $\{gs, a\}$ to W, e.g. $W = \{\{$'Warszawa', 4$\}$, $\{$'Warszawom', 2$\}$, $\{$'Warszawi', 1$\}$, $\{$'Warszawiee', 1$\}\}$.
5. Return the set W.

3.3 Morphological Generator as a RuleSim Similarity Function

In the following, let:

$s = \{$'Warszawie'$\}$
$N = \{$'Warszawa', 'Kraków', 'Wrocław', 'Werszawa', 'Warszawom'$\}$ – a proper names dictionary as a set of strings

In order to calculate the similarity value for words $\langle s_1, s_2 \rangle$, first we have to determine the longest common prefix lcp_s for words: s_1, s_2. Next, the prefix is removed from beginnings of words of the pair $\langle s_1, s_2 \rangle$ and the pair of endings $\langle ks_1, ks_2 \rangle$ is preserved. According to definitions in Sect. 3, if an element $\{\{ks_1, ks_2\}, a\}$ is in B_S, then a is returned as the value of $\text{RuleSim}(s_1, s_2)$ similarity function. In other case 0 is returned.

The inflection rule-based method uses the algorithm described in Sect. 3.2 to generate inflected forms for the input word w. The given set:

$W = \{\{$'Warszawa', 4$\}$, $\{$'Warszawom', 2$\}$, $\{$'Warszawi', 1$\}$, $\{$'Warszawiee', 1$\}\}$

contains the similarity function values for each candidate in W. The result is a subset of $\{gs, a\}$ from W where $gs \in N$. In the following example the result is $\{\{$'Warszawa', 4$\}$, $\{$'Warszawom', 2$\}\}$.

4 Evaluation

We adapted the evaluation process proposed in [3]. The similarity function is applied to find the lemma of an input word w_I or any of its morphological word forms in the NE lexicon if they are included in it. We define *similar* words as words which share the same lemma and agree with respect to all available attributes, except *case* and *number*. We took under consideration only words which belong to the following grammatical classes: *noun (subst), depreciative form (depr), gerund (ger), non-3rd person pronoun (ppron12), 3rd-person pronoun (ppron3), main numeral (num), collective numeral (numcol), adjective (adj), active adj. participle (pact), passive adj. participle (ppas)*[4]. These words can be described with *case* and *number* attributes. Also chosen classes cover all one-word PNs and most of multi-word PNs' components.

For the purposes of the tests, we used *NELexicon*[5] – a very large lexicon of about 1.4 million Polish PNs and NEs available on the Creative Commons licence. It includes not only lemmas, but also inflected word forms for some PNs.

In practice (and also in the prepared evaluation set), the set P of similar words returned by the similarity function for w_I should contain its proper lemma w_L. Moreover, the decision function value for all pairs $\langle w_I, w_O \rangle$ such that $w_O \in P$ and $w_O \neq w_L$ should be lower than for $\langle w_I, w_L \rangle$. This task is different than morphological guessing, e.g. [5] where authors performed generation of lemmas for unknown words on the basis of an *a tergo* index. However, generating of all possible inflected forms of the given word form is a generalization of the morphological guessing (as presented in [5]), in which, instead of *a tergo* index, we use a set of inflection rules (also with the observed rule frequency in morphological dictionary, see Sect. 3.2). The verification process of existence (in the given PN dictionary) is also performed for each generated word form for the input word.

For the sake of comparison with [7], we reproduced test sets from [7,3], i.e. analogical test sets were prepared on the basis of the description in [7,3]. During experiments with single similarity metrics (baseline tests) we obtained the same results as presented by the authors. So the reproduced test sets seem to be a good approximation of the original ones and can be used for the comparison of our own solutions with the methods of [7,3]. They concentrated on selecting a lemma (from the search space) for a PN inflected form on the input. On the basis of *NELexicon* the following test sets of pairs: lemma – inflected form, were generated:

[4] Morphosyntactic tags come from the National Corpus of Polish Tagset [9]

[5] `http://nlp.pwr.wroc.pl/nelexicon`

person_first_nam – a set of Polish first names,
country_nam – Polish country names,
city_nam – Polish city names – not used in [7],
person_full_nam – Polish person full names (multi-word).

Because the considered similarity functions are limited only to one-word PNs, the last test set was not used. Following [7,8] all experiments were performed in two variants, for two different search space sizes:

a *small search space (0 mode)* – only base forms of the test examples,
a *large search space (1 mode)* – all base forms from the named entity category.

Table 1 shows the size of test sets and search spaces for different experiment modes and categories.

Table 1. Test sets and search spaces for different experiment modes and categories

Category	Size		
	Small search space (s_space)	Large search space (l_space)	Tests
person_first_nam	480	15208	1720
country_nam	157	332	621
city_nam	8144	38256	30323

Let:

a be the number of all test examples,
s – the number of tests, in which a single result was returned,
m – the number of tests with more than one result returned,
sc (*single correct*) – the number of tests, in which a single result was returned and it was correct,
mc (*multiple and correct*) – the number of tests with more than one result, but including the correct one,
$mc2$ (*multiple with best one correct*) – the number of tests with more than one result and with the correct result as the top one (i.e. having the highest decision function value assigned).

We used the three measures proposed by [7]:

All answer accuracy: $\mathrm{AA} = \frac{sc}{s+m}$
Single result accuracy: $\mathrm{SR} = \frac{sc}{s}$
Relaxed all answer accuracy: $\mathrm{RAA} = \frac{sc+mc}{s+m}$

In a similar way to [3] we decided to use the modified version of AA measure (by adding $mc2$ parameter) in order to better analyse the cases in which more than one result was returned. Because the complex similarity function and the function utilising morphological generator always return a decision function value as a value of similarity (not only binary decision), we used mACC measure (see [3]) aimed at the comparison of different similarity functions in the domain of their values:

Modified all answer accuracy: $\text{mAA} = \frac{sc+mc2}{s+m}$
Modified global accuracy: $\text{mACC} = \frac{sc+mc2}{a}$

As a baseline we used the similarity metrics included in *SimMetrics* package (as also proposed in [3]). The experiment was performed on the person first name test set in two variants: with small (person_first_nam0) and large (person_first_nam1) search space (see Table 1 for the given sets details). The results are presented in Table 4.

Table 2. Baseline test for person_first_nam with small (s_space) and large (l_space) search space

Similarity metric	s_space variant			l_space variant		
	AA	SR	RAA	AA	SR	RAA
ChapmanLengthDeviation	0.32260	0.57387	0.53503	0.06	0.30721	0.40293
Jaro	0.83164	0.86895	0.87062	0.30501	0.64447	0.63590
JaroWinkler	0.84859	0.87275	0.87514	0.55599	0.64407	0.66517
MatchingCoefficient	0.74011	0.96608	0.97119	0.32133	0.93148	0.94260
Soundex	0.66158	0.97502	0.97401	0.63084	0.68395	0.69893
OverlapCoefficient	0.76780	0.82815	0.83164	0.61171	0.67016	0.68149
QGramsDistance	0.85198	0.86717	0.86893	0.61902	0.67568	0.68542

Single metrics expressed good accuracy in tests with small search spaces, but the results are not satisfactory in tests with large search spaces. Values for the modified evaluation measures are not presented in Table 4.

Table 4 presents the size of sets returned by NamEnSim5. Evaluation measures are based on these results. Resources with suffix *0* are variants with small search space, and with suffix *1* are variants with large search space.

Table 3. Examples of sets size returned in experiments for NamEnSim5 (description of parameteres in Sect. 4)

similarity metric	resource	a	s	m	sc	mc	mc2
NamEnSim5	person_first_nam0	1720	1296	372	1289	371	356
	person_first_nam1	1720	758	923	751	909	746
	country_nam0	621	572	27	572	27	25
	country_nam1	621	501	99	500	99	93
	city_nam0	29492	20495	8121	20258	8022	6674
	city_nam1	29492	11858	17177	11579	16701	12404

Table 4 shows values of the evaluation measures. Presented results for the method utilising morphological generator (RuleSim) are compared with the similarity function NamEnSim5 described in [3] and single similarity metric (CP_δ) described in [7]. RuleSim significantly outperforms other methods in most categories and cases, especially for two important measures: mAA (modified all

answer accuracy) and mACC (modified global accuracy). NamEnSim5 has better global accuracy only for test cases with a small search space, but in practice (where the search space is large) the results of NamEnSim5 and CP_δ are not satisfactory. In most cases the results achieved by RuleSim are much better than similar results achieved on lemmatisation of Slovene words for which the method proposed in [2] (utilising a statistics-based trigram tagger) achieves the accuracy of 81%. RuleSim results are also better than one-word similarity methods described in [7,8,3].

Table 4. Evaluation results for the method utilising a morphological generator (RuleSim) in comparison with CP_δ and NamEnSim5

similarity metric	resource	mAA	SR	RAA	mACC
NamEnSim5	person_first_nam0	0.9862	0.9946	0.9952	0.9564
	person_first_nam1	0.8905	0.9908	0.9875	0.8703
	country_nam0	**0.9967**	**1.0000**	**1.0000**	0.9614
	country_nam1	0.9883	0.9980	0.9983	**0.9549**
	city_nam0	0.9412	0.9884	0.9883	0.9132
	city_nam1	0.8260	0.9765	0.9740	0.8132
CP_δ	person_first_nam0	0.9683	0.9810	0.9812	0.9593
	person_first_nam1	0.7915	0.8636	0.8662	0.7878
	country_nam0	0.9885	0.9950	0.9951	**0.9678**
	country_nam1	0.9672	0.9866	0.9869	0.9501
	city_nam0	0.9175	0.9322	0.9306	0.9168
	city_nam1	0.7734	0.8168	0.8170	0.7733
RuleSim	person_first_nam0	**0.9924**	**0.9981**	**0.9982**	**0.9826**
	person_first_nam1	**0.9366**	**1.0000**	**0.9982**	**0.9273**
	country_nam0	0.9950	**1.0000**	**1.0000**	0.9533
	country_nam1	**0.9899**	**1.0000**	**1.0000**	0.9485
	city_nam0	**0.9880**	**0.9998**	**0.9998**	**0.9328**
	city_nam1	**0.8887**	**0.9984**	**0.9986**	**0.8401**

5 Summary

Experiments have shown that it is possible to obtain reasonable results, even better than previously proposed complex similarity function [3], without using a complete morphological dictionary. The quality of the morphological base forms (lemmas) produced by the proposed generator for the unknown words (i.e. not covered by the morphological analyser) is very high.

The achieved results are better than the results of NamEnSim5 complex similarity function which is based on Logistic Regression applied to combine results produced by the selected single similarity metrics (see [3]). The proposed method does not require linguistic knowledge for the identification of endings and should achieve similar results for other languages (where the suffix plays the major role in word inflection), especially for languages with rich inflection (e.g. Slavic).

As in the previous work, we used realistic data set (also containing misspelled words) that might cause slightly worse results than expected, but still the results achieved by morphological generator are better than those of the methods proposed in [2,3,7,8].

References

1. Christen, P.: A comparison of personal name matching: Techniques and practical issues. In: International Conference on Data Mining Workshops, pp. 290–294 (2006)
2. Džeroski, S., Erjavec, T.: Learning to Lemmatise Slovene Words. In: Cussens, J., Džeroski, S. (eds.) LLL 1999. LNCS (LNAI), vol. 1925, pp. 69–88. Springer, Heidelberg (2000), http://dx.doi.org/10.1007/3-540-40030-3_5
3. Kocoń, J., Piasecki, M.: Heterogeneous Named Entity Similarity Function. In: Sojka, P., Horák, A., Kopeček, I., Pala, K. (eds.) TSD 2012. LNCS, vol. 7499, pp. 223–231. Springer, Heidelberg (2012), http://dx.doi.org/10.1007/978-3-642-32790-2_27
4. Lubenko, I., Ker, A.D.: Steganalysis using logistic regression. In: Proc. SPIE 7880, p. 78800K (2011)
5. Piasecki, M., Radziszewski, A.: Polish Morphological Guesser Based on a Statistical A Tergo Index. In: Proceedings of the International Multiconference on Computer Science and Information Technology — 2nd International Symposium Advances in Artificial Intelligence and Applications (AAIA 2007), pp. 247–256 (2007), http://www.proceedings2007.imcsit.org/pliks/150.pdf
6. Piskorski, J.: Named-Entity Recognition for Polish with SProUT. In: Bolc, L., Michalewicz, Z., Nishida, T. (eds.) IMTCI 2004. LNCS (LNAI), vol. 3490, pp. 122–133. Springer, Heidelberg (2005), http://dx.doi.org/10.1007/11558637_13
7. Piskorski, J., Sydow, M.: Usability of String Distance Metrics for Name Matching Tasks in Polish. In: Human Language Technologies as a Challenge for Computer Science and Linguistics, Proc. of LTC 2007, pp. 403–407. Wydawnictwo Poznańskie, Sp. z o.o (2007)
8. Piskorski, J., Sydow, M., Kupść, A.: Lemmatization of Polish person names. In: Proceedings of the Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies, ACL 2007, pp. 27–34. Association for Computational Linguistics, Stroudsburg (2007), http://dl.acm.org/citation.cfm?id=1567545.1567551
9. Przepiórkowski, A., Bańko, M., Górski, R.L., Lewandowska-Tomaszczyk, B. (eds.): Narodowy Korpus Języka Polskiego. Wydawnictwo Naukowe PWN, Warsaw (2012)
10. Woliński, M.: Morfeusz – a Practical Tool for the Morphological Analysis of Polish. In: Kłopotek, M.A., Wierzchoń, S.T., Trojanowski, K. (eds.) Intelligent Information Processing and Web Mining. AISC, vol. 5, pp. 511–520. Springer, Berlin (2006)