Multilingual Finite-State Noun Phrase Extraction

Anne Schiller
Rank Xerox Research Centre
38240 Meylan, France
Anne.Schiller@grenoble.rxrc.xerox.com

Abstract.
The paper describes a tool for noun phrase mark-up based on finite-state techniques and statistical part-of-speech disambiguation. We illustrate the proceeding by examples from realizations for seven languages (Dutch, English, French, German, Italian, Portuguese, and Spanish).

1 Introduction

For the purpose of terminology extraction from technical documents we designed a tool which applies finite-state techniques to mark potential terms, especially noun phrases corresponding to given regular patterns. The paper describes the general architecture of the tool and shows how finite-state transducers representing noun phrase patterns are used for noun phrase mark up for a range of languages.

The noun phrase extraction is the continuation of a chain of finite-state tools which include tokenizing, lexicon and guesser construction, and a statistical part-of-speech disambiguator (tagger) which uses a finite-state lexicon and guesser.

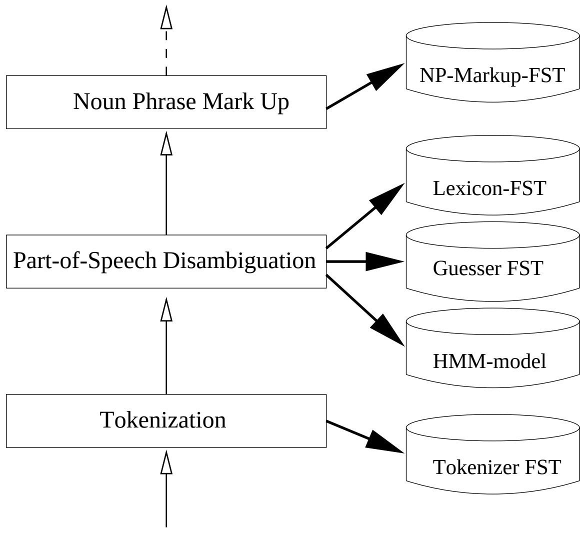

Figure 1. Architecture

2 Architecture

The noun-phrase extraction tool consists of several modules: language independent programs (tokenizer, part-of-speech disambiguator, and noun phrase mark-up) and language dependant data (finite-state transducers and transition probabilities). This modular architecture allows rapid extension to different languages.

Finite-state tools [7] and programming scripts serve to compute the finite-state transducers (FST). Manually tagged corpora and tagger training tools [8] provide transition probabilities for the statistical part-of-speech disambiguator.

Currently, implementations for 7 languages (Dutch, English, French, German, Italian, Portuguese, Spanish) exist; more languages (eg. Czech) are in preparation. We estimate that adding a new language involves about 1 man year of a lexicographer's time (including work on the morphological lexicon).

3 Tokenization

The tokenizing process devides a sequences of input characters into *tokens* which serve as input units for subsequent processing. In general, a token corresponds to an inflected word form, a number, a punctuation mark, etc.

The tokenizer program simply matches an input text with the lower side of the tokenizer transducer and returns the output text corresponding to the upper side. The program contains no general assumptions about character classes (word boundaries, brackets, punctuations etc.) or special tokens (such as abbreviations, numbers, SGML-markup). Thus, the tokenizer transducer, not the tokenizer program, defines the tokenizer output format.

The tokenizer transducers are language dependent. They handle abbreviations and multi-word expressions and allow white space insertion within special (contracted) word forms.

The construction of the tokenizer transducer employs the directed replace operator [4]. The resulting automaton is non-ambiguous, i.e. every sequence of input characters is mapped to one sequence of output tokens. Therefore, multi-word units are always tokenized as single tokens and are tagged as a unit in the subsequent processing.

© 1996 A. Schiller
Proceedings of the ECAI 96 Workshop
Extended Finite State Models of Language
Edited by A. Kornai.

Example: (French)

INPUT:

```
Vois-tu l'arbre à coté de la maison?
```

OUTPUT:

```
Vois-
-tu
l'
arbre
à coté de
la
maison
?
```

This short example shows how elided determiners ("l'") and clitics ("-tu") are tokenized in French. The sample input also contains a complex preposition ("á côté de") which is treated as multi-word unit.

4 Part-of-speech disambiguation

The tagging process consists in disambiguating the potential part-of-speech categories of a word form according to its context.

For that purpose we use a statistical disambiguator. The disambiguation program (a C-implementation of the Xerox tagger presented in [2]) applies the Viterbi algorithm. It relies on a transition model which results from a manually tagged training corpus and the Xerox authoring tools for Hidden Markov models [8]. The tagger requires a finite-state lexicon and guesser transducer to provide the input tags.

4.1 Tagger Lexicon

The tagger lexicon is a finite-state transducer which provides potential part-of-speech categories for every input word form. Basically, these categories correspond to those given by a morphological analyser. But in general, we have to reduce the rich set of morphological categories (eg. omitting tense information) or we have to introduce new distinctions (eg. auxiliary vs. function verb) for tagging purposes.

Finite-state lexicons for morphological analyzers [3] exist for several languages (eg. [5]). The tagger lexicons are derived from the morphological lexicons by means of finite-state mapping rules (cf. [1]). These rules are compiled into a mapping transducer which is composed with the morphological lexicon transducer to obtain the final tagger lexicon. Thus, the lexical information for an inflected word form depends on the basic morphological lexicon and on the mapping rules. The output side of the lexicon transducer must at least contain the part-of-speech tag which is used for disambiguation, but it may as well include the base form (lemma) or inflectional categories corresponding to the input token.

The following regular expression is an example for a mapping transducer which adds part-of-speech tags to the inflectional categories of a morphological lexicon. The first tagger lexicon (1) results from the composition of these rules with the morphological lexicon. The second tagger lexicon (2) omits base form and inflectional categories and provides only the part-of-speech tags.

Example: (Italian)

Mapping rules:

```
[
+Noun ?* +Sg ?* /NOUNSG:0 |
+Noun ?* +Pl ?* /NOUNPL:0 |
+Verb +Inf /VINF:0 |
+Verb +Inf ?* +Pron ?* /VINFCL:0 |
...
]
```

Morphological lexicon:

```
casa →
1. casa+Noun+Fem+Sg
case →
1. casa+Noun+Fem+Pl
dare →
1. dare+Verb+Inf
darmi →
1. dare+Verb+Inf|io+Pron+Acc+1P+Sg
2. dare+Verb+Inf|io+Pron+Dat+1P+Sg
```

Tagger lexicon (1):

```
casa →
1. casa+Noun+Fem+Sg/NOUNSG
case →
1. casa+Noun+Fem+Pl/NOUNPL
dare →
1. dare+Verb+Inf/VINF
darmi →
1. dare+Verb+Inf|io+Pron+Acc+1P+Sg/VINFCL
2. dare+Verb+Inf|io+Pron+Dat+1P+Sg/VINFCL
```

Tagger lexicon (2):

```
casa →     1. /NOUNSG
case →     1. /NOUNPL
dare →     1. /VINF
darle →    1. /VINFCL
```

4.2 Tagger Guesser

In addition to the lexicon, the tagging program applies a finite-state transducer for non-lexicalised word forms. The guesser transducer contains word form patterns to determine the potential part-of-speech categories of a given string.

These patterns rely basically on derivational affixes as shown in the following example for German:

Example: (German)

```
[
Up Low+ u n g (e n)                    NOUN:0 |
(Dig+) Low+ i g                        ADJP:0 |
(Dig+) Low+ i g e [m|n|s|r]            ADJA:0 |
...
]
```

- Consider strings starting with an uppercase character and ending with a suffix "ung" as nouns.
  Ex: "Aufwendungen", "FIBU-Abteilung"

- A string of lower case characters (possibly preceeded by digits) and a suffix "ig" is very likely to be an adjective: the corresponding tag is therefore either "ADJP" for a predicative (uninflected) adjective or "ADJA" for an attributive (inflected) adjective.
  Ex: "birnenförm**ig**", "27teil**ige**"

4.3 Tagger Model

The HMM-tagger applies a transition model which results from a manually tagged training corpus which was obtained by (iterative) automatic tagging and manual correction. The transition model is computed by an HMM-authoring tool [8].

The tagsets for the different languages cover the major part-of-speech classes (nouns, adjectives, verbs, pronouns, determiners etc.), but they differ with respect to subclassifications (number, gender, inflection, etc.) which are considered as language specific. The size of tagsets varies between 40 (for French) and 75 tags (for Italian).

The following example for Portuguese illustrates a tag classification which includes number information for nouns, adjectives, determiners and pronouns. This distinction does not appear in the German tagset as shown in the example below.

Example: (Portuguese)

```
INPUT:
Enriquecer as suas colecções através da aquisiçao
de obras de reconhecido valor bibliográfico e/ou
cultural.
```

```
OUTPUT:
Enriquecer/INF as/DETPL suas/POSSPL
colecções/NPL através da/PREPDETSG
aquisiçao/NSG de/PREP obras/NPL de/PREP
reconhecido/ADJSG valor/NSG
bibliográfico/ADJSG e/ou/CONJ cultural/ADJSG
./SENT
```

Example: (German)

```
INPUT:
Nun hofft man, auch über die Beziehungen und
Wechselwirkungen der ozeanischen Bewohner
untereinander und mit ihrer Umwelt, in der sich ein
Großteil der Evolution abgespielt hat, mehr zu
erfahren.
```

```
OUTPUT:
Nun/ADV hofft/VVFIN man/INDPRO ,/KOMMA
auch/ADV über/PREP die/ART
Beziehungen/NOUN und/COORD
Wechselwirkungen/NOUN der/ART
ozeanischen/ADJA Bewohner/NOUN
untereinander/ADV und/COORD mit/PREP
ihrer/POSDET Umwelt/NOUN ,/KOMMA
in/PREP der/RELPRO sich/REFLPRO ein/ART
Großteil/NOUN der/ART Evolution/NOUN
abgespielt/VVPP hat/VAFIN ,/KOMMA
mehr/ADV zu/PTKINF erfahren/VVINF ./SENT
```

The tagger program allows alternative output formats depending on the information coded in the tagger lexicon. The output may simply add part-of-speech tags to tokens (as in the example above), but it may also include base forms or morhological categories. The disambiguation program, however, considers only the part-of-speech tags. Thus, lexical information which is ambiguous with respect to a single tag, remains ambiguous in the tagger output. eg. including the lexical forms

Example: (German)

```
OUTPUT (including base forms):
 Nun     nun      ADV
 hofft   hoffen   VVFIN
 man     man      INDPRO
 ...
```

```
OUTPUT (including morphological categories):
 Nun     nun+Adv                      ADV
 hofft   hoffen+V+3P+Sg+Pres          VVFIN
         hoffen+V+2P+Pl+Pres          VVFIN
 man     man+Pron+Indef+Nom+Sg        INDPRO
 ...
```

5 Noun phrase markup

Noun phrase (NP) markup applies finite-state automata describing noun phrase patterns. These patterns rely on the simple (non-ambiguous) tagger output format, i.e. they consist of regular expressions on sequences of tokens and tags.

A very simple noun phrase description for a given language (eg. French) may consist in a (possibly empty) sequence of adjectives followed by a noun and another sequence of adjectives. Given part-of-speech tags for singular and plural nouns and singular and plural adjectives, the NP-automaton is defined as follows:

Example:

```
Tag = [ NOUNSG |NOUNPL |ADJSG |ADJPL ]
Char = ¬ [ Tag |Space ]
Word = Char [ Char |Space ]*

ASG = Word  "/"  ADJSG
APL = Word  "/"  ADJPL
NSG = Word  "/"  NOUNSG
NPL = Word  "/"  NOUNPL

NP =
[
[ ASG Space ]* NSG  [ Space ASG ]*  |
[ ASG Space ]* NSG  [ Space ASG ]*
]
```

The automata which describe noun phrases (as in the example above) are compiled into the final NP-markup. The compilation script uses the directed replace operation for the longest match and inserts brackets around maximal NPs (according to the NP patterns). The final NP-markup transducers are non-ambiguous, i.e. for every input they provide a single output containing non-recursive bracketing for NPs.

The NP-markup tool applies the NP-markup transducer (cf. tokenizer tool in section 3) to tagged text.

The follwing examples from the current realizations for French, Dutch and Spanish illustrate the application of the complete chain of tokenizing, part-of-speech disambiguation and noun phrase markup:

Example: (French)

INPUT:
Lorsqu'on tourne le commutateur de démarrage sur la position auxiliaire, l'aiguille retourne alors à zéro.

OUTPUT:
Lorsqu'/CONN on/PRON tourne/VERBP3SG le/DETSG [commutateur/NOUNSG de/PREPDE démarrage/NOUNSG]$_{NP}$ sur/PREP la/DETSG [position/NOUNSG auxiliaire/ADJSG]$_{NP}$,/CM l'/DETSG [aiguille/NOUNSG]$_{NP}$ retourne/VERBP3SG alors/ADV à/PREPA [zéro/NOUNSG]$_{NP}$ ./SENT

Example: (Dutch)

INPUT:
De reparatie- en afstelprocedures zijn bedoeld ter ondersteuning voor zowel de volledig gediplomeerde monteur als de monteur met minder ervaring.

OUTPUT:
De/ART [reparatie-/CMPDPART en/CON afstelprocedures/NOUN]$_{NP}$ zijn/VAFIN bedoeld/VVPP ter/PREP [ondersteuning/NOUN]$_{NP}$ voor/PREP zowel/CON de/ART [volledig/ADJA gediplomeerde/ADJA monteur/NOUN]$_{NP}$ als/PREP de/ART [monteur/NOUN]$_{NP}$ met/PREP minder/INDDET [ervaring/NOUN]$_{NP}$ ./SENT

Example: (Spanish)

INPUT:
Para asegurar el funcionamiento óptimo de los vehículos, así como la seguridad personal del técnico, es imprescindible seguir los métodos apropiados de trabajo y los procedimientos correctos de reparación.

OUTPUT:
Para/PREP asegurar/VINF el/DETSG [funcionamiento/NOUNSG óptimo/ADJSG de/PREP los/DETPL vehículos/NOUNPL]$_{NP}$,/COMA así como/CONJ la/DETSG [seguridad/NOUNSG personal/ADJSG del/PREPDET técnico/NOUNSG]$_{NP}$,/COMA es/AUX imprescindible/ADJSG seguir/VINF los/DETPL [métodos/NOUNPL apropiados/VPASTPARTPL de/PREP trabajo/NOUNSG]$_{NP}$ y/CONJ los/DETPL [procedimientos/NOUNPL correctos/ADJPL de/PREP reparación/NOUNSG]$_{NP}$ ./SENT

6 Evaluation

Each step in the processing (cf. figure 1) may be a source of errors. We are interested in the overall quality of the noun-phrase extraction tool, but also in the impact of errors in one process on the accuracy of subsequent processes.

Section 6.1 contains some statistics about the part-of-speech disambiguation of the seven languages. Work on evaluating the noun phrase markup has only started. Some preliminary results are shown in section 6.2.

6.1 Tagger Evaluation

During the tagger development for the given languages [6] the evaluatuion revealed a tagger accuracy between 97% and 98%. The table belows shows tagger error statistics for seven currently implemented languages:

- The first two columns concern the *lexical* properties of the disambiguation data: the number of tags defined in the tagset and the number of different ambiguity classes which appear in the lexicon.
- The last two columns contain statistics about the *test corpus*: the percentage of ambiguous word forms in the text and the error rate of the automatic tagging.

Language	Tags	Classes	Ambiguity	Errors
dutch	50	235	47 %	2.7 %
english	76	285	36 %	2.2 %
french	45	287	50 %	1.5 %
german	66	365	36 %	2.1 %
italian	74	444	27 %	3.1 %
portuguese	66	296	28 %	2.2 %
spanish	54	253	26 %	1.5 %

For the subsequent processing, ie. the noun phrase extraction, we are especially interested in tagger errors involving noun tags. Some results are shown in the table below.

- The first column shows the different noun tags which are defined for a given language.
- The second column (E1) contains the percentage of errors pertaining to noun tags, i.e. cases where the manually *or* the automatically attributed tag is in the class of noun categories.
- The last column (E2) contains the percentage of errors (E1) for cases where *both* the manual tag *and* the automatic tag are within the noun class.

Language	noun tags	E1	E2
dutch	NOUN	42 %	–
english	NN,NNS,NP,NPS	51 %	31 %
french	NSG,NPL,NINV	43 %	11 %
german	NOUN	20 %	–
italian	NSG,NPL,PROP	35 %	13 %
portuguese	NSG,NPL,NINV,PROP	44 %	22 %
spanish	NSG,NPL,NINV,PROP	61 %	19 %

6.2 Noun Phrase Evaluation

The current NP-markup was basically designed for terminology extraction from technical manuals. It covers "simple" noun phrase detection, i.e. constructions like coordination or relative clauses are not included.

The following table contains some statistics for the French and German noun phrase mark-up applied to a small test corpus of a technical manual. The size of the NP-transducers reflect the different structures of noun phrases in these languages: prepositional phrases ("pression d'huile") in French vs. compounds ("Öldruck") in German.

	French	German
NP-Transducer:		
states	24	54
arcs	325	1698
Sentences	145	146
marked NPs	355	361
single nouns	152	340
	43 %	94 %
missing NPs	2	2
partial NPs	6	–
wrong NPs	4	4

7 Conclusion

We presented a modular architecture of a tool for noun-phrase markup which is based on finite-state tools and an HMM-tagger. The overall architecture is language-independent and can be adapted for multiple languages or for different tagsets or noun-phrase definitions of one language. Moreover, the finite-state tools, which are used to compile and apply transducers, allow further cascading of transducers to extend the finite-state mark-up applications (eg. for verb phrase mark-up).

ACKNOWLEDGEMENTS

We would like to thank the linguists and lexicographers who worked on the tagger lexicons and guessers, on the tagger training and on the noun phrase descriptions for Dutch (D. van der Ende), French (J.-P. Chanod), Italian (A. Romano), Portuguese (C. Hagege), and Spanish (C. Arnaudo).

REFERENCES

[1] Jean-Pierre Chanod and Pasi Tapanainen, 'Creating a Tagset, Lexicon and Guesser for a French Tagger', in *Proceedings of the ACL SIGDAT Workshop*, Dublin, Ireland, (1995).

[2] Doug Cutting, Julian Kupiec, Jan Pedersen, and Penelope Sibun, 'A Practical Part-of-speech Tagger', in *Proceedings of ANLP-92*, Trento, Italy, (1992).

[3] Lauri Karttunen, 'Constructing Lexical Transducer', in *Proceedings of COLING-94*, Kyoto, Japan, (1994).

[4] Lauri Karttunen, 'Directed Replacement', in *Proceedings of ACL-96*, (1996). (to appear).

[5] Anne Schiller, 'DMOR: Benutzeranleitung', Internal report, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, (1995).

[6] Anne Schiller, 'Multilingual Tagger Development', Technical report, Rank Xerox Research Center, Meylan, France, (1996). (in preparation).

[7] Pasi Tapanainen, 'RXRC Finite-State Compiler', Technical Report MLTT-020, Rank Xerox Research Center, Meylan, France, (1995).

[8] Mike Wilkens and Julian Kupiec, 'Training Hidden Markov Models for Part-of-speech Tagging', Internal document, Xerox Corporation, (1995).