Building a morphosyntactic lexicon and a pre-syntactic processing chain for Polish

Benoît Sagot

IPI PAN (Warsaw) – Signes, INRIA Futurs (Bordeaux, France)

August 10, 2006

Context

The central goal of my work is to model and parse natural languages

- Developing formalisms (lexical, grammatical)
- Developing resources (lexicon, grammar,. . .)
- Developing tools (pre-syntactic tools, parser generators, automatic acquisition tools,. . .)

Until now, I mostly applied it to French, with some digressions on Slovak

Objectives

The aim of my stay here is to adapt and extend these tools for Polish:

- Morphological, and later syntactic lexicon (cf. the Lefff lexicon for French),
- Pre-parsing processing chain: SxPipe,
- Phrase-level LFG grammar and parser, which require the two others.

As for now, I worked mostly on the two first points.

1. Baseline pre-parsing processing: a Polish SxPipe

Representing lexical information

- An NLP lexicon has to represent several kinds of information: morphological, syntactic, and possibly semantic
  - We call **extensional lexicon** a resource that associates with each form a structure that represents all this information
    - Such a lexicon is typically used by parsers
  - We call **intensional lexicon** a resource that factorizes the information, by associating with each lemma a morphological class and a syntactic class
- We developed a formalism to describe **morphological classes** (conjugation and declension patterns,. . .)
- and a formalism to describe **syntactic classes** (inheritance graph of atomic syntactic properties)
- Both model derivational morphology

[Sagot *et al.*, 2006].

Representing lexical information: examples

pracować v-3alt @intransitive_verb	
młody adj @std_adj	*[base lemma]*
> nominalization -'eniec	*młodzeniec*
> nominalization -'ież	*młodzież*
>> adjectival derivative -owy	*młodzieżowy*
> abstract nominalization -ość	*młodość*

Representing lexical information: examples

pracujesz	fin:sg:sec:imperf	[pred='pracować$_1$<suj:sn>', cat=fin,@sg,@sec,@imperf]
pracowała	praet:sg:f:imperf	[pred='pracować$_1$<suj:sn>', cat=praet,@sg,@f,@imperf]
...		
młodej	adj:loc:f:pos	[pred='młody$_1$<suj:(sn)>',cat=adj,@loc,@f]
...		
bez mała	adv	[pred="bez mała$_1$", cat=adv]

The Lefff

Based on this architecture, we developed a **large-coverage morphological and syntactic lexicon for French**, the Lefff (Lexique des formes fléchies du français)

- 520,000 lexical entries...
- ...representing 420,000 wordforms
- used in all our tools (LFG and TAG parsers,...), as well as by other teams
- **freely available** (downloadable at `www.lefff.net`)
- ongoing collaborations to use a consensual representation model for syntactic information (with Claire Gardent, Laurence Danlos, Lionel Clément and others) and to compare this resources to the very few other resources for French (with Claire Gardent)

[Sagot *et al.*, 2005], [Sagot *et al.*, 2006], [Danlos and Sagot, 2006], [Danlos, Sagot and Salmon-Alt, 2006].

The Polish lexicon

- Our aim is to build a large lexicon for Polish that could be compared with the Lefff.
- The starting point is the very large high quality IPI PAN corpus
- Possible steps are:
  1. extraction of a morphological form-level lexicon directly from the corpus
  2. conversion into a lemma-level morphological lexicon + definition of morphological classes (inflection patterns)
  3. extension of this lexicon thanks to automatic extraction techniques and manual validation
  4. acquisition of syntactic information $\longrightarrow$ syntactic lexicon

Step 1 is done, step 2 in in progress, step 3 is at preliminary stages.

The Polish lexicon: direct extraction from the corpus

- It seems straightforward to extract a form-level lexicon from the corpus:

```
<tok>
<orth>Chcial</orth>
<lex disamb="1"><base>chcieć</base><ctag>praet:sg:m1:imperf</ctag></lex>
<lex><base>chcieć</base><ctag>praet:sg:m2:imperf</ctag></lex>
<lex><base>chcieć</base><ctag>praet:sg:m3:imperf</ctag></lex>
</tok>
```

- Two details needed to be fixed:
  - Identification of Proper Nouns (in the corpus, "Warszawa" has the lemma (lex→base) "warszawa")
  - Insertion of this automatic extraction process in the general lexicon architecture (to allow manually input lemmas as well)

Identifying proper nouns

- Little heuristics to identify proper nouns at a lemma level (cf. Łódź vs. łódź)
  - Any lemma which is not a `subst` or an `adj` is not a proper noun
  - Any lemma whose corresponding `orths` start with a capital letter more that 30% of all cases exists as a PN
  - Any lemma whose corresponding `orths` start with a capital letter more that 90% of all cases is only a PN
- Results are reasonable ; specific problems for words that are part of very frequent capitalized phrases (“Atlantycki”, “Demokratyczny”)

The result: a form-level Polish lexicon

- The result of this process is a form-level Polish lexicon
- It is a starting point both in terms of quality and coverage
  - Almost 550,000 forms
  - 30 categories (out of 32 existing ones. . .)
- We will discuss later about how to go beyond this baseline

What is pre-parsing processing?

- Current parsers, both shallow and deep, are able to deal with large corpus
- This makes it possible to achieve large-scale linguistic processing, such as acquisition, information extraction, and many others
- However, parsers often rely on lexicons and grammars designed to deal with "correct" language, which differs significantly from what can be found in real-life corpus
- Hence, there is an unavoidable need for pre-parsing processing methods to turn real-life corpus into less inacceptable inputs for sentence-level parsers
- If one wants to take into account most attested phenomena in a global way, this pre-parsing step is not as basic as it could seem, in particular because it has to be non-deterministic

SxPipe

We developped SxPipe, a set of tools (perl and C) which perform

- "named entities" recognition
  - pre-tokenization n.e. (URLs, emails, dates, addresses, numbers,...)
  - lexicon-aware n.e. (phrases in foreign languages,...)
  - multi-words n.e. (numbers in full text,...)
- segmentation in sentences
- tokenization and spelling error correction with SxSpeller
- non-deterministic "light" correction (re-accentuation, re-capitalization,...) and non-deterministic multi-words identification with text2dag

Until now, a large-scale SxPipe existed only for French [Sagot and Boullier, 2005a], [Sagot and Boullier, 2005b].

SxSpell

- SxSpeller and text2dag both rely on a form-level spelling correction module, named SxSpell
- Strong simplification of SxSpell's underlying algorithm:
  - The set of wordforms or forms that are in the lexicon are compiled into a huge automaton $\mathcal{L}$
  - For a given input string, if it is not recognized by $\mathcal{L}$, a set of possible corrections is computed by applying (weighted) correction rules until
  - All these possible corrections are compiled into another (a bit less) huge automaton $\mathcal{C}$
  - We compute the automaton of valid corrections by computing the intersection $\mathcal{L} \cap \mathcal{C}$
  - We output the best-weighted valid correction(s)

SxSpell

- Ontop of this module:
  - SxSpeller performs sophisticated heuristics to segment and/or re-glue tokens into forms
  - text2dag identifies in a non-deterministic way multi-token forms ("compound words")

  Of course, both tasks interact very strongly (and in a quite complicated way) with the spelling correction

A Polish version of SxPipe

Most of SxPipe modules depend, partly of completely, from the language

- Most "named entities" recognition tools had to be adapted, sometimes almost re-written, always extended (and corrected)
  - specific ways to say things (addresses, but also dates, times. . .)
- Other tools can be used or compiled directly as such, with the Polish
  - segmentation in sentences uses the Polish lexicon (for abbreviations ending with ".")
  - spelling correction rules are partly encoding-specific (re-accentuation,. . .) and partly language-specific ("ż" vs. "rz")
  - but then a simple re-compilation with the Polish lexicon as input leads to Polish versions of the tools
  - however, some packaging (=autotools) work required

Adaptation to the XCES format ; preliminary Slovak version as well

Using SxPipe to improve the corpus

- Of course, this Polish SxPipe is an important step before any parser
- But it can be used as well to improve the corpus:
  - By comparing SxPipe's segmentation into sentences and forms with the corpus' segmentation into sentences and tokens
  - Because it can give an interpretation for tokens that Morfeusz couldn't analyse, so-called **ign tokens**
    - in the "law" corpus (70 million tokens), 4% are ign tokens (3 million tokens)
    - uneven repartition

Tokens vs. forms in a corpus

- However, both techniques point out an important issue:
  - The corpus is a sequence of tokens, with which form-level interpretations are associated
  - SxPipe's output is a graph of forms, with which token-level anchors are associated, and with a definition of a "form" which is parsing-oriented
- Tokens and forms do not always correspond directly
  - {piątek 10.5.90 r.} _DATE
  - po prostu, bez mała,...
  - pomału, zawsze, niespełna, naprawdę
  - szliśmy, szliście
  - dwakroć

Most frequent problems solved by SxPipe

- The "special double-quote" tokenization-based errors: 5%
- "Named entities", esp.:
  - numbers: 43%
  - proper nouns (tokens starting with a capital letter): 14%
- Productive prefixes (wielko-, post-, agro-, anty-,...)
- Spelling errors
  - aberacji (aberracji), abmasadora (ambasadora), abowiem (albowiem), abp (aby), abrbitralności (arbitralności), absolutniej (absolutnej)...

  Still remains at this point: unknown words (and foreign words):
  - abolicjonistycznej, abonamencka, aborcyjna, abortera, absolutoryjny...

2. Towards a lemma-level lexicon for Polish

Why is lemma-level better?

A lemma-level lexicon is more appropriate than a form-level one for several reasons:

- Factorization of the information
- Benefit from the linguistic regularities in the morphology
- Encode and benefit (including at the syntactic level) from morphological derivation
- Automatic acquisition techniques to eliminate unknown words as much as possible
- Easier to maintain

But to go from a form-level to a lemma-level lexicon, we need a morphological description of Polish...

Morphological description

A morphological description of a language should have four main goals:

- Factoriziation of the information as much as possible
- Readability and maintainability
- Coverage and accuracy
- Can be used by a morphological compiler to generate automatically both
  - An inflection tool: lemma $\rightarrow$ forms
  - A (non-deterministic) lemmatization tool: form $\rightarrow$ lemmas

Morphological formalism

- We developed such a formalism and the associated morphological compiler (and applied it to French and Slovak)
- It relies on the following scheme:
  - A set of inflection classes, which can inherit (partly or completely) the one from the other
  - Each inflection class contains a set of forms represented as suffixes that have to be added to the stem
  - Forms can be controlled by tests over the stem
  - Forms can be selected by "variants" of the inflection classes
  - Collision patterns allow to link the real form to the sequence stem_suffix

[Sagot, 2005c]

Morphological formalism: examples

<letterclass name="**hard**" letters="**b p f w m n ł t d r s z ch h**"/>
...
<translitteration source=żz"target="ř"/> ...
<collision source="**r_**" target="**rz_**"/>
<collision source="**[:soft:]_y**" target="**[:soft:]_i**"/>
<collision source="**[:kg:]_e**" target="**[:kg:]_ie**" final="**+**"/>
...

Morphological formalism: examples

```
<table name="subst-m1" tag_suffix=":m1" stems="...*»
    <form suffix="" tag="sg:nom"/>
    <form like="sg:gen" tag="sg:acc"/>
    <form suffix="a" tag="sg:gen" except="(wol|bawol)"/>
    <form suffix="u" tag="sg:gen" stems="(wol|bawol)"/>
    <alt>
        <form suffix="owi" tag="sg:dat" var="Dowi"/>
        <form suffix="u" tag="sg:dat" var="Du"/>
    </alt>
    <form suffix="em" tag="sg:inst"/>
    <form suffix="'e" tag="sg:loc" stems="..*[:hard:]"
                                except="(syn|dom|pan)"/>
    <form suffix="u" tag="sg:loc" except="..*[:hard:]"/>
    <form suffix="u" tag="sg:loc" stems="(syn|dom|pan|bor)"/>
...
```

Morphological formalism: examples

```
<table name="adj" canonical_tag="sg:nom:m1" tag_suffix=":pos" stems="...*
    <form suffix="y" tag="sg:nom:m1"/>
    <form suffix="y" tag="sg:nom:m2"/>
    <form suffix="y" tag="sg:nom:m3"/>
    <form suffix="a" tag="sg:nom:f"/>
    <form suffix="e" tag="sg:nom:n"/>
...
    <form suffix="”i" tag="pl:nom:m1" except="..*[cczdzdżrz]"/>
    <form suffix="i" tag="pl:nom:m1" stems="..*[cczdzdżrz]"/>
...
    <derivation suffix="’ejszy" table="adj" except=".*(str|[ldkcz]n|pł|tw)"/>
    <derivation suffix="szy" table="adj" stems=".*(str|[ldkcz]n|pł|tw|dr|łt|st)"/
</table>
```

Morphological formalism: new features

- Apart from *heavy* improvements and corrections in the compiler, some features were added so as to model Polish in a satisfying way:
  - in-table form inheritance
  - alternances in collision patterns (e.g., [:soft:] / [:no acute:] : ć/c, ź/z, dź/dz, ś/s, ń/n)
  - `canonical_tag`: the lemmatized form is one of the inflected forms. This implies, for example when we want to inflect a lemma, to first "un-inflect" it to get the stem, and then inflect the stem
  - small practical extensions
- Maybe still required: a proper treatment of prefixes (e.g., naj-)

Towards a morphological description of Polish

At this point, only adjectives and common nouns are covered by the description

- 1 table for adjectives, + 2 other tables (comparatives and superlatives) that exactly inherit from the standard table and are here for technical reasons only (tag in `-comp` or `-sup` instead of `-pos`)

Towards a morphological description of Polish

- 10 tables for substantives:
  - table m1
  - table m1a of m1 substantives in -a
  - table m2 that inherits from m1 and redefines the nominative, vocative and accusative plural
  - table m3 that inherits from m2 and redefines the accusative and genitive singular
  - table n of standard neutral substantives
  - table num of neutral substantives in -um (inherits from table n, all singular forms in -um, gen. pl. in -ów)
  - tables nen and net respectively for types ramię/ramiona and cielę/cielęta
  - table fv for feminine substantives in -a or -i
  - table fc for feminine substantives with a zero ending at the nominative singular

Validation of the morphological model

- We do not want to lose information when going from the form-level lexicon to a lemma-level lexicon
- Hence, any form which is in the form-level lexicon must be analysable by the ambiguous lemmatizer
  - with the appropriate category and tag
  - with the appropriate lemma
- Maybe this will help to discover problems in Morfeusz
  - absolutu absolut sg:acc:m2 subst
  - absolutu absolut sg:gen:m2 subst
  - abonamentów abonamentowy sg:nom:m3:pos adj

Automatic acquisition of lexical information

We developed a technique to acquire automatically lexical information

- from a raw corpus and a morphological description of the language
- applied to
  - French verbs
  - all open categories of Slovak

[Clément, Sagot and Lang, 2004], [Sagot, 2005c]

Underlying idea

The underlying idea is the following:

- Firslty, we use the **ambiguous lemmatizer** generated from the morphological description: we build all hypothetical lemmas that have at least one inflected form attested in the corpus
- Then, we **inflect these lemmas** and **rank** them according to their likelihood given the corpus (fix-point algorithm)
- Many kinds of information are taken into account (derivational morphology, prefixes, frequency of tags depending on the category,. . .)
- **Manual validation** is performed on the best-ranked hypothetical lemmas
- The whole process is launched anew, and benefits from the manual validation step

Manual validation environment

byť	v-byť
európsky, a, e	adj-l
ten článok , bez článku , o článku (lok) , tie články , s/so článkami	nc-mid:Gu:0:Leu:ami
ten štát , bez štátu , o štáte (lok) , tie štáty, s/so štátmi	nc-mid:Gu:0:Leu:mi
ta únia, tie únie , sto únií, o úniách (lok)	nc-fiv
členský, á, é	adj-b
mať, mám, majú	v-ať
ta komisia, tie komisie , sto komisií, o komisiách (lok)	nc-fiv
to právo	nc-no
môcť, môžem, môžu, mohol	v-môcť
ta rada, tie rady	nc-fdv
ta oblasť , tie oblasti	nc-fm2
to opatrenie	nc-nie
ta návrha, tie návrhy	nc-fdv
ta krajina, tie krajiny	nc-fdv
spoločný, á, é	adj-b
to rámce	nc-ne
ta politika, tie politiky	nc-fdv
iný, á, é	adj-b
ten cieľ , tí cieli , s/so cieľmi	nc-mac:i:0:mi
ten rok , bez roka , o roku (lok) , tie roky, s/so rokmi	nc-mid:Ga:0:Leu:mi
ta voda, tie vody	nc-fdv

Extending the lexicon from ign tokens

- We would like to apply this technique to the remaining ign tokens
  - To extend the lexicon
  - To improve the corpus
- To reach high-quality results, two things are needed
  - A full-featured morphological description (incl. verbs)
  - A few adaptations so that the algorithm can
    - benefit from quantitative data about all words
    - still work only on ign tokens

Conclusions

The lexicon

Are still needed

- a more complete morphological description (verbs, more derivational morphology)
- to perform the from-level to lemma-level switch
- to acquire automatically new lexemes from ign tokens
- basic (default) syntactic information

Towards parsing

Because the aim of this work remains

- the development of a phrase-level LFG grammar and the associated parser (which will take as input the output of SxPipe)
- automatic acquisition of syntactic information from the output of this parser (sub-categorization frames,...)
- improvements of the parser thanks to these information
- towards a full-featured LFG (and parser) of Polish...