Lemmatization of Polish Person Names

Jakub Piskorski
European Commission
Joint Research Centre
Via Fermi 1
21020 Ispra, Italy
Jakub.Piskorski@jrc.it

Marcin Sydow
Polish-Japanese Institute
of Information Technology
Koszykowa 86
02-008 Warsaw, Poland
msyd@pjwstk.edu.pl

Anna Kupść
Université Paris3/LLF, PAS ICS
Case Postale 7031
2, place Jussieu
75251 Paris Cedex 05
akupsc@univ-paris3.fr

Abstract

The paper presents two techniques for lemmatization of Polish person names. First, we apply a rule-based approach which relies on linguistic information and heuristics. Then, we investigate an alternative knowledge-poor method which employs string distance measures. We provide an evaluation of the adopted techniques using a set of newspaper texts.

1 Introduction

Proper names constitute a significant part of natural language texts (estimated to about 10% in newspaper articles) and are important for NLP applications, such as Information Extraction, which rely on automatic text understanding.[1] In particular, coreference resolution (e.g., identifying several name variants as referring to the same entity) plays a crucial role in such systems. Although automatic recognition of proper names in English, French and other major languages has been in the research focus for over a decade now, cf. (Bikel et al., 1997), (Borthwick, 1999), (Li et al., 2003), only a few efforts have been reported for Slavic languages, cf. (Cunningham et al., 2003) (Russian and Bulgarian), (Piskorski, 2005) (Polish). Rich inflection and a more relaxed word order make recognition of proper names in Slavic more difficult than for other languages. Moreover, inflection of proper names is usually quite different from common nouns, which complicates the lemmatization process necessary for correct coreference resolution. In this paper, we focus on lemmatization of Polish person names, the most idiosyncratic class of proper names in this language. First, we report results of a rule-based symbolic approach. We apply different heuristics, mostly based on the internal (morphological and syntactic) structure of proper names but also on the surrounding context. Sometimes, however, the required information is not available, even if the entire document is considered, and lemmatization cannot be performed. Therefore, we experimented with various knowledge-poor methods, namely string distance metrics, in order to test their usefulness for lemmatization of Polish person names as an alternative technique, especially for cases where document-level heuristics are insufficient.

Lemmatization of proper names in Slavic has not attracted much attention so far but some work has been done for Slovene: (Erjavec et al., 2004) present a machine-learning approach to lemmatization of unknown single-token words, whereas (Pouliquen et al., 2005) report on a shallow approach to find base forms.

The organization of the paper is as follows. First, we present a description of phenomena which make lemmatization of Polish person names a difficult task. Next, a rule-based approach and its evaluation are presented. Then, various string distance metrics are introduced, followed by the results of experiments on newspaper texts. The final section presents conclusions and perspectives for future work.

[1]The research presented in this paper was partially founded by the Ministry of Education and Science (Poland), grant number 3T11C00727.

Balto-Slavonic Natural Language Processing 2007, June 29, 2007, pages 27–34,
Prague, June 2007.

case	male name	female name
nom	*Kazimierz Polak*	*Kazimiera Polak*
gen	*Kazimierza Polaka*	*Kazimiery Polak*
dat	*Kazimierzowi Polakowi*	*Kazimierze Polak*
acc	*Kazimierza Polaka*	*Kazimierę Polak*
ins	*Kazimierzem Polakiem*	*Kazimierą Polak*
loc	*Kazimierzu Polaku*	*Kazimierze Polak*
voc	*Kazimierzu Polaku*	*Kazimiero Polak*

Table 1: Declension of Polish male vs. female names

case	sg	pl	sg	pl
nom	*gołąb*	*gołębie*	*Gołąb*	*Gołąbowie*
gen	*gołębia*	*gołębi*	*Gołąba*	*Gołąbów*
dat	*gołębiowi*	*gołębiom*	*Gołąbowi*	*Gołąbom*
acc	*gołębia*	*gołębie*	*Gołąba*	*Gołąbów*
ins	*gołębiem*	*gołębiami*	*Gołąbem*	*Gołąbami*
loc	*gołębia*	*gołębie*	*Gołąbiu*	*Gołąbach*
voc	*gołębiu*	*gołębie*	*Gołąb*	*Gołąbowie*

Table 2: Common noun vs. person name inflection

2 Declension Patterns of Polish Person Names

Polish is a West Slavic language with rich nominal inflection: nouns and adjectives are inflected for case, number and gender. There are 7 cases, 2 numbers and traditionally 3 genders are distinguished: masculine, feminine and neuter. Just like common nouns, Polish person names undergo declension but their inflectional patterns are more complicated. A typical Polish name consists of a first name and a last name; unlike in Russian or Bulgarian, there are no patronymics. Additionally, titles (e.g., *dr* 'Phd', *inż.* 'engineer', *prof.* 'professor') or honorific forms (*pan* 'Mr.' or *pani* 'Mrs./Miss') are often used. In general, both the first and the last name can be inflected, e.g., *Jan Kowalski* (nominative) vs. *Jana Kowalskiego* (genitive/accusative). If the surname is also a regular word form, things get more complicated. Whether the last name can be inflected in such cases depends on several factors, e.g., on the gender of the first name, a category (part-of-speech) and gender of the (common) word used as a surname. For instance, if the surname is a masculine noun, it is inflected only if the first name is also masculine. This is illustrated in Table 1 with declension of the male name *Kazimierz Polak* 'Casimir Pole' and its variant with the female first name *Kazimiera*.

If the surname is an adjective (e.g., *Niski* 'Short'), it is inflected (according to the adjectival paradigm) and agrees in gender with the first name, i.e., male and female last name forms are different (e.g., *Niski* 'Short' (masc.) vs. *Niska* 'Short' (fem.)). The declension of foreign surnames may strongly depend on their origin, and in particular on the pronunciation. For example, the name *Wilde* is pronounced differently in English and German, which impacts its declension in Polish. If it's of English origin, a nominal declension is applied, i.e., *Wilde'a* (gen.), whereas if it comes from German, an adjective-like declension is adopted: *Wildego* (gen.).

Declension of surnames which are also common nouns can be different from the declension of common nouns.[2] In Table 2, we present a comparison of the common noun *gołąb* 'dove' in singular and plural with the corresponding forms used for the surname. A comprehensive overview of this rather intriguing declension paradigm of Polish names is given in (Grzenia, 1998).

Finally, first name forms present problems as well. Foreign masculine first names, whose pronounced version ends in a consonant or whose written version ends in *-a*, *-o*, *-y* or *-i* do in general get inflected (e.g., *Jacques* (nom.) vs. *Jacques'a* (gen./acc.)), whereas names whose pronounced version ends in a vowel and are stressed on the last syllable (e.g., *François*) usually do not change form. For female first names created from a male first name, e.g., *Józef* (masc.) vs. *Józefa* (fem.), there is a frequent homonymy between the nominative form of the female name and the genitive/accusative form of the corresponding male form, e.g., *Józefa* is nominative of *Józefa* (fem.) and genitive/accusative of *Józef* (masc.).

3 Rule-Based Approach to Person Name Lemmatization

3.1 Experiment

Our rule-based approach to person name lemmatization exploits existing resources (a dictionary of first names and contextual triggers) and relies on contextual information (heuristics). It has been implemented using SProUT, a shallow processing platform, integrated with a Polish morphological anal-

[2]The declension of such surnames depends on the local tradition and sometimes can be identical with the pattern used for common nouns.

yser (Piskorski et al., 2004). For first names, all inflected forms of the most frequent Polish first names are stored in a database so a simple gazetteer look-up associates names with the corresponding base form. We also used a list of ca 30 000 foreign first names (nominative forms). For last names, we applied several heuristic rules in order to recognize and produce their base forms. First, we identify most common types of Polish surnames, e.g., capitalized words ending in *-skiego*, *-skim*, *-skiemu* or *-icza*, *-iczem*, *-iczu* (typical last name suffixes), and convert them to the corresponding base forms (i.e., words ending in *-ski* and *-icz*, respectively). In this way, a significant number of names can be lemmatized in a brute-force manner.

For all remaining surnames, more sophisticated rules have to be applied. As discussed in sec. 2, these rules have to take into account several pieces of information such as part-of-speech and gender of the (common) word which serves as a surname, but also gender of the first name. The major problem we encountered while applying these rules is that the information necessary to trigger the appropriate rule is often missing. For example, in sentence (1), inferring gender of the surname/first name could involve a subcategorization frame for the verb *powiadomić* 'inform', which requires an accusative NP argument. In this way we might possibly predict that the base form of *Putina* is *Putin*, as *-a* is the typical accusative ending of masculine names. Since the subcategorization lexicon is not available, such instances are either not covered or different heuristics are employed for guessing the base form.

(1) *Powiadomiono wczoraj wieczorem V. Putina o ataku.*
informed yesterday evening V. Putin$_{acc}$ about attack
'Yesterday evening they informed V. Putin about the attack.'

Additionally, grammar rules may produce variants of recognized full person names. For example, for the full name *CEO dr Jan Kowalski* the following variants can be produced: *Kowalski*, *CEO Kowalski*, *dr Kowalski*, etc. As the grammar rules always return the longest match, a shorter form may not be recognized. The produced variants are therefore used in the second pass through the text in order to identify 'incomplete' forms. As no morphological generation is involved, only base forms can be identified in this way. The system evaluation indicates that 23.8% of the recognized names were identified by this partial coreference resolution mechanism.

An analysis of incorrectly recognized named entities (NEs) revealed that major problems concerned (a) classical ambiguities, such as a proper name vs. a common word, and (b) person vs. organization name, caused by a specific word order and a structural ambiguity of phrases containing NEs. Let us consider the following examples to illustrate the problems.

(2) *Dane Federalnego Urzędu Statystycznego*
Data$_{nom}$ federal$_{gen}$ office$_{gen}$ statistical$_{gen}$
'Data of the federal office for statistics'

(3) *prezes Della*
president$_{nom}$ Dell$_{gen}$
'president of Dell'

(4) *kanclerz Austriaków*
chancellor$_{nom}$ Austrians$_{gen}$
'chancellor of the Austrians'

(5) *... powiedział prezes spółki Kruk*
said president$_{nom}$ company$_{gen}$ Kruk$_{nom}$
'. . . said the president of Kruk company / Kruk, the president of the company'

The text fragment *Dane Federalnego* in (2) is recognized by the grammar as a person name since *Dane* is a gazetteer entry for a foreign (English) first name. Consequently, *Federalnego Urzędu Statystycznego* could not be recognized as an organization name. Potentially, heuristics solving such NE overlapping collisions could improve the precision. Similar techniques have been applied to other languages. In (3) and (4) the names *Della* 'of Dell' and *Austriaków* 'of Austrians' were erroneously recognized as surnames. The rule matching a token representing a title followed by a capitalized word, adopted for English person names, is less reliable for Polish due to declension of proper names and lack of prepositions in genitive constructions. One solution to this problem would involve matching *Della* and *Austriaków* with their base forms (*Dell* and *Austriacy*, resp.), which might appear in the immediate context. In this way, the name type could be validated. However, a corpus inspection revealed that quite frequently no base form appears in the same document. The last example, (5), illustrates another problem, which is even harder to solve. The phrase *prezes* *spółki Kruk* is structurally ambiguous, i.e., it can be bracketed as [*prezes* [*spółki Kruk*]] or [[*prezes spółki*] *Kruk*]. Consequently, the name *Kruk* might either refer to a company name ('... said the president of the Kruk company') or to a person name ('... said Kruk, the president of the company'). Inferring the proper interpretation might not be possible even if we consider the subcategorization frame of the verb *powiedzieć* 'to say'.

3.2 Evaluation

For evaluation of recognition and lemmatization of person names, a set of 30 articles on various topics (politics, finance, sports, culture and science) has been randomly chosen from *Rzeczpospolita* (Weiss, 2007), a leading Polish newspaper. The total number of person name occurrences in this document set amounts to 858. Evaluation of recognition's precision and recall yielded 88.6% and 82.6%, respectively. Precision of lemmatization of first names and surnames achieved 92.2% and 75.6%, respectively. For 12.4% of the recognized person names more than one output structure was returned. For instance, in case of the person name *Marka Belki*, the first name *Marka* is interpreted by the gazetteer either as an accusative form of the male name *Marek* or as a nominative form of a foreign female name *Marka*. In fact, 10% of the Polish first-name forms in our gazetteer are ambiguous with respect to gender. As for the last name *Belki*, it is a genitive form of the common Polish noun *belka* 'beam', so the base form can be obtained directly. Nevertheless, as inflection of proper names differs from that of common nouns, various combinations of the regular noun *Belka* and the special proper name form *Belki* are possible, which increases ambiguity of the identified form. All possible lemmatizations are as follows:

(6) *Marek Belka* (masc.),
Marka Belka (fem.),
Marek Belki (masc.),
Marka Belki (fem.)

A good heuristics to reduce such ambiguous lemmatizations is to prioritize rules which refer to morphological information over those which rely solely on orthography and/or token types.

4 Application of String Distance Metrics for Lemmatization

Since knowledge-based lemmatization of Polish NEs is extremely hard, we also explored a possibility of using string distance metrics for matching inflected person names with their base forms (and their variants) in a collection of document, rather than within a single document. The rest of this section describes our experiments in using different string distance metrics for this task, inspired by the work presented in (Cohen et al., 2003) and (Christen, 2006).

The problem can be formally defined as follows. Let A, B and C be three sets of strings over some alphabet Σ, with $B \subseteq C$. Further, let $f : A \rightarrow B$ be a function representing a mapping of inflected forms (A) into their corresponding base forms (B). Given, A and C (the search space), the task is to construct an approximation of f, namely $\widehat{f} : A \rightarrow C$. If $\widehat{f}(a) = f(a)$ for $a \in A$, we say that $\widehat{f}$ returns the correct answer for a; otherwise, $\widehat{f}$ is said to return an incorrect answer. For another task, a multi-result experiment, we construct an approximation $f^* : A \rightarrow 2^C$, where f^* returns the correct answer for a if $f(a) \in f^*(a)$.

4.1 String distance metrics

In our experiments, we have explored mainly character-level string metrics[3] applied by the database community for record linkage.

Our point of departure is the well-known *Levenshtein* edit distance metric specified as the minimum number of character-level operations (insertion, deletion or substitution) required for transforming one string into another (Levenshtein, 1965) and *bag distance* metric (Bartolini et al., 2002) which is a time-efficient approximation of the *Levenshtein* metric. Next, we have tested the *Smith-Waterman* (Smith and Waterman, 1981) metric, which is an extension of *Levenshtein* metric and allow a variable cost adjustment to edit operations and an alphabet mapping to costs.

Another group of string metrics we explored is based on a comparison of character-level *n*-grams in two strings. The *q-gram* metric (Ukkonen, 1992) is

[3]Distance (similarity) metrics map a pair of strings s and t to a real number r, where a smaller (larger) value of r indicates greater (lower) similarity.

computed by counting the number of q-grams contained in both strings. An extension to q-grams is to add positional information, and to match only common q-grams that occur at a specified distance from each other (*positional q-grams*) (Gravano et al., 2001). Finally, the *skip-gram* metric (Keskustalo et al., 2003) is based on the idea that in addition to forming bigrams of adjacent characters, bigrams that skip characters are considered as well. *Gram classes* are defined that specify what kind of skip-grams are created, e.g. $\{0, 1\}$ class means that regular bigrams (0 characters skipped) and bigrams that skip one character are formed. We have explored $\{0, 1\}$, $\{0, 2\}$ and $\{0, 1, 2\}$ gram classes.

Taking into account the Polish declension paradigm, we also added a basic metric based on the longest common prefix, calculated as follows:

$$CP_\delta(s,t) = ((|lcp(s,t)| + \delta)^2/(|s| \cdot |t|),$$

where $lcp(s,t)$ denotes the longest common prefix for s and t. The symbol δ is a parameter for favoring certain suffix pairs in s (t). We have experimented with two variants: CP_{δ_1} with $\delta = 0$ and CP_{δ_2}, where δ is set to 1 if s ends in: *o*, *y*, *ą*, *ę*, and t ends in an *a*, or 0 otherwise. The latter setting results from empirical study of the data and the declension paradigm.

For coping with multi-token strings, we tested a similar metric called *longest common substrings* (LCS) (Christen, 2006), which recursively finds and removes the longest common substring in the two strings compared, up to a specified minimum length. Its value is calculated as the ratio of the sum of all found longest common substrings to the length of the longer string. We extended LCS by additional weighting the lengths of the longest common substrings. The main idea is to penalize the longest common substrings which do not match the beginning of a token in at least one of the compared strings. In such cases, the weight for $lcs(s,t)$ (the longest common substring for s and t) is computed as follows. Let α denote the maximum number of non-whitespace characters which precede the first occurrence of $lcs(s,t)$ in s or t. Then, $lcs(s,t)$ is assigned the weight:

$$w_{lcs(s,t)} = \frac{|lcs(s,t)| + \alpha - \max(\alpha, p)}{|lcs(s,t)| + \alpha}$$

where p has been experimentally set to 4. We refer to the 'weighted' variant of LCS as $WLCS$.

Good results for name-matching tasks (Cohen et al., 2003) have been reported using the *Jaro* metric and its variant, the *Jaro-Winkler* (JW) metric (Winkler, 1999). These metrics are based on the number and order of common characters in two compared strings. We have extended the *Jaro-Winkler* metric to improve the comparison of multi-token strings. We call this modification JWM and it can be briefly characterized as follows. Let $J(s,t)$ denote the value of the *Jaro* metric for s and t. Then, let $s = s_1 \ldots s_K$ and $t = t_1 \ldots t_L$, where s_i (t_i) represent i-th token of s and t respectively, and assume, without loss of generality, $L \leq K$. $JWM(s,t)$ is defined as:

$$JWM(s,t) = J(s,t) + \delta \cdot boost_p(s,t) \cdot (1 - J(s,t))$$

where δ denotes the common prefix adjustment factor and $boost_p$ is calculated as follows:

$$boost_p(s,t) = \frac{1}{L} \cdot \sum_{i=1}^{L-1} \min(|lcp(s_i,t_i)|, p) + \frac{\min(|lcp(s_L, t_L..t_K)|, p)}{L}$$

The main idea behind JWM is to boost the *Jaro* similarity for strings with the highest number of agreeing initial characters in the corresponding tokens in the compared strings.

Finally, for multi-token strings, we tested a recursive matching pattern, known also as *Monge-Elkan* distance (Monge and Elkan, 1996). The intuition behind this measure is the assumption that a token in s (strings are treated as sequences of tokens) corresponds to a token in t which has the highest number of agreeing characters. The similarity between s and t is the mean of these maximum scores. Two further metrics for multi-token strings were investigated, namely *Sorted-Tokens* and *Permuted-Tokens*. The first one is computed in two steps: (a) first, tokens forming a full string are sorted alphabetically, and then (b) an arbitrary metric is applied to compute the similarity for the 'sorted' strings. The latter compares all possible permutations of tokens forming the full strings and returns the calculated maximal similarity value.

A detailed description of string metrics used here is given in (Christen, 2006) and in (Piskorski et al., 2007).

4.2 Test Data

For the experiments on coreference of person names, we used two resources: (a) a lexicon of the most frequent Polish first names (PL-F(IRST)-NAMES) consisting of pairs of an inflected form and the corresponding base form, and (b) an analogous lexicon of inflected full person names (first name + surname) (PL-FULL-NAMES).[4] The latter resource was created semi-automatically as follows. We have automatically extracted a list of 22485 full person-name candidates from a corpus of 15724 on-line news articles from *Rzeczpospolita* by using PL-F-NAMES lexicon and an additional list of 30000 uninflected foreign first names. Subsequently, we have randomly selected a subset of about 1900 entries (inflected forms) from this list.

In basic experiments, we simply used the base forms as the search space. Moreover, we produced variants of PL-F-NAMES and PL-FULL-NAMES by adding to the search space base forms of foreign first names and a complete list of full names extracted from the *Rzeczpospolita* corpus, respectively. Table 3 gives an overview of our test datasets.

Dataset	#inflected	#base	search space
PL-F-NAMES	5941	1457	1457
PL-F-NAMES-2	5941	1457	25490
PL-FULL-NAMES	1900	1219	1219
PL-FULL-NAMES-2	1900	1219	2351
PL-FULL-NAMES-3	1900	1219	20000

Table 3: Dataset used for the experiments

4.3 Evaluation Metrics

Since for a given string more than one answer can be returned, we measured the accuracy in three ways. First, we calculated the accuracy on the assumption that a multi-result answer is incorrect and we defined *all-answer accuracy* (AA) measure which penalizes multi-result answers. Second, we measured the accuracy of single-result answers (*single-result accuracy* (SR)) disregarding the multi-result answers. Finally, we used a weaker measure which treats a multi-result answer as correct if one of the results in the answer set is correct (*relaxed-all-answer accuracy* (RAA)).

[4] Inflected forms which are identical to their corresponding base form were excluded from the experiments since finding an answer for such cases is straightforward.

Let s denote the number of strings for which a single result (base form) was returned. Analogously, m is the number of strings for which more than one result was returned. Let s_c and m_c denote, respectively, the number of correct single-result answers returned and the number of multi-result answers containing at least one correct result. The accuracy metrics are computed as: $AA = s_c/(s + m)$, $SR = s_c/s$, and $RAA = (s_c + m_c)/(s + m)$.

4.4 Experiments

We started our experiments with the PL-F-NAME dataset and applied all but the multi-token strings distance metrics. The results of the accuracy evaluation are given in Table 4. The first three columns give the accuracy figures, whereas the column labeled **AV** gives an average number of results returned in the answer set.

Metrics	AA	SR	RAA	AV
Bag Distance	0.476	0.841	0.876	3.02
Levenshtein	0.708	0.971	0.976	2.08
Smith-Waterman	0.625	0.763	0.786	3.47
Jaro	0.775	0.820	0.826	2.06
Jaro-Winkler	0.820	0.831	0.831	2.03
q-grams	0.714	0.974	0.981	2.09
pos q-grams	0.721	0.976	0.982	2.09
skip grams	0.873	0.935	0.936	2.14
LCS	0.696	0.971	0.977	12.69
WLCS	0.731	**0.983**	**0.986**	2.97
CP_{δ_1}	0.829	0.843	0.844	2.11
CP_{δ_2}	**0.947**	0.956	0.955	2.18

Table 4: Results for PL-F-NAMES

Interestingly, the simple linguistically-aware common prefix-based measure turned out to work best in the **AA** category, which is the most relevant one, whereas *WLCS* metrics is the most accurate in case of single-result answers and the **RAA** category. Thus, a combination of the two seems to be a reasonable solution to further improve the performance (i.e., if *WLCS* provides a single answer, return this answer, otherwise return the answer of CP_{δ_2}). Next, the time-efficient *skip grams* metrics performed surprisingly well in the **AA** category. This result was achieved with $\{0, 2\}$ gram classes. Recall that about 10% of the inflected first name forms in Polish are ambiguous, as they are either a male or a female person name, see sec. 2.

Clearly, the **AA** accuracy figures in the experiment run on the PL-F-NAME-2 (with a large search space) was significantly worse. However, the **SR** accuracy for some of the metrics is still acceptable. The top ranking metrics with respect to **SR** and **AA** accuracy are given in Table 5. Metrics which return more than 5 answers on average were excluded from this list. Also in the case of PL-F-NAME-2 the combination of $WLCS$ and CP_{δ_2} seems to be the best choice.

Metrics	SR	AA
WLCS	**0.893**	0.469
CP_{δ_2}	0.879	**0.855**
pos 2-grams	0.876	0.426
skip grams	0.822	0.567
2-grams	0.810	0.398
LCS	0.768	0.340
CP_{δ_1}	0.668	0.600
JW	0.620	0.560

Table 5: Top results for PL-F-NAMES-2

Finally, we have made experiments for full person names, each represented as two tokens. It is important to note that the order of the first name and the surname in some of the entities in our test datasets is swapped. This inaccuracy is introduced by full names where the surname may also function as a first name. Nevertheless, the results of the experiment on PL-FULL-NAMES given in Table 6 are nearly optimal. JWM, $WLCS$, LCS, skip grams and *Smith-Waterman* were among the 'best' metrics.

Internal Metrics	AA	SR	RAA	AV
Bag Distance	0.891	0.966	0.966	3.13
Smith-Waterman	0,965	0,980	0,975	3,5
Levenshtein	0.951	0.978	0.970	4.59
Jaro	0.957	0.970	0.964	3.54
JW	0.952	0.964	0.958	3.74
JWM	0.962	0.974	0.968	3.74
2-grams	0.957	0.988	0.987	3.915
pos 3-grams	0.941	0.974	0.966	4.32
skip-grams	0.973	0.991	0.990	5.14
LCS	0.971	0.992	0.990	5.7
WLCS	**0.975**	**0.993**	**0.992**	6.29

Table 6: Results for PL-FULL-NAMES

The *Monge-Elkan*, *Sorted-Tokens* and *Permuted-Tokens* scored in general only slightly better than the basic metrics. The best results oscillating around 0.97, 0.99, and 0.99 for the three accuracy metrics were obtained using *LCS*, *WLCS*, *JWM* and CP_δ metrics as internal metrics. The highest score was achieved by applying *Sorted-Tokens* with *JWM* with 0.976 in **AA** accuracy.

Further, in order to get a better picture, we have compared the performance of the aforementioned 'recursive' metrics on PL-FULL-NAMES-2, which has a larger search space. The most significant results for the **AA** accuracy are given in Table 7. The JWM metric seems to be the best choice as an internal metric, whereas $WLCS$, CP_{δ_2} and *Jaro* perform slightly worse.

Internal M.	Monge-Elkan	Sorted-Tokens	Permuted-Tokens
Bag Distance	0.868	0.745	0.745
Jaro	0.974	0.961	0.968
JWM	**0.976**	**0.976**	**0.975**
SmithWaterman	0.902	0.972	0.967
3-grams	0.848	0.930	0.911
pos 3-grams	0.855	0.928	0.913
skip-grams	0.951	0.967	0.961
LCS	0.941	0.960	0.951
WLCS	0.962	0.967	0.967
CP_{δ_1}	0.969	n.a.	n.a.
CP_{δ_2}	0.974	n.a.	n.a.

Table 7: **AA** accuracy for PL-FULL-NAMES-2

In our last experiment we selected the 'best' metrics so far and tested them against PL-FULL-NAMES-3 (largest search space). The top results for non-recursive metrics are given in Table 8. JWM and $WLCS$ turned out to achieve the best scores.

Metrics	AA	SR	RAA	AV
Levenshtein	0.791	0.896	0.897	2.20
Smith-Waterman	0.869	0.892	0.889	2.35
JW	0.791	0.807	0.802	2.11
JWM	**0.892**	0.900	0.901	2.11
skip-grams	0.852	0.906	0.912	2.04
LCS	0.827	0.925	0.930	2.48
WLCS	0.876	**0.955**	**0.958**	2.47

Table 8: Results for PL-FULL-NAMES-3

The top scores achieved for the recursive metrics on PL-FULL-NAMES-3 were somewhat better. In particular, *Monge-Elkan* performed best with CP_{δ_2} (0.937 **AA** and 0.946 **SR**) and slightly worse results were obtained with *JWM*. *Sorted-Tokens* scored best in **AA** and **SR** accuracy with *JWM* (0.904) and *WLCS* (0.949), respectively. Finally, for *Permuted-Tokens* the identical setting yielded the best results, namely 0.912 and 0.948, respectively.

5 Conclusions and Perspectives

For Slavic languages, rich and idiosyncratic inflection of proper names presents a serious problem for lemmatization. In this paper we investigated two different techniques for finding base forms of person names in Polish. The first one employs heuristics and linguistic knowledge. This method does not provide optimal results at the moment as necessary tools and linguistic resources, e.g., a morphological generator or a subcategorization lexicon, are still underdeveloped for Polish. Moreover, contextual heuristics do not always find a solution as the required information might not be present in a single document. Therefore, we considered string distance metrics as an alternative approach. The results of applying various measures indicate that for first names, simple common prefix (CP_δ) metric obtains the best results for all-answer accuracy, whereas the weighted longest common substrings ($WLCS$) measure provides the best score for the single-result accuracy. Hence, a combination of these two metrics seems the most appropriate knowledge-poor technique for lemmatizing Polish first names. As for full names, our two modifications ($WLCS$ and JWM) of standard distance metrics and CP_δ obtain good results as internal metrics for recursive measures and as stand-alone measures.

Although the results are encouraging, the presented work should not be considered a final solution. We plan to experiment with the best scoring metrics (e.g., for **AA** and **SR**) in order to find optimal figures. Additionally, we consider combining the two techniques. For example, string distance metrics can be used for validation of names found in the context. We also envisage applying the same methods to other types of proper names as well as to lemmatization of specialized terminology.

References

I. Bartolini, P. Ciacca, and M. Patella. 2002. String matching with metric trees using an approximate distance. In *Proceedings of SPIRE*, LNCS 2476, Lisbon, Portugal.

D. Bikel, S. Miller, R. Schwartz, and R. Weischedel. 1997. Nymble: A High-performance Learning Name-finder. In *Proceedings of ANLP-1997*, Washington DC, USA.

A. Borthwick. 1999. A Maximum Entropy Approach to Named Entity Recognition. PhD Thesis, Department of Computer Science, New York University.

P. Christen. 2006. A Comparison of Personal Name Matching: Techniques and Practical Issues. Technical report, TR-CS-06-02, Computer Science Laboratory, The Australian National University, Canberra, Australia.

W. Cohen, P. Ravikumar, and S. Fienberg. 2003. A comparison of string metrics for matching names and records. In *Proceedings of the KDD2003*.

H. Cunningham, E. Paskaleva, K. Bontcheva, and G. Angelova. 2003. Information extraction for Slavonic languages. In *Proceedings of the Workshop IESL*, Borovets, Bulgaria.

T. Erjavec and S. Džeroski. 2004. Machine Learning of Morphosyntactic Structure: Lemmatising Unknown Slovene Words. In *Journal of Applied Artificial Intelligence*, 18(1), pages 17-40.

L. Gravano, P. Ipeirotis, H. Jagadish, S. Koudas, N. Muthukrishnan, L. Pietarinen, and D. Srivastava. 2001. Using q-grams in a DBMS for Approximate String Processing. *IEEE Data Engineering Bulletin*, 24(4):28–34.

J. Grzenia. 1998. *Słownik nazw własnych — ortografia, wymowa, słowotwórstwo i odmiana*. PWN.

H. Keskustalo, A. Pirkola, K. Visala, E. Leppanen, and K. Jarvelin. 2003. Non-adjacent digrams improve matching of cross-lingual spelling variants. In *Proceedings of SPIRE*, LNCS 22857, Manaus, Brazil, pages 252–265.

V. Levenshtein. 1965. Binary Codes for Correcting Deletions, Insertions, and Reversals. *Doklady Akademii Nauk SSSR*, 163(4):845–848.

W. Li, R. Yangarber, and R. Grishman. 2003. Bootstrapping Learning of Semantic Classes from Positive and Negative Examples. In *Proceedings of the ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*.

A. Monge and C. Elkan. 1996. The Field Matching Problem: Algorithms and Applications. In *Proceedings of Knowledge Discovery and Data Mining 1996*, pages 267–270.

J. Piskorski, P. Homola, M. Marciniak, A. Mykowiecka, A. Przepiórkowski, and M. Woliński. 2004. Information Extraction for Polish Using the Sprout Platform. *Proceedings of ISMIS 2004, Zakopane*.

J. Piskorski. 2005. Named-entity Recognition for Polish with SProUT. In *Proceedings of IMTCI 2004*, LNCS Vol 3490, Warsaw, Poland.

J. Piskorski and M. Sydow. 2007. Usability of String Distance Metrics for Name Matching Tasks in Polish. In progress.

B. Pouliquen, R. Steinberger, C. Ignat, I. Temnikova, A. Widiger, W. Zaghouani and J. Žižka. 2005. Multilingual person name recognition and transliteration. *CORELA - Cognition, Représentation, Langage. Numéros spéciaux, Le traitement lexicographique des noms propres*, ISSN 1638-5748.

T. Smith and M. Waterman. 1981. Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, 147:195–197.

E. Ukkonen. 1992. Approximate String Matching with q-grams and Maximal Matches. *Theoretical Computer Science*, 92(1):191–211.

D. Weiss. 2007. Korpus Rzeczpospolitej. Web document: *http://www.cs.put.poznan.pl/dweiss/rzeczpospolita*

W. Winkler. 1999. The state of record linkage and current research problems. Technical report, U.S. Bureau of the Census, Washington, DC.