NATO Science for Peace and Security Series
D: Information and Communication Security - Vol. 21

Language Engineering for Lesser-Studied Languages

Edited by
Sergei Nirenburg

IOS Press

NATO OTAN

This publication is supported by:

The NATO Science for Peace and Security Programme

LANGUAGE ENGINEERING FOR LESSER-STUDIED LANGUAGES

NATO Science for Peace and Security Series

This Series presents the results of scientific meetings supported under the NATO Programme: Science for Peace and Security (SPS).

The NATO SPS Programme supports meetings in the following Key Priority areas: (1) Defence Against Terrorism; (2) Countering other Threats to Security and (3) NATO, Partner and Mediterranean Dialogue Country Priorities. The types of meeting supported are generally "Advanced Study Institutes" and "Advanced Research Workshops". The NATO SPS Series collects together the results of these meetings. The meetings are co-organized by scientists from NATO countries and scientists from NATO's "Partner" or "Mediterranean Dialogue" countries. The observations and recommendations made at the meetings, as well as the contents of the volumes in the Series, reflect those of participants and contributors only; they should not necessarily be regarded as reflecting NATO views or policy.

Advanced Study Institutes (ASI) are high-level tutorial courses to convey the latest developments in a subject to an advanced-level audience.

Advanced Research Workshops (ARW) are expert meetings where an intense but informal exchange of views at the frontiers of a subject aims at identifying directions for future action.

Following a transformation of the programme in 2006 the Series has been re-named and re-organised. Recent volumes on topics not related to security, which result from meetings supported under the programme earlier, may be found in the NATO Science Series.

The Series is published by IOS Press, Amsterdam, and Springer Science and Business Media, Dordrecht, in conjunction with the NATO Public Diplomacy Division.

Sub-Series

A.	Chemistry and Biology	Springer Science and Business Media
B.	Physics and Biophysics	Springer Science and Business Media
C.	Environmental Security	Springer Science and Business Media
D.	Information and Communication Security	IOS Press
E.	Human and Societal Dynamics	IOS Press

http://www.nato.int/science
http://www.springer.com
http://www.iospress.nl

Sub-Series D: Information and Communication Security – Vol. 21 ISSN 1874-6268

Language Engineering for Lesser-Studied Languages

Edited by

Sergei Nirenburg

University of Maryland Baltimore County, USA

IOS Press

Amsterdam • Berlin • Oxford • Tokyo • Washington, DC

Published in cooperation with NATO Public Diplomacy Division

Proceedings of the NATO Advanced Study Institute on Recent Advances in Language Engineering for Low- and Middle-Density Languages
Batumi, Georgia
15–27 October 2007

ISBN 978-1-58603-954-7
Library of Congress Control Number: 2008941928

Publisher
IOS Press
Nieuwe Hemweg 6B
1013 BG Amsterdam
Netherlands
fax: +31 20 687 0019
e-mail: order@iospress.nl

Distributor in the UK and Ireland
Gazelle Books Services Ltd.
White Cross Mills
Hightown
Lancaster LA1 4XS
United Kingdom
fax: +44 1524 63232
e-mail: sales@gazellebooks.co.uk

Distributor in the USA and Canada
IOS Press, Inc.
4502 Rachael Manor Drive
Fairfax, VA 22032
USA
fax: +1 703 323 3668
e-mail: iosbooks@iospress.com

PRINTED IN THE NETHERLANDS

Language Engineering for Lesser-Studied Languages
S. Nirenburg (Ed.)
IOS Press, 2009

Preface

Technologies enabling the computer processing of specific languages facilitate economic and political progress of societies where these languages are spoken. Development of methods and systems for language processing is, therefore, a worthy goal for national governments as well as for business entities and scientific and educational institutions in every country in the world. Significant progress has been made over the past 20–25 years in developing systems and resources for language processing. Traditionally, the lion's share of activity concentrated on the "major" languages of the world, defined not so much in terms of the number of speakers as with respect to the amount of publications of various kinds appearing in the language. Thus, much of the work in the field has been devoted to English, with Spanish, French, German, Japanese, Chinese and, to some extent, Arabic also claiming strong presence. The term "high-density" has been used to describe the above languages.

The rest of the languages of the world have fewer computational resources and systems available. As work on systems and resources for the "lower-density" languages becomes more widespread, an important question is how to leverage the results and experience accumulated by the field of computational linguistics for the major languages in the development of resources and systems for lower-density languages. This issue has been at the core of the NATO Advanced Studies Institute on language technologies for middle- and low-density languages held in Batumi, Georgia in October 2007. This book is a collection of publication-oriented versions of the lectures presented there.

The book is divided into three parts. The first part is devoted to the development of tools and resources for the computational study of lesser-studied languages. Typically, this is done on the basis of describing the work on creating an existing resource. Readers should find in this part's papers practical hints for streamlining the development of similar resources for the languages on which they are about to undertake comparable resource-oriented work. In particular, Dan Tufis describes an approach to test tokenization, part of speech tagging and morphological stemming as well as alignment for parallel corpora. Rodolfo Delmonte describes the process of creating a treebank of syntactically analyzed sentences for Italian. Marjorie McShane's chapter is devoted to the important issue of recognizing, translating and establishing co-reference of proper names in different languages. Ivan Derzhanski analyzes the issues related to the creation of multilingual dictionaries.

The second part of the book is devoted to levels of computational processing of text and a core application, machine translation. Kemal Oflazer describes the needs of and approaches to computational treatment of morphological phenomena in language. David Tugwell's contribution discusses issues related to syntactic parsing, especially parsing for languages that feature flexible word order. This topic is especially important for lesser-studied languages because much of the work on syntactic parsing has traditionally been carried out in languages with restricted word order, notably, English, while a much greater variety exists in the languages of the world. Sergei Nirenburg's section describes the acquisition of knowledge prerequisites for the analysis of meaning. The approach discussed is truly interlingual – it relies on an ontological metalanguage

for describing meaning that does not depend on any specific natural language. Issues of reusing existing ontological-semantic resources to speed up the acquisition of lexical semantics for lesser-studied languages are also discussed. Leo Wanner and François Lareau discuss the benefits of applying the meaning-text theory to creating text generation capabilities into multiple languages. Finally, Stella Makrantonatou and her co-authors Sokratis Sofianopoulos, Olga Giannoutsou and Marina Vassiliou describe an approach to building machine translation systems for lesser-studied languages.

The third and final part of the book contains three case studies on specific language groups and particular languages. Shuly Wintner surveys language resources for Semitic languages. Karine Megerdoomian analyzes specific challenges in processing Armenian and Persian and Oleg Kapanadze describes the results of projects devoted to applying two general computational semantic approaches – finite state techniques and ontological semantics – to Georgian.

The book is a useful source of knowledge about many core facets of modern computational-linguistic work. By the same token, it can serve as a reference source for people interested in learning about strategies that are best suited for developing computational-linguistic capabilities for lesser-studied languages – either “from scratch” or using components developed for other languages. The book should also be quite useful in teaching practical system- and resource-building topics in computational linguistics.

Contents

A. Tools and Resources

This page intentionally left blank

Language Engineering for Lesser-Studied Languages
S. Nirenburg (Ed.)
IOS Press, 2009

doi:10.3233/978-1-58603-954-7-3

Algorithms and Data Design Issues for Basic NLP Tools

Dan TUFIŞ
Research Institute for Artificial Intelligence of the Romanian Academy

Abstract. This chapter presents some of the basic language engineering pre-processing steps (tokenization, part-of-speech tagging, lemmatization, and sentence and word alignment). Tagging is among the most important processing steps and its accuracy significantly influences any further processing. Therefore, tagset design, validation and correction of training data and the various techniques for improving the tagging quality are discussed in detail. Since sentence and word alignment are prerequisite operations for exploiting parallel corpora for a multitude of purposes such as machine translation, bilingual lexicography, import annotation etc., these issues are also explored in detail.

Keywords. BLARK, training data, tokenization, tagging, lemmatization, aligning

Introduction

The global growth of internet use among various categories of users populated the cyberspace with multilingual data which the current technology is not quite prepared to deal with. Although it is relatively easy to select, for whatever processing purposes, only documents written in specific languages, this is by no means the modern approach to the multilingual nature of the ever more widespread e-content. On the contrary, there have been several international initiatives such as [1], [2], [3], [4] among many others, all over the world, towards an integrative vision, aiming at giving all language communities the opportunity to use their native language over electronic communication media. For the last two decades or so, multilingual research has been the prevalent preoccupation for all major actors in the multilingual and multicultural knowledge community. One of the fundamental principles of software engineering design, separating the data from the processes, has been broadly adhered to in language technology research and development, as a result of which numerous language processing techniques are, to a large extent, applicable to a large class of languages.

The success of data-driven and machine learning approaches to language modeling and processing as well as the availability of unprecedented volumes of data for more and more languages gave an impetus to multilingual research. It has been soon noticed that, for a number of useful applications for a new language, raw data was sufficient, but the quality of the results was significantly lower than for languages with longer NLP research history and better language resources. While it was clear from the very beginning that the quality and quantity of language specific resources were of crucial importance, with the launching of international multilingual projects, the issues of interchange and interoperability became research problems in themselves. Standards and recommendations for the development of language resources and associated

processing tools have been published. These best practice recommendations (e.g. Text Encoding Initiative (http://www.tei-c.org/index.xml), or some more restricted specifications, such as XML Corpus Encoding Standard (http://www.xml-ces.org/), Lexical Markup Framework (http://www.lexicalmarkupframework.org/) etc.) are language independent, abstracting away from the specifics, but offering means to make explicit any language-specific idiosyncrasy of interest.

It is worth mentioning that the standardization movement is not new in the Language Technology community, but only in recent years the recommendations produced by various expert bodies took into account a truly global view, trying to accommodate most of (ideally, all) natural languages and as many varieties of language data as possible. Each new language covered can in principle introduce previously overlooked phenomena, requiring revisions, extensions or even reformulations of the standards.

While there is an undisputed agreement about the role of language resources and the necessity to develop them according to international best practices in order to be able to reuse a wealth of publicly available methodologies and linguistic software, there is much less agreement on what would be the basic set of language resources and associated tools that is "necessary to do any pre-competitive research and education at all." [5]. A minimal set of such tools, known as BLARK (Basic LAnguage Resource Kit), has been investigated for several languages including Dutch [6], Swedish [7], Arabic [8], Welsh (and other Celtic languages) [9].

Although the BLARK concept does not make any commitment with respect to the symbolic-statistical processing dichotomy, in this paper, when not specified otherwise, we will assume a corpus-based (data-driven) development approach towards rapid prototyping of essential processing requirements for a new language.

In this chapter we will discuss the use of the following components of BLARK for a new language:

- (for monolingual processing) tokenization, morpho-lexical tagging and lemmatization; we will dwell on designing tagsets and building and cleaning up the training data required by machine learning algorithms;
- (for multilingual processing) sentence alignment and word alignment of a parallel corpus.

1. Tokenization

The first task in processing written natural language texts is breaking the texts into processing units called tokens. The program that performs this task is called segmenter or tokenizer. Tokenization can be done at various granularity levels: a text can be split into paragraphs, sentences, words, syllables or morphemes and there are already various tools available for the job. A sentence tokenizer must be able to recognize sentence boundaries, words, dates, numbers and various fixed phrases, to split clitics or contractions etc. The complexity of this task varies among the different language families. For instance in Asian languages, where there is no explicit word delimiter (such as the white space in the Indo-European languages), automatically solving this problem has been and continues to be the focus of considerable research efforts. According to [10], for Chinese "sentence tokenization is still an unsolved problem". For most of the languages using the space as a word delimiter, the tokenization process

was wrongly considered, for a long time, a very simple task. Even if in these languages a string of characters delimited by spaces and/or punctuation marks is most of the time a proper lexical item, this is not always true. The examples at hand come from the agglutinative languages or languages with a frequent and productive compounding morphology (consider the most-cited *Lebensversicherungsgesellschaftsangestellter,* the German compound which stands for "life insurance company employee"). The non-agglutinative languages with a limited compounding morphology frequently rely on analytical means (multiword expressions) to construct a lexical item. For translation purposes considering multiword expressions as single lexical units is a frequent processing option because of the differences that might appear in cross-lingual realization of common concepts. One language might use concatenation (with or without a hyphen at the joint point), agglutination, derivational constructions or a simple word. Another language might use a multiword expression (with compositional or non-compositional meaning). For instance the English *in spite of*, *machine gun*, *chestnut tree*, *take off* etc. or the Romanian *de la* (from), *gaura cheii* (keyhole), *sta în picioare* (to stand), *(a)-şi aminti* (to remember), etc. could be arguably considered as single meaningful lexical units even if one is not concerned with translation. Moreover, cliticized word forms such as the Italian *damelo* or the Romanian *dă-mi-le* (both meaning "give them to me"), need to be recognized and treated as multiple lexical tokens (in the examples, the lexical items have distinct syntactic functions: predicate (*da/dă*), indirect object (*me/mi*) and direct object (*lo/le*).

The simplest method for *multiword expression* (MWE) recognition during text segmentation is based on (monolingual) lists of most frequent compound expressions (collocations, compound nouns, phrasal verbs, idioms, etc) and some regular expression patterns for dealing with multiple instantiations of similar constructions (numbers, dates, abbreviations, etc). This linguistic knowledge (which could be compiled as a finite state transducer) is referred to as tokenizer's MWE resources. In this approach the tokenizer would check if the input text contains string sequences that match any of the stored patterns and, in such a case, the matching input sequences are replaced as prescribed by the tokenizer's resources. The main criticism of this simple text segmentation method is that the tokenizer's resources are never exhaustive. Against this drawback one can use special programs for automatic updating of the tokenizer's resources using collocation extractors. A statistical collocation extraction program is based on the insight that words that appear together more often than would be expected under an independence assumption and conform to some prescribed syntactic patterns are likely to be collocations. For checking the independence assumption, one can use various statistical tests such as mutual information, DICE, log-likelihood, chi-square or left-Fisher exact test (see, for instance, http://www.d.umn.edu/~tpederse/code.html). As these tests are considering only pairs of tokens, in order to identify collocations longer than two words, bigram analysis must be recursively applied until no new collocations are discovered. The final list of extracted collocations must be filtered out as it might include many spurious associations.

For our research we initially used Philippe di Cristo's multilingual segmenter MtSeg (http://www.lpl.univ-aix.fr/projects/multext/MtSeg/) built in the MULTEXT project. The segmenter comes with tokenization resources for many Western European languages, further enhanced, as a result of the MULTEXT-EAST project, with corresponding resources for Bulgarian, Czech, Estonian, Hungarian, Romanian and Slovene. MtSeg is a regular expression interpreter whose performance depends on the

coverage of available tokenization resources. Its main advantage is that for tokens with the same cross-lingual interpretation (numbers dates, clitics, compounds, abbreviations etc) the same label will be assigned, irrespective of the language. We re-implemented MtSeg in an integrated tokenization-tagging and lemmatization web service called TTL [11], available at http://nlp.racai.ro, for processing Romanian and English texts.

For updating the multiword expressions resource file of the tokenizer, we developed a statistical collocation extractor [12] which is not constrained by token adjacency and thus can detect token combinations which are not contiguous. The criteria for considering a pair of tokens as a possible interesting combination are:

- stability of the distance between the two lexical tokens within texts (estimated by a low standard deviation of these distances)
- statistical significance of co-occurrence for the two tokens (estimated by a log-likelihood test).

The set of automatically extracted collocations are hand-validated and added to the multiword expressions resource file of the tokenizer.

2. Morpho-lexical Disambiguation

Morpho-lexical ambiguity resolution is a key task in natural language processing [13]. It can be regarded as a classification problem: an ambiguous lexical item is one that in different contexts can be classified differently and given a specified context the disambiguation/classification engine decides on the appropriate class.

Any classification process requires a set of distinguishing features of the objects to be classified, based on which a classifier could make informed decisions. If the values of these features are known, then the classification process is simply an assignment problem. However, when one or more values of the classification criteria are unknown, the classifier has to resort to other information sources or to make guesses. In a well-defined classification problem each relevant feature of an entity subject to classification (here, lexical tokens) has a limited range of values.

The decisions such as what is a lexical token, what are the relevant features and values in describing the tokens of a given language, and so on, depend on the circumstances of an instance of linguistic modeling (what the modeling is meant for, available resources, level of knowledge and many others). Modeling language is not a straightforward process and any choices made are a corollary of a particular view of the language. Under different circumstances, the same language will be more often than not modeled differently. Therefore, when speaking of a natural language from a theoretical-linguistics or computational point of view, one has to bear in mind this distinction between language and its modeling. Obviously this is the case here, but for the sake of brevity we will use the term *language* even when an accurate reference would be *(X's) model of the language*.

The features that are used for the classification task are encoded in tags. We should observe that not all lexical features are equally good predictors for the correct contextual morpho-lexical classification of the words. It is part of the corpus linguistics lore that in order to get high accuracy level in statistical part-of-speech disambiguation, one needs small tagsets and reasonably large training data.

Earlier, we mentioned several initiatives towards the standardization of morpho-lexical descriptions. They refer to a neutral, context independent and maximally informative description of the available lexical data. Such descriptions in the context of the Multext-East specifications are represented by what has been called *lexical tags*. *Lexical tagsets* are large, ranging from several hundreds to several thousands of tags. Depending on specific applications, one can define subsets of tagsets, retaining in these reduced tagsets only features and values of interest for intended applications. Yet, given that the statistical *part of speech (POS*) *tagging* is a distributional method, it is very important that the features and values preserved in a tagset be sensitive to the context and to the distributional analysis methods. Such reduced tagsets are usually called *corpus tagsets.*

The effect of tagset size on tagger performance has been discussed in [14] and several papers in [13] (the reference tagging monograph). If the underlying language model uses only a few linguistic features and each of them has a small number of attributes, than the cardinality of the necessary tagset will be small. In contrast, if a language model uses a large number of linguistic features and they are described in terms of a larger set of attributes, the necessary tagset will be necessarily larger than in the previous case. POS-tagging with a large tagset is harder because the granularity of the language model is finer-grain. Harder here means slower, usually less accurate and requiring more computational resources. However, as we will show, the main reason for errors in tagging is not the number of feature-values used in the tagset but the adequacy of selected features and of their respective values. We will argue that a carefully designed tagset can assure an acceptable accuracy even with a simple-minded tagging engine, while a badly designed tagset could hamper the performance of any tagging program.

It is generally believed that the state of the art in POS tagging still leaves room for significant improvements as far as correctness is concerned. In statistics-based tagging, besides the adequacy of the tagset, there is another crucial factor[1], the quantity and quality of *the training data* (evidence to be generalized into a language model). A training corpus of anywhere from 100,000 up to over a million words is typically considered adequate. Although some taggers are advertised as being able to learn a language model from raw texts and a word-form lexicon, they require post-validation of the output and a bootstrapping procedure that would take several iterations to bring the tagger's error rate to an acceptable level.

Most of the work in POS-tagging relies on the availability of high-quality training data and concentrates on the engineering issues to improve the performance of learners and taggers [13-25]. Building a high-quality training corpus is a huge enterprise because it is typically hand-made and therefore extremely expensive and slow to produce. A frequent claim justifying poor performance or incomplete evaluation for POS taggers is the dearth of training data. In spite of this, it is surprising how little effort has been made towards automating the tedious and very expensive hand-annotation procedures underlying the construction or extension of a training corpus. The utility of a training corpus is a function not only of its correctness, but also of its size and diversity. Splitting a large training corpus into register-specific components

[1] We don't discuss here the training and the tagging engines, which are language-independent and obviously play a fundamental role in the process.

can be an effective strategy towards building a highly accurate combined language model, as we will show in Section 2.5.

2.1. Tagsets encoding

For computational reasons, it is useful to adopt an encoding convention for both lexical and corpus tagsets. We briefly present the encoding conventions used in the Multext-East lexical specifications (for a detailed presentation, the interested reader should consult the documentation available at http://nl.ijs.si/ME/V3/msd/).

The morpho-lexical descriptions, referred to as *MSD*s, are provided as strings, using a linear encoding. In this notation, the position in a string of characters corresponds to an attribute, and specific characters in each position indicate the value for the corresponding attribute. That is, the positions in a string of characters are numbered 0, 1, 2, etc., and are used in the following way (see Table 1):

- the character at position 0 encodes part-of-speech;
- each character at position 1, 2,...,n, encodes the value of one attribute (person, gender, number, etc.), using the one-character code;
- if an attribute does not apply, the corresponding position in the string contains the special marker '-' (hyphen).

Table 1. The Multilingual Multext-East Description Table for the Verb

Position	Attribute	Value	Code
0	POS	verb	V
1	Type	main	m
		auxiliary	a
		modal	o
		copula	c
	l.s.	base	b
2	Vform	indicative	i
		subjunctive	s
		imperative	m
		conditional	c
		infinitive	n
		participle	p
		gerund	g
		supine	u
	l.s.	transgress	t
	l.s.	quotative	q
3	Tense	present	p
		imperfect	i
		future	f
		past	s
	l.s.	pluperfect	l
	l.s.	aorist	a
4	Person	first	1
		second	2
		third	3
5	Number	singular	s
		plural	p
	l.s.	dual	d
6	Gender	masculine	m
		feminine	f
		neuter	n
7	Voice	active	a
		passive	p
8	Negative	no	n
		yes	y
9	Definite	no	n
		yes	y
	l.s.	short_art	s
	l.s.	ful_art	f
	l.s.	1s2s	2
10	Clitic	no	n
		yes	y
11	Case	nominative	n
		genitive	g
		dative	d
		accusative	a
		locative	l
		instrumental	i
		illative	x
		inessive	2
		elative	e
		translative	4
		abessive	5
12	Animate	no	n
		yes	y
13	Clitic_s	no	n
		yes	y

The "does not apply" marker ('-') in the MSD encoding must be explained. Besides the basic meaning that the attribute is not valid for the language in question, it

also indicates that a certain combination of other morpho-lexical attributes makes the current one irrelevant. For instance, non-finite verbal forms are not specified for Person.

The EAGLES recommendations (http://www.ilc.cnr.it/EAGLES96/morphsyn/morphsyn.html) provide another special attribute value, the dot ("."), for cases where an attribute can take any value in its domain. The 'any' value is especially relevant in situations where word-forms are underspecified for certain attributes but can be recovered from the immediate context (by grammatical rules such as agreement). By convention, trailing hyphens are not included in the MSDs. Such specifications provide a simple and relatively compact encoding, and are in intention similar to feature-structure encoding used in unification-based grammar formalisms.

As can be seen from Table 1, the MSD **Vmmp2s, will** be unambiguously interpreted as a Verb+Main+Imperative+Present+Second Person+Singular for any language.

In many languages, especially those with a productive inflectional morphology, the word-form is strongly marked for various feature-values, so one may take advantage of this observation in designing the reduced corpus tagset. We will call the tags in a reduced corpus tagset *c-tag*s. For instance, in Romanian, the suffix of a finite verb together with the information on person, almost always determine all the other feature values relevant for describing an occurrence of a main verb form. When this dependency is taken into account, almost all of the large number of Romanian verbal MSDs will be filtered out, leaving us with just three MSDs: Vm--1, Vm--2 and Vm—3, each of them subsuming several MSDs, as in the example below:

```
Vm--2 ⇒ {Vmii2s----y Vmip2p Vmip2s Vmsp2s----y Vmip2p----y Vmm-2p Vmm-2s
         Vmil2p----y Vmis2s----y Vmis2p Vmis2s Vmm-2p----y Vmii2p----y
         Vmip2s----y Vmsp2p----y Vmii2p Vmii2s Vmil2s----y Vmis2p----y
         Vmil2p Vmil2s Vmm-2s----y Vmsp2p Vmsp2s}
```

The set of MSDs subsumed by a c-tag is called its *MSD-coverage* denoted by `msd_cov(c-tag)`. Similar correspondences can be defined for any c-tag in the reduced corpus tagset. The set of these correspondences defines the mapping M between a corpus tagset and a lexical tagset. For reasons that will be discussed in the next section, a proper mapping between a lexical tagset and a corpus tagset should have the following properties:

- the set of *MSD-coverage*s for all c-tags represents a partition of MSD tagset
- for any MSD in the lexical tagset there exists a unique c-tag in the corpus tagset.

By definition, for any MSD there exists a unique c-tag that observes the properties above and for any c-tag there exists a unique MSD-coverage. The mapping M represents the essence of our tiered-tagging methodology.

As we will show, given a lexical tagset one could automatically build a corpus tagset and a mapping M between the two tagsets. If a training corpus is available and disambiguated in terms of lexical tags, the tiered tagging design methodology may generate various corpus tagsets, optimized according to different criteria. The discussion that follows concentrates on Romanian but similar issues arise and must be resolved when dealing with other languages.

2.2. The Lexical Tagset Design: A Case Study on Romanian

An EAGLES-compliant MSD word-form lexicon was built within the MULTEXT-EAST joint project within the Copernicus Program. A lexicon entry has the following structure:

`word-form` <TAB> `lemma` <TAB> `MSD`

where `word-form` represents an inflected form of the `lemma`, characterized by a combination of feature values encoded by `MSD` code. According to this representation, a word-form may appear in several entries, but with different MSDs or different lemmas. The set of MSDs with which a word-form occurs in the lexicon represents its *ambiguity class*. As an ambiguity class is common to many word-forms, another way of saying that the ambiguity class of word w_k is A_m, is to say that (from the ambiguity resolution point of view) the word w_k belongs to the ambiguity class A_m.

When the word-form is identical to the lemma, then an equal sign is written in the lemma field of the entry ('='). The attributes and most of the values of the attributes were chosen considering only word-level encoding. As a result, values involving compounding, such as compound tenses, though familiar from grammar textbooks, were not chosen for the MULTEXT-EAST encoding.

The initial specifications of the Romanian lexical tagset [26] took into account all the morpho-lexical features used by the traditional lexicography. However, during the development phase, we decided to exploit some regular syncretic features (gender and case) which eliminated a lot of representation redundancy and proved to be highly beneficial for the statistics-based tagging. We decided to use two special cases (*direct* and *oblique*) to deal with the nominative-accusative and genitive-dative syncretism, and to eliminate neuter gender from the lexicon encoding. Another feature which we discarded was *animacy* which is required for the vocative case. However, as vocative case has a distinctive inflectional suffix (also, in normative writing, an exclamation point is required after a vocative), and given that metaphoric vocatives are very frequent (not only in poetic or literary texts), we found the *animacy* feature a source of statistical noise (there are no distributional differences between animate and inanimate noun phrases) and, therefore, we ignored it.

With redundancy eliminated, the word-form lexicon size decreased more than fourfold. Similarly the size of the lexical tagset decreased by more than a half. While any shallow parser can usually make the finer-grained case distinction and needs no further comment, eliminating neuter gender from the lexicon encoding requires explanation. Romanian grammar books traditionally distinguish three genders: masculine, feminine and neuter. However there are few reasons – if any – to retain the neuter gender and not use a simpler dual gender system. From the inflectional point of view, neuter nouns/adjectives behave in singular as masculine nouns/adjectives and in plural as feminine ones. Since there is no intrinsic semantic feature specific to neuter nouns (inanimacy is by no means specific to neuter nouns; plenty of feminine and masculine nouns denote inanimate things) preserving the three-valued gender distinction creates more problems than it solves. At the lookup level, considering only gender, any adjective would be two-way ambiguous (masculine/neuter in singular and feminine/neuter in plural). However, it is worth mentioning that if needed, the neuter nouns or adjectives can be easily identified: those nouns/adjectives that are tagged with masculine gender in singular and with feminine gender in plural are what the traditional

Romanian linguistics calls neuter nouns/adjectives. This position has recently found adherents among theoretical linguists as well. For instance, in [27] neuter nouns are considered to be underspecified for gender in their lexical entries, having default rules assigning masculine gender for occurrences in singular and feminine gender for occurrences in plural.

For the description of the current Romanian word-form lexicon (more then one million word-forms, distributed among 869 ambiguity classes) the lexical tagset uses 614 MSD codes. This tagset is still too large because it requires very large training corpora for overcoming data sparseness. The need to overcome data sparseness stems from the necessity to ensure that all the relevant sequences of tags are seen a reasonable number of times, thus allowing the learning algorithms to estimate (as reliably as possible) word distributions and build robust language models. Fallback solutions for dealing with unseen events are approximations that significantly weaken the robustness of a language model and affect prediction accuracy. For instance in a trigram-based language model, an upper limit of the search space for the language model would be proportional to N^3 with N denoting the cardinality of the tagset. Manually annotating a corpus containing (at least several occurrences of) all the legal trigrams using a tagset larger than a few hundreds of tags is practically impossible.

In order to cope with the inherent problems raised by large tagsets one possible solution is to apply a tiered tagging methodology.

2.3. Corpus Tagset Design and Tiered Tagging

Tiered tagging (TT) is a very effective technique [28] which allows accurate morpho-lexical tagging with large lexicon tagsets and requires reasonable amounts of training data. The basic idea is using a hidden tagset, for which training data is sufficient, for tagging proper and including a post-processing phase for transforming the tags from the hidden tagset into the more informative tags from the lexicon tagset. As a result, for a small price in tagging accuracy (as compared to the direct reduced tagset approach), and with practically no changes to computational resources, it is possible to tag a text with a large tagset by using language models built for reduced tagsets. Consequently, for building high quality language models, training corpora of moderate size would suffice.

In most cases, the word-form and the associated MSD taken together contain redundant information. This means that the word-form and several attribute-value pairs from the corresponding MSD (called *the determinant* in our approach) uniquely determine the rest of the attribute-value pairs (*the dependent*). By dropping the dependent attributes, provided this does not reduce the cardinality of *ambiguity classes* (see [28]), several initial tags are merged into fewer and more general tags. This way the cardinality of the tagset is reduced. As a result, the tagging accuracy improves even with limited training data. Since the attributes and their values depend on the grammar category of the word-forms we will have different determinants and dependents for each part of speech. Attributes such as part of speech (the attribute at position 0 in the MSD encoding) and *orth*, whose value is the given word form, are included in every determinant. Unfortunately, there is no unique solution for finding the rest of the attributes in the determinants of an MSD encoding. One can identify the smallest set of determinant attributes for each part of speech but using the smallest determinant (and implicitly the smallest corpus tagset) does not necessarily ensure the best tagging accuracy.