**INTERNATIONAL WORKSHOP**

# ADAPTATION OF LANGUAGE RESOURCES AND TECHNOLOGY TO NEW DOMAINS (AdaptLRTtoND)

*held in conjunction with the International Conference*
*RANLP - 2009, 14-16 September 2009, Borovets, Bulgaria*

# PROCEEDINGS

Edited by
Núria Bel, Erhard Hinrichs, Kiril Simov and Petya Osenova

The Workshop is endorsed by FLaReNet Project


http://www.flarenet.eu/
The Workshop is related to CLARIN Project
http://www.clarin.eu/

Borovets, Bulgaria
17 September 2009

**International Workshop**

**ADAPTATION OF LANGUAGE RESOURCES AND TECHNOLOGY TO NEW DOMAINS**

# PROCEEDINGS

Borovets, Bulgaria
17 September 2009

ISBN 978-954-452-009-0

Designed and Printed by INCOMA Ltd.
Shoumen, Bulgaria

## Workshop Organisers

**Núria Bel**
Pompeu Fabra University, Spain
**Erhard Hinrichs**
Tuebingen University, Germany (co-chair)
**Petya Osenova**
Bulgarian Academy of Sciences and Sofia University, Bulgaria
**Kiril Simov**
Bulgarian Academy of Sciences, Bulgaria (co-chair)

## Workshop Programme Committee

# Table of Contents

# Conference Program

Session 1

09:20-09:30 Opening

09:30-10:30 TBA
Atanas Kiryakov (Invited Speaker), OntoText AD, Sofia, Bulgaria

10:30-11:00 *Enabling Adaptation of Lexicalised Grammars to New Domains*
Valia Kordoni and Yi Zhang

Session 2

11:30-12:00 *Personal Health Information Leak Prevention in Heterogeneous Texts*
Marina Sokolova, Khaled El Emam, Sean Rose, Sadrul Chowdhury, Emilio Neri, Elizabeth Jonker and Liam Peyton

12:00-12:30 *Mix and Match Replacement Rules*
Dale Gerdemann

Session 3

14:00-14:30 *Cross-lingual Adaptation as a Baseline: Adapting Maximum Entropy Models to Bulgarian*
Georgi Georgiev, Preslav Nakov, Petya Osenova and Kiril Simov

14:30-15:00 *Exploiting the Russian National Corpus in the Development of a Russian Resource Grammar*
Tania Avgustinova and Yi Zhang

15:00-15:30 *LEXIE - an Experiment in Lexical Information Extraction*
John J. Camilleri and Michael Rosner

16:00-16:30 *Maximal Phrases Based Analysis for Prototyping Online Discussion Forums Postings*
Gaston Burek and Dale Gerdemann

16:30-17:00 *QALL-ME needs AIR: a portability study*
Constantin Orăsan, Iustin Dornescu and Natalia Ponomareva

17:00-17:10 Closing Remarks

# Exploiting the Russian National Corpus in the Development of a Russian Resource Grammar

Tania Avgustinova
DFKI GmbH & Saarland University
P.O. Box 151150
66041 Saarbrücken, Germany
avgustinova@dfki.de

Yi Zhang
DFKI GmbH & Saarland University
P.O. Box 151150
66041 Saarbrücken, Germany
yzhang@coli.uni-sb.de

## Abstract

In this paper we present the on-going grammar engineering project in our group for developing in parallel resource precision grammars for Slavic languages. The project utilizes DELPH-IN software (LKB/[incr tsdb()]) as the grammar development platform, and has strong affinity to the LinGO Grammar Matrix project. It is innovative in that we focus on a closed set of related but extremely diverse languages. The goal is to encode mutually interoperable analyses of a wide variety of linguistic phenomena, taking into account eminent typological commonalities and systematic differences. As one major objective of the project, we aim to develop a core Slavic grammar whose components can be commonly shared among the set of languages, and facilitate new grammar development. As a showcase, we discuss a small HPSG grammar for Russian. The interesting bit of this grammar is that the development is assisted by interfacing with existing corpora and processing tools for the language, which saves significant amount of engineering effort.

## Keywords

Parallel grammar engineering, corpora, Slavic languages

## 1. Introduction

Our long-term goal is to develop grammatical resources for Slavic languages and to make them freely available for the purposes of research, teaching and natural language applications. As one major objective of the project, we aim to develop and implement a core Slavic grammar whose components can be commonly shared among the set of languages, and facilitate new grammar development. A decision on the proper set up along with a commitment to a reliable infrastructure right from the beginning are essential for such an endeavor because the implementation of linguistically-informed grammars for natural languages draws on a combination of engineering skills, sound grammatical theory, and software development tools.

### 1.1 DELPH-IN initiative

Current international collaborative efforts on deep linguistic processing with Head-driven Phrase Structure Grammar [1-3] exploit the notion of shared grammar for the rapid development of grammars for new languages and for the systematic adaptation of grammars to variants of the same language. This international partnership, which became popular under the name DELPH-IN[1], is based on a shared commitment to re-usable, multi-purpose resources and active exchange. Its leading idea is to combine linguistic and statistical processing methods for getting at the meaning of texts and utterances. Based on contributions from several member institutions and joint development over many years, an open-source repository of software and linguistic resources has been created that already enjoys wide usage in education, research, and application building.

In accord with the DELPH-IN community we view rule-based precision grammars as linguistically-motivated resources designed to model human languages as accurately as possible. Unlike statistical grammars, these systems are hand-built by grammar engineers, taking into account the engineer's theory and analysis for how to best represent various syntactic and semantic phenomena in the language of interest. A side effect of this, however, is that such grammars tend to be substantially different from each other, with no best practices or common representations.[2]

As implementations evolved for several languages within the same common formalism, it became clear that homogeneity among existing grammars could be increased and development cost for new grammars greatly reduced by compiling an inventory of cross-linguistically valid (or at least useful) types and constructions. To speed up and simplify the grammar development as well as provide a common framework, making the resulting grammars more

[1] Deep Linguistic Processing with HPSG Initiative (DELPH-IN), URL: http://www.delph-in.net/

[2] Exceptions do exist, of course: ParGram (Parallel Grammar) project is one example of multiple grammars developed using a common standard. It aims at producing wide coverage grammars for a wide variety of languages. These are written collaboratively within the linguistic framework of Lexical Functional Grammar (LFG) and with a commonly-agreed-upon set of grammatical features.

URL: http://www2.parc.com/isl/groups/nltt/pargram/



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 1–11

comparable the LinGO[3] Grammar Matrix[4] has been set up as a multi-lingual grammar engineering project [4] which provides a web-based tool designed to support the creation of linguistically-motivated grammatical resources in the framework of HPSG [5].

The Grammar Matrix is written in the TDL (type description language) formalism, which is interpreted by the LKB[5] grammar development environment [6]. It is compatible with the broader range of DELPH-IN tools, e.g., for machine translation [7], treebanking [8] and parse selection [9].

## 1.2 LinGO Grammar Matrix

Generally speaking, the Grammar Matrix is an attempt to distill the wisdom of already existing broad coverage grammars and document it in a form that can be used as the basis for new grammars. The main goals are to develop in detail semantic representations and the syntax-semantics interface, consistent with other work in HPSG; to represent generalizations across linguistic objects and across languages; and to allow for very quick start-up as the Matrix is applied to new languages.

The fact that different parts of a single grammar can be abstracted into separate, independent modules, either for processing or grammar development, is approached in [10] from the perspective of reuse of grammar code. A web-based configuration system elicits typological information from the user-linguist through a questionnaire [10, 11] and then outputs a grammar consisting of the Matrix core plus selected types, rules and constraints from the libraries according to the specifications in the questionnaire, and lexical entries for the language in question. In other words, users specify phenomena relevant to their particular language, with their selections being compiled from libraries of available analyses into a starter grammar which can be immediately loaded into the LKB grammar development environment [6], as well as the PET parser [12], in order to parse sentences using the rules and constraints defined therein. The regression testing facilities of [incr tsdb()] allow for rapid experimentation with alternative analyses as new phenomena are brought into the grammar [13]. The original Grammar Matrix consisted of types defining the basic feature geometry, e.g. [14], types for lexical and syntactic rules encoding the ways that heads combine with arguments and adjuncts, and configuration files for the LKB grammar development environment [6] and the PET system [12]. Subsequent releases have refined the original types and developed a lexical hierarchy, including linking types for relating syntactic to semantic arguments, and the constraints required to compositionally build up semantic representations in the format of Minimal Recursion Semantics [15-17]. The constraints in this 'core' Matrix are intended to be language-independent and monotonically extensible in any given grammar. In its recent development, the Grammar Matrix project aims at employing typologically motivated, customizable extensions to a language-independent core grammar.

The implemented prototype consists of a small set of modules targeting basic word order (addressing the relative order of subjects, verbs, and verbal complements), sentential negation, main-clause yes-no questions, and a small range of lexical entries. In particular:

- The Matrix core grammar provides definitions of basic head-complement and head-subject schemata which are consistent with the implementation of compositional semantics [16], as well as definitions of head-initial and head-final phrase types. The **word order** module creates subtypes joining the head-complement and head-subject schemata with the types specifying head/dependent order, creates instances of those types as required by the LKB parser, and constrains the rules to eliminate spurious ambiguity in the case of free word order.
- For **yes-no questions**, four alternatives have been implemented: inversion of the subject and a main or auxiliary verb relative to declarative word order and sentence initial or final question particles.
- The **sentential negation** module handles two general negation strategies: via verbal inflection or via a negative adverb. Neither, either or both of these strategies may be selected.
- In a strongly lexicalist theory like HPSG, words tend to carry quite a bit of information, which is reflected in the **lexicon** structure. This information is encoded in lexical types; lexical entries merely specify the type they instantiate, their orthographic form, and their semantic predicate. Many of the constraints required (e.g., for the linking of syntactic to semantic arguments) are already provided by the core Matrix. However, there also is cross-linguistic variation. The forms are assumed to be fully inflected (modulo negation), support morphological processes awaiting future work. This information and the knowledge

[3] The Linguistic Grammars Online (LinGO) team is committed to the development of linguistically precise grammars based on the HPSG framework, and general-purpose tools for use in grammar engineering, profiling, parsing and generation. URL: http://lingo.stanford.edu/

[4] URL: http://www.delph-in.net/matrix/

[5] LKB (Linguistic Knowledge Builder) system is a grammar and lexicon development environment for use with unification-based linguistic formalisms. While not restricted to HPSG, the LKB implements the DELPH-IN reference formalism of typed feature structures (jointly with other DELPH-IN software using the same formalism).

base are used to produce a set of lexical types inheriting from the types defined in the core Matrix and specifying appropriate language-specific constraints, and a set of lexical entries.

In a lexicalist constraint-based framework, the grammars are expressed as a collection of typed feature structures which are arranged into a hierarchy such that information shared across multiple lexical entries or construction types is represented only on a single supertype. As a result, a cross-linguistic type hierarchy comes with a collection of phenomenon-specific libraries.

## 2. Typologically motivated modularity

Aiming at typologically motivated modularity [10] describe a method for extending a language-independent core grammar with modules handling cross-linguistically variable but still recurring patterns. This method allows for extremely rapid prototyping of HPSG-conform grammars in such a way that the prototypes themselves can serve as the basis for sustained development, being able to scale up to broad-coverage resource grammars. The authors envision four potential uses for such grammar prototyping: (i) in pedagogical contexts, where it would allow grammar engineering students to more quickly work on cutting-edge problems; (ii) in language documentation, where a documentary linguist in the field might be collaborating remotely with a grammar engineer to propose and test hypotheses; (iii) in leveraging the results from economically powerful languages to reduce the cost of creating resources for minority languages; and (iv) in supporting typological or comparative studies of linguistic phenomena or interactions between phenomena across languages.

The modular approach of [10] has been designed to handle two kinds of typological variation. On the one hand, there are systems (formal or functional) which must be represented in every language. For example, every language has some set of permissible word orders (formal) and a means of expressing sentential negation (functional). On the other hand, there are linguistic phenomena which appear in only some languages, and are not typically conceptualized as alternative realizations of some universal function, phenomena such as noun incorporation, numeral classifiers, and auxiliary verbs. It is indeed expected that the constraint definitions which are supplied to grammar developers can be extended to capture generalizations holding only for subsets of languages.

The strategy of [10] is actually consistent with data driven investigation of linguistic universals and constraints on cross-linguistic variation. Therefore, we refer to this approach of grammar development (with the help from Grammar Matrix) as the "bottom-up" approach, because it is driven purely by the specific phenomena in the target language. It is certainly efficient: Specifying the choice file and building a small working grammar can be done within an hour (excluding the time spent on deciding specific choices for the given language). For instance [18] reports a relatively short development time required to create a precision, hand-built grammar for the Australian language Wambaya as a qualitative evaluation of the Grammar Matrix as a cross-linguistic resource.[6] The major drawback of this approach, however, is that, for the set of customized grammars for a group of languages, it soon becomes difficult (if not impossible) to harmonize the treatment of related phenomena across languages. With grammars being created individually, the treatment of shared phenomena would work to the degree that satisfies but does not guarantee cross-linguistic compatibility. As the number and breadth of implemented grammars grows, linguistic predictions are expected to emerge and become part of improved modules, particularly with respect to interactions among the distinct phenomena covered. Our focus in creating a Slavic Grammar Matrix is therefore somewhat different: since we are dealing with representatives of a language family, this effectively enables a “top-down” perspective in the multilingual grammar design.

It is an appealing goal indeed to develop a theoretical account of the way that language variation may be described in HPSG. Crucial in this respect is the fact that the HPSG framework allows a clean way of encoding at least some aspects of language variability within the type system. Another more ambitious line of research is initiated for investigating whether the description of differences among any set of two or more languages can be reduced to a minimal set of types. Progress in this research will lead to shared portions of grammar since the similarity of phenomena among the different languages will then be reflected in identical HPSG descriptions within the type systems. The goal is a grammar matrix designed for maximum reusability, lifting out the elements that can and should be common across HPSG grammars. Therefore, it is important to determine which analyses or building blocks of analyses appear to be cross-linguistically applicable. As the grammar matrix won’t be a complete grammar fragment by itself, it will be used in combination with mini-grammars for various languages. For a language family, and closely related languages in general, it is certainly justified to introduce intermediate

[6] Despite large typological differences between Wambaya and the languages on which the development of the resource was based, the Grammar Matrix is found to provide a significant jump-start as the creation of grammar itself is reported to have taken less than 5.5 person-weeks of effort.

parameterizations of the cross-linguistic core of the grammar matrix.

## 2.1 SlaviGraM: Slavic Grammar Matrix

The common properties of Slavic languages have been observed both in literature and related research at various intermediate levels of linguistic abstraction. Intermediate levels of typological variation are essential to our project because we work with a closed set of well-studied, well-documented and generally resource-rich languages belonging to the same language family. In this context, the interesting question arises whether minimal differences are also detectable as parameters of systematic variation.

Our concept of Slavic core grammar (Figure 1) will shape up and crystallize through rigorous testing in parallel grammar engineering for a closed set of richly documented and well studied genetically related languages for which a variety of linguistic resources is already available. We use Grammar Matrix to quickly build small grammars for individual languages, utilizing the online Matrix configuration system[7] to specify choice files for representatives of each Slavic subgroup, namely for Russian (East Slavic), Bulgarian (South Slavic) and Polish (West Slavic), as an initial step.

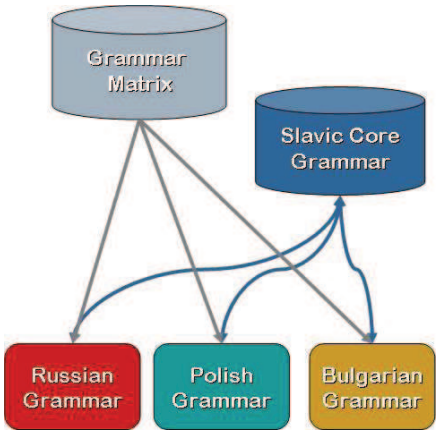
Apart from the shared core in the Grammar Matrix, however, the customization script treats the individual languages as separate instances, which means that the fact that we have to do with a group of closely related languages cannot be taken into account in the original setting. Therefore, shared analyses from individual languages are put into the Slavic Core in the form of generalized Slavic hierarchy and libraries. When new language is added, the Slavic core helps to more efficiently build the new grammar, and potentially receives cross-Slavic validation.

[7] The system consists of the following three parts:

- **Customization Page**. In order for the system to create a starter grammar, the required information must be elicited from the user-linguist. The medium for this elicitation is a web interface.
- **Choices File**. The options selected by the user are saved in a plain text file, called the choices file. Before a grammar is built, the choices file is verified to be internally consistent and contain all the information it needs.
- **Customization Script**. Matrix grammars are written in a type description language (TDL). The customization script is a Python script that reads in the choices file, and uses the information it contains to select or construct relevant sections of TDL code. The output is a collection of files containing the language-specific TDL code. This is then bundled with the core Matrix files to provide a small but functioning grammar fragment.

Our approach to Slavic grammatical resources is unique in the sense that grammar engineering for each individual language takes place in a common Slavic setting. This in particular means that if for example two possibilities are conceivable of how to model a particular phenomenon observed in a certain Slavic language, then we strongly prefer the option that would potentially be consistent with what is found in the other grammars. As a result the Matrix-driven starter grammars for Russian and Bulgarian, the two typological extremes within the Slavic language family, eventually incorporate novel theoretical decisions even for seemingly trivial tasks.

Figure 1: Matrix-driven starter grammars in Slavic core grammar setting




The Grammar Matrix in combination with the Slavic Core Grammar allows new grammars to directly leverage the expertise in grammar engineering gained in extensive work on previous grammars of the same language family. Both the general LinGO grammar Matrix and the Slavic Core Grammar are not static objects, but are designed to evolve and be refined as more languages are covered. The advantage of separating the Slavic Core Grammar from the general grammar Matrix is that the closed set of languages under consideration allow our Slavic Core to evolve more liberally than the grammar Matrix, without concerns over unstudied languages.

# 3. Focus on Russian Resource Grammar

As a showcase, let us consider the he Russian HPSG grammar, which is currently under active construction in our group. In fact, the Russian Resource Grammar has a central position in the SlaviGraM project and is anticipated as a major outcome in terms of end product and a large-scale experimental set up for hypothesis testing. The interesting aspect of the initial Russian grammar is that its development is assisted by interfacing with existing corpora and processing tools for the

language, which saves significant amount of engineering effort.

## 3.1 Morphological pre-processing

The morphological information associated with word forms in the disambiguated part of the Russian National Corpus, i.e. where the full analysis is displayed, is structured into four fields:

(i) lexeme and its part of speech;

(ii) (word-classifying invariable features (for example, gender for nouns and transitivity for verbs);

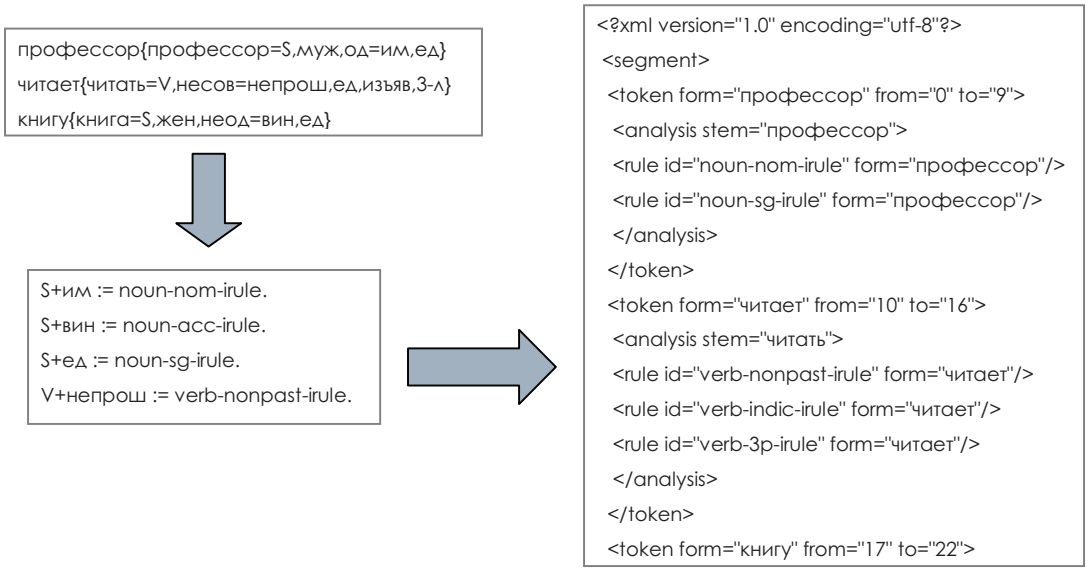
(iii) word-form specific inflectional features (for example, case for nouns and number for verbs);

(iv) non-standard forms, orthographic variations, etc. In the rest of the corpus only the lexeme and the part of speech are displayed.

Morphological analysis is the basic enabling technology for many kinds of text processing. Integrating a morphological analyzer is a crucial prerequisite for all grammar development activities involving Slavic languages. For research purposes, such systems are by and large freely available nowadays, and the LKB grammar engineering environment provides the required interface for integrating a morphological pre-processor.

For the pre-processing module we have considered two morphological analyzers for Russian: Mystem [19] and Dialing [20]. Both systems of are based on finite state transducers and have been used in the Russian National Corpus. Unlike Mystem, the system Dialing covers both inflectional and derivational morphology and is based on a large dictionary which also contains information on inflections, prefixes and affixes and stress patterns.

Figure 2: Morphological input via inflectional rules


профессор{профессор=S,муж,од=им,ед}
читает{читать=V,несов=непрош,ед,изъяв,3-л}
книгу{книга=S,жен,неод=вин,ед}
S+им := noun-nom-irule.
S+вин := noun-acc-irule.
S+ед := noun-sg-irule.
V+непрош := verb-nonpast-irule.
<?xml version="1.0" encoding="utf-8"?>

<token form="профессор" from="0" to="9">
<analysis stem="профессор">
<rule id="noun-nom-irule" form="профессор"/>
<rule id="noun-sg-irule" form="профессор"/>
</analysis>
</token>
<token form="читает" from="10" to="16">
<analysis stem="читать">
<rule id="verb-nonpast-irule" form="читает"/>
<rule id="verb-indic-irule" form="читает"/>
<rule id="verb-3p-irule" form="читает"/>
</analysis>
</token>
<token form="книгу" from="17" to="22">

Mystem, however, is the morphological component used by the popular Russian search engine Yandex. The underlying algorithm for analysis and synthesis achieves quite precise inflectional morphology of a wide lexical coverage without implying any particular morphological model of the dictionary.

The fact that Mystem is available for Polish too is an additional criterion in favor of adopting it in our project. Thus, during the preparatory phase, we have chosen the system Mystem, and it is already integrated as a morphological pre-processor in the LKB environment, as illustrated in Figure 2.

The Russian National Corpus is without a doubt an important source of structured grammatical knowledge to be integrated in our Russian grammar. A snapshot of the main search interface to the RNC is given in Figure 3, while Figure 6 illustrates the access to the syntactically annotated and disambiguated sub-corpus of the RNC. Furthermore, Figure 4 gives us the inventory of morphologically relevant "grammatical features" to select from in the main corpus. Note, however, that the inventory of grammatical features accessed from the syntactic search page is somewhat different, as shown by Figure 5.

Unlike the morphologically annotated portion of the RNC, the deeply annotated sub-corpus only contains fully disambiguated annotations (i.e. both morphological and syntactic ambiguity is resolved).

## 3.2 Syntactic dependencies

In the deeply annotated sub-corpus of the RNC, every sentence is marked up with a dependency syntactic structure – cf. Figure 7, with nodes corresponding to the words of the sentence, and labeled edges encoding the syntactic relations.

Figure 3: RNC search

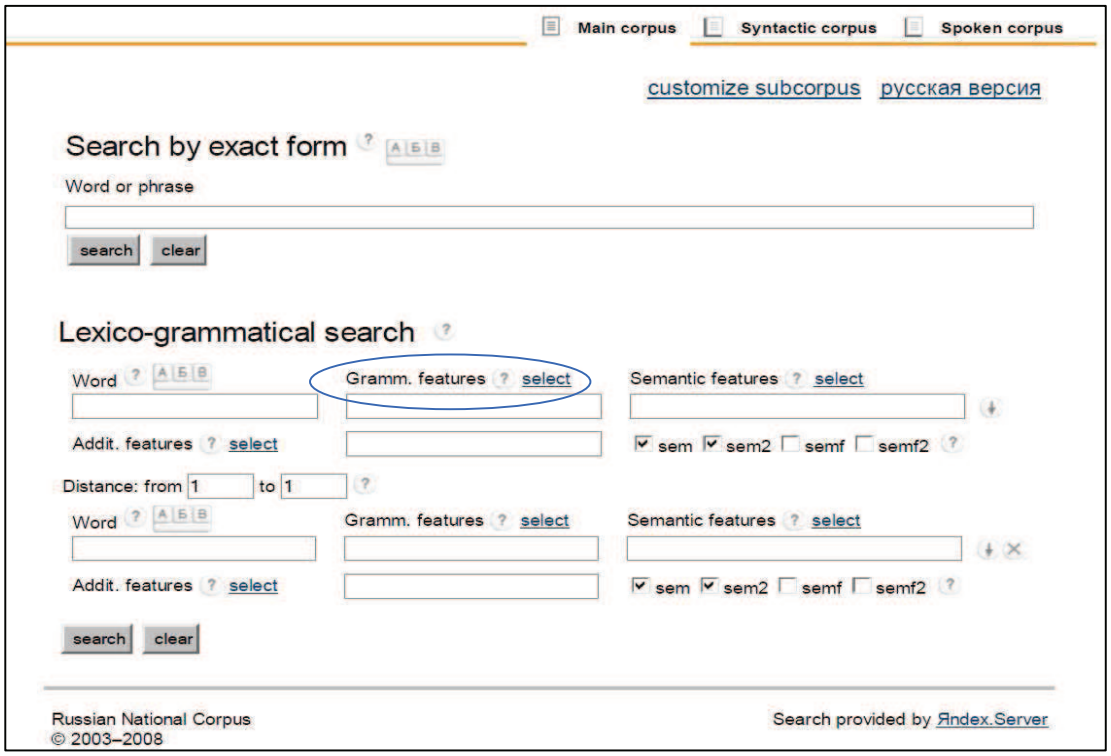



Figure 4: Morphological information in RNC

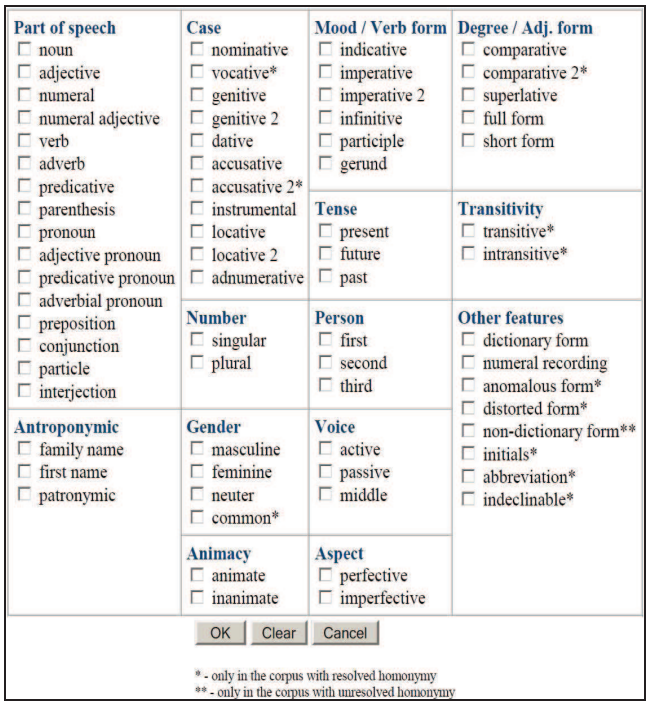



Figure 5: Grammatical features used in the syntactic sub-corpus

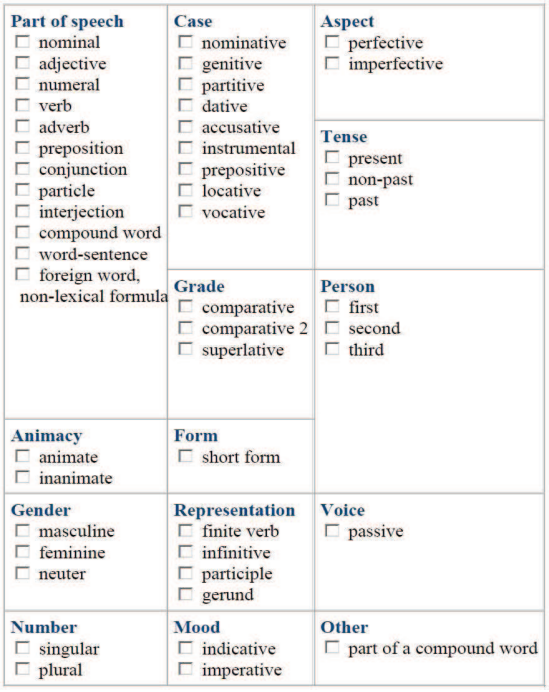
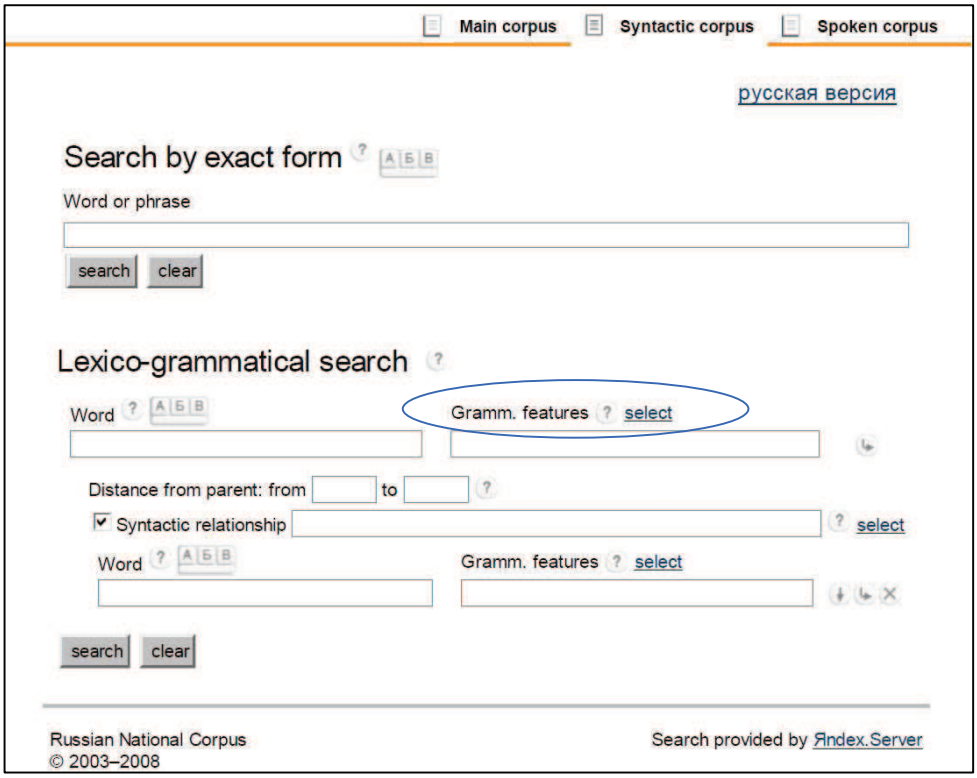
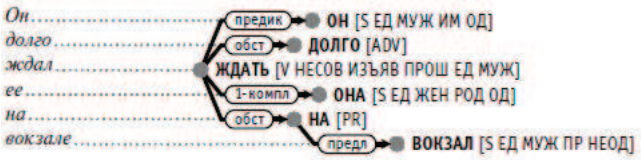


Figure 6: RNC access to the syntactic sub-corpus




The syntactic formalism originates in the Meaning-Text Theory [21], but the inventory of syntactic relations has been extended for the purposes of corpus annotation, incorporating a number of specific linguistic decisions [22, 23].

Figure 7: A sample structure




We, therefore, observe the following straightforward convention when the components of a RNC dependency relation (cf. the inventory in

Figure **8**) are to be mapped on HPSG categories: in a given syntactic dependency relation, the "governor X" corresponds in HPSG to the lexical head of the head daughter, while the "dependent Y" corresponds to the lexical head of the non-head daughter.

As actantial surface syntactic relations connect a predicate word [X] with its syntactic argument [Y], they would by and large map to headed phrases saturating valence requirements. For instance, the first syntactic argument [Y] stands in a predicative, dative-subjective, agentive or quasi-agentive relation to its head [X]. In HPSG, this corresponds to the first position on the head's ARG-ST (argument structure) list, e.g. to the "a-subject". Only in the prototypical predicative relation, however, this also is the single element on the SUBJ (subject) valence list. A non-first syntactic argument [Y] stands in a completive relation to its head [X]. As a rule, the direct object of a transitive verb stands in the first-completive relation to its head while non-transitive single-argument verbs like "sleep", for instance, take no completive relations whatsoever. Eventually, there could be several completive relations, depending on the actual valence requirements of the head. In HPSG this corresponds to the second, third, etc. positions on the head's ARG-ST (argument structure) list. A second large group of surface syntactic relations contains attributive dependencies. These relations connect a word [X] with its dependent word [Y] which functions as a modifier, i.e. is not subcategorized, and by and large would map in an HPSG setup to head–adjunct phrases

As for coordinative constructions, these are conceived in dependency syntax as directed asymmetric relations and in this respect do not stand out from the rest. In an HPSG setup, however, this group of relations would correspond to various types of (non-headed) coordinate phrases. The

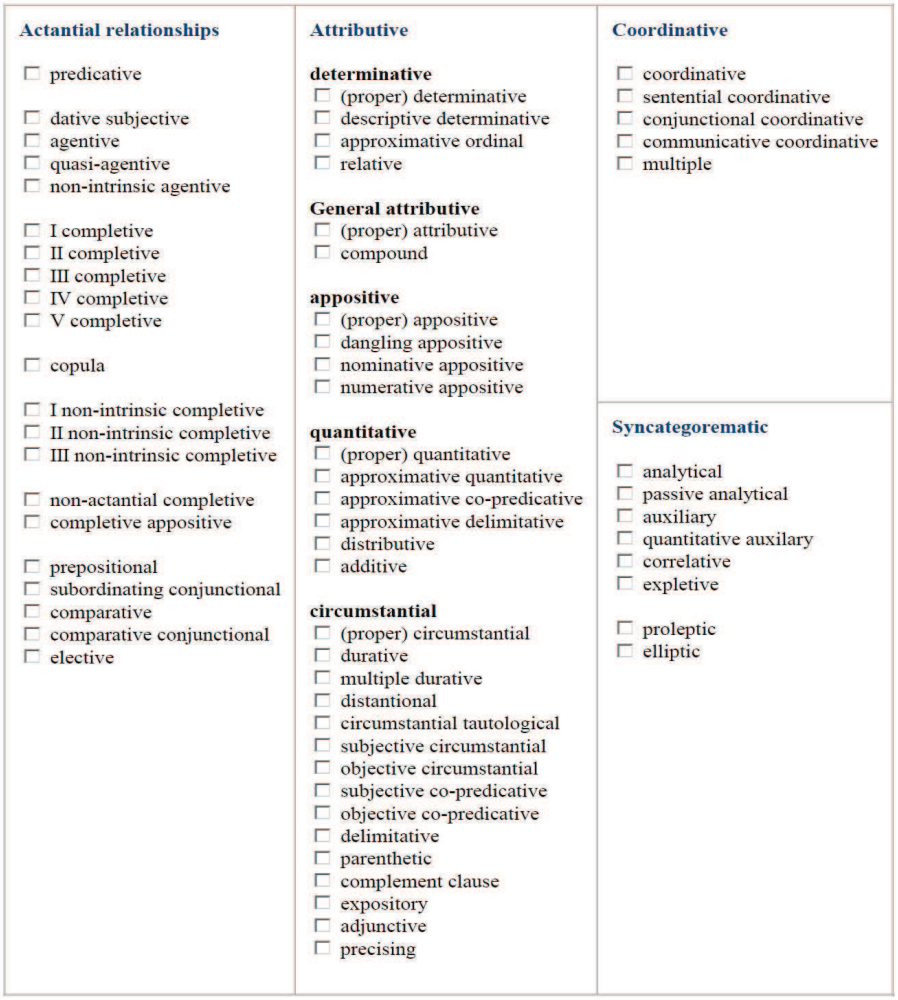
so-called syncategorematic dependencies connect two tightly bound elements [X] and [Y] that are often conceived as intrinsic parts of a larger unit, e.g. of a compound. In an HPSG setup, this group of relations would only partly correspond to headed phrases with functional categories, e.g. auxiliary verbs.

In this valuable resource, even more structured grammatical knowledge is accessible, e.g. with regard to multi-word expressions (MWE), syntactic ellipsis and gapping. The RNC website contains structured lists of orthographically multi-componential lexical units enriched with frequency information from the disambiguated sub-corpus. Based on the collocation analysis and lexicographic resources, two general MWE types are distinguished.

Inasmuch as the components of a MWE can be neither changed nor separated, it is considered equivalent to a single word and represented as a separate node in the syntactic structure. To this first type belong fixed expressions functioning as: (i) prepositions, e.g. по отношению (in relation to); (ii) conjunctions, e.g. коль скоро (as soon); (iii) particles, e.g. разве что (unless), что ни есть (no matter), не то чтобы (not that), нет-нет да и (once in while); (iv) adverbs, e.g. пока что (as yet), как бы то ни было (anyway), чуть ли не (almost), скрепя сердце (reluctantly), из рук вон плохо (thoroughly bad), стало быть (thus), то и дело (time and again), в обнимку (embracing each other), испокон веков (since the beginning of time).

On the other hand, there are syntactically transparent expressions whose components show certain degree of inflectional variation or allow other words to intervene in between. For such a MWE no standard syntactic structure is built, but (some of) its components are combined in an auxiliary dependency relation. It is assumed to hold (from X to Y) in the following examples: сам[Y] себя[X] (oneself); изо[X] дня в[Y] день (from day to day);

Figure 8: Syntactic relations in RNC



так[Y] называемый[X] (so called); все[Y] равно[X] (all the same); знать[Y] не знаю[X] (me having no idea whatsoever); дурак[Y]-то он дурак[X] (him being admittedly a fool).

In elliptical constructions the missing words are reconstructed in the syntactic annotation as "phantom" units which participate in the respective syntactic dependencies without introducing any changes in the original text. Similar approach is adopted in case of gapping, i.e. in constructions with missing verb of "vague" semantic content. An additional empty node is included the dependency structure, with its lemma set to "non-specific verb" assigning it the most plausible characteristics and, based on them, an indication of a lexeme that would represents a "natural hypothesis" for the missing verb.

Having adopted linguistically informed strategies in the modular grammar design, we deliberately concentrate on making the most of the freely available structured grammatical knowledge in the Russian National Corpus. Interfacing with existing corpora and processing tools is therefore fundamental to the Russian Resource Grammar development.

## 4. Proof-of-concept Implementation

To wrap up, here are some basic figures on the current state of the Russian grammar: ~1000 lines of code (excluding Matrix files and lexicon); ~350 newly introduced types (excluding Matrix types). The invested grammar engineering effort can be estimated as approximately 100 person hours of collaborative grammar development, plus some help from student assistants. Already at this initial stage, the Russian grammar covers the basic word order and agreement phenomena, as well as linking of syntactic to semantic arguments, case assignment by verbs to dependents, 'pro-drop' and argument optionality (2, 3, 9), passive (5) and various impersonal constructions (8, 11, 13, 14), among other mosphosyntactic phenomena.

(1) Профессор читает книгу
professor[nom.mask.sg] read[pres.act.3sg]
book[acc.fem.sg]
'The professor reads the book.'

(2) Профессор читает
professor[nom.mask.sg] read[pres.act.3sg]
'The professor reads.'

(3) Читает книгу
read[pres.act.3sg] book[acc.fem.sg]
'(pro-drop) reads the book.'

(4) Студент решает задачу
student[nom.mask.sg] solve[pres.act.3sg]
task[acc.fem.sg]
'The student solves the task.'

(5) Задача решена
task[nom.fem.sg] solve[pcp.pass.sg.fem]
'The task is solved.'

(6) Профессор дал задачу студентам
professor[nom.mask.sg] give[past.sg.masc]
task[acc.fem.sg] student[dat.pl]
'The professor gave the task to the students.'

(7) Вопрос требует особого внимания
question[nom.masc.sg] require[pres.3sg]
special[gen.neut.sg] attention[gen.neut.sg]
'The question requires special attention.'

(8) Быстро светает.
quickly dawn[pres.3sg]
'It's dawning quickly.'

(9) Пишем новую статью
write[pres.1pl] new[acc.fem.sg] article[acc.fem.sg]
'We write a new article.'

(10) Президент скоро лишится доверия
presient[nom.masc.sg] soon be-deprived[non-past.3sg]
trust[gen.neut.sg]
'The president will soon lose credibility.'

(11) Петра тошнит
Peter[acc.mask.sg] feel-sick[pres.3sg]
'Peter feels sick.'

(12) Старый профессор гордится студентами.
old[nom.mask.sg] professor[nom.mask.sg] be-proud[pres.3sg] student[ins.pl]
'The old professor is proud of the students.'

(13) Быстро темнело.
quickly get-dark[past.sg.neut]
'It was getting dark quickly.'

(14) Отцу нездоровится.
father[dat.masc,sg] feel-unwell[pres.3sg]
'Father does not feel well.

The linguistic analyses encoded in the grammar serve to map the surface strings to semantic representations in Minimal Recursion Semantics (MRS) format [15]. For instance, the MRS in Figure 9 is assigned to the example in (6). It includes the basic propositional structure: a situation of 'giving' in which the first argument, or agent, is 'professor', the second (recipient) is 'student', and the third (patient), is 'task'. The relations are given English predicate names for the convenience of the grammar developer. A simple tree display in **Figure 10** offers an abbreviated view over the HPSG derivation while hiding the detailed typed feature structures beneath away from the user.

Figure 9: MRS representation.

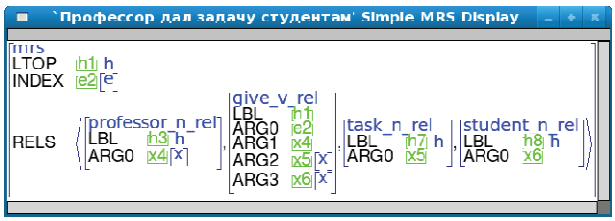
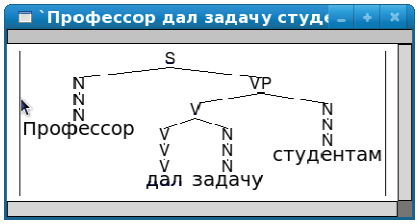



Figure 10: Tree representation




# 5. Outlook

In the elaboration of the individual grammars and especially for discovering structured linguistic knowledge to be reflected in the respective modules we shall systematically exploit the open access to rich linguistically interpreted corpora available for Slavic languages.

All individual grammars will be designed to support the innovative implementation of a Slavic core module that consolidates strategies for constructing a cross-linguistic resource based on concepts of shared and non-shared morphosyntactic phenomena.

An important desideratum for the individual resource grammars is to eventually couple them with treebanks which either pre-exist or will be constructed in parallel.

# 6. Acknowledgements

In all these areas, we anticipate international cooperation with distinguished research groups from the Russian Academy of Sciences (Leonid Iomdin), Bulgarian Academy of Sciences (Kiril Simov and Petya Osenova), Polish Academy of Sciences (Adam Przepiórkowski), and others. We envisage for this international exchange to eventually result in an international infrastructural project on Slavic corpora and grammars (SlaviCoGram), and are grateful to all these colleagues for preliminary discussions and constructive cooperation.

## References

[1] Uszkoreit, H., D. Flickinger, and S. Oepen, Proposal of Themes and Modalities for International Collaboration on Deep Linguistic Processing with HPSG. 2001, DFKI LT Lab and Saarland University, CSLI Stanford and YY Technologies.

[2] Uszkoreit, H. DELPHIN: Deep Linguistic Processing with HPSG -- an International Collaboration. 2002 http://hans.uszkoreit.net/delphinhome.html.

[3] Uszkoreit, H. New Chances for Deep Linguistic Processing. The 19th International Conference on Computational Linguistics COLING'02. 2002. Taipei, Taiwan.

[4] Bender, E.M., D. Flickinger, and S. Oepen. The Grammar Matrix: An Open-Source-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammars. Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics. 2002. Taipei, Taiwan.

[5] Pollard, C. and I. Sag, Head-Driven Phrase Structure Grammar. 1994, Chicago: University of Chicago Press.

[6] Copestake, A., Implementing Typed Feature Structure Grammars. CSLI Publications. 2002.

[7] Lønning, J.T. and S. Oepen. Re-usable tools for precision machine translation. COLING|ACL 2006 Interactive Presentation Sessions. 2006. Sydney, Australia.

[8] Oepen, S., et al., LinGO Redwoods. A rich and dynamic treebank for HPSG. Journal of Research on Language and Computation, 2004. 2(4 ): p. 575-596.

[9] Toutanova, K., et al., Stochastic HPSG parse selection using the Redwoods corpus. Journal of Research on Language and Computation, 2005. 3(1 ): p. 83-105.

[10] Bender, E.M. and D. Flickinger. Rapid Prototyping of Scalable Grammars: Towards Modularity Extensions to a Language-Independent Core. 2nd International Joint Conference on Natural Language Processing. 2005. Jeju, Korea.

[11] Drellishak, S. and E.M. Bender. A Coordination Module for a Crosslinguistic Grammar Resource. 12th International Conference on Head-Driven Phrase Structure Grammar. 2005. Stanford: CSLI.

[12] Callmeier, U., PET - a platform for experimentation with efficient HPSG processing techniques. Natural Language Engineering, 2000. 6 p. 99-107

[13] Oepen, S., et al. The LinGO Redwoods treebank. Motivation and preliminary applications. The 19th International Conference on Computational Linguistics. 2002. Taipei, Taiwan.

[14] Copestake, A., A. Lascarides, and D. Flickinger. An algebra for semantic construction in constraint-based grammars. The 39th Meeting of the Association for Computational Linguistics. 2001. Toulouse, France.

[15] Copestake, A., et al., Minimal Recursion Semantics: An Introduction. Journal of Research on Language and Computation, 2005. 3(4): p. 281-332.

[16] Flickinger, D. and E.M. Bender. Compositional Semantics in a Multilingual Grammar Resource. ESSLLI Workshop on Ideas and Strategies for Multilingual Grammar Development. 2003.

[17] Flickinger, D., E.M. Bender, and S. Oepen, MRS in the LinGO Grammar Matrix: A Practical User's Guide. 2003.

[18] Bender, E.M. Evaluating a Crosslinguistic Grammar Resource: A Case Study of Wambaya. ACL08 : HLT,. 2008. Columbus, Ohio.

[19] Segalovich, I. A fast morphological algorithm with unknown word guessing induced by a dictionary for a web search engine. MLMTA-2003. 2003. Las Vegas.

[20] Sokirko, A., Semantic Dictionaries in Automatic Text Processing (based on materials of the system DIALING) (in Russian). 2007.

[21] Mel'cuk, I.A., The Russian Language in the Meaning-Text Perspective. Wiener slawistischer Almanach. Vol. Sonderband 39. 1995, Moskau - Wien: Gesellschaft zur Förderung slawistischer Studien.

[22] Apresjan, J., et al. A Syntactically and Semantically Tagged Corpus of Russian: State of the Art and Prospects. The fifth international conference on Language Resources and Evaluation, LREC 2006. 2006. Genoa, Italy.

[23] Boguslavsky, I., et al. Development of a dependency treebank for Russian and its possible applications in NLP. The third International Conference on Language Resources and Evaluation (LREC-2002). 2002. Las Palmas

# Maximal Phrases Based Analysis for Prototyping Online Discussion Forums Postings

Gaston Burek Dale Gerdemann
Department of Linguistics
Tuebingen University
72074 Tuebingen, Germany
*[gaston.burek, dale.gerdemann]@googlemail.com*

## Abstract

Chat texts produced in an educational environment are categorized and rated for the purpose of positioning (or placement) of the learner with respect to a learning program (appropriate courses, textbooks, etc). The difficulty lies in the fact that the texts are short and informal. A standard LSA/vector-space model is therefore combined with techniques appropriate for short texts. The approach uses phrases rather than words in the term-document matrix, and for determining prototypical documents of each category, a nonparametric permutation test is used.

## 1 Introduction

Text categorization is a well-established sub-field of computational linguistics. We are interested in applying the techniques of text categorization for the purpose of positioning *Life Long Learners* with respect to educational programs and instructional materials. Quite simply, we can rate learner texts, likely to be short and generated informal educational settings, by using a vector-space comparison to gold-standard, expert texts. Then if the similarity is high enough, the learner will proficiency out of the course. This approach is straightforward, but is unlikely to be successful. The problems concern accuracy, suitability and justification of the categorization.

### 1.1 Accuracy

The categorization obviously must be accurate. A false positive, indicating learner proficiency in a particular domain, could be dangerous as it could lead to work place incompetence. A false negative, on the other hand, could lead to boredom, as the learner is forced to take courses on topics that he or she has already mastered.

The problem of accuracy is compounded by the fact that texts (selected from text collections known as *ePortfolios*) are often short. To deal with this problem, our approach attempts to lose as little information from the text as possible. Traditional approaches to categorization lose information by case normalization, stemming and ignoring word order. The idea of the traditional approach is to deal with the data sparseness problem by collapsing textual features into equivalence classes, losing information in the process.

In our approach, we attempt to balance the problem of data sparseness with the goal of not losing information. This balance is obtained in two ways. First, as discussed in section 2, we use *Latent Semantic Analysis* (LSA) as a technique for dimensionality reduction [8]. It is well known that LSA can be used to discover weighted clusters of words, which are loosely understood to be "concepts." Since these clusters can contain derivationally related terms, the need for stemming (and also case normalization) is reduced. Second, as discussed in 3, our more innovative contribution is to flexibly use bigrams, trigrams and other ngrams as opposed to strictly unigrams in the traditional bag-of-words model. Our approach to extracting such ngrams is to use a extension of the suffix array approach of [12].

### 1.2 Suitability

Suppose that learner texts could be accurately classified as similar or not similar to the gold standard text (or set of texts). Then the question arises as to whether or not the gold standard text is a suitable prototype for a good learner text. One approach to choosing a gold standard text would be to use a published journal article in the field. But such a text is unlikely to be similar to learner texts either in tone or in content. It is well known that effective teachers use *scaffolding* to present material within the *zone of proximal development* of the learner.

So perhaps a better gold standard would be a textbook, or other learning material, written at the level of the student. This is certainly an improvement, but on the other hand, it is still rather unreasonable to expect learners' texts to closely match the tone of a textbook, unless of course the learners are copying from the text. In fact, the texts that we have consist of online discussions of medical students on several topics related to safe prescribing. These texts have been categorized as to subtopic and graded for quality (excellent, good, fair, poor) by annotaters at the University of Manchester. The texts contain serious conversations, with very little off-topic wandering. But the tone of the texts is chatty, and not at all similar to textbook writing. So rather than to use an external gold standard, we have opted for an internal gold standard. The prototypical "excellent" text is simply one that was rated as "excellent" by the Manchester annotators. But not



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 12–18

all "excellent" texts are equally good as prototypes. Clearly, for any text $t$, the remaining texts can be ranked in order of similarity to $t$. If $t$ is a good prototype, then this ranking should other "excellent" texts as most similar to $t$ and "poor" should be least similar. In subsection 4, we discuss a method for choosing good prototypes that uses the nonparametric *permutation test*.

### 1.3 Justification: communities of practice distinctive language

It is widely accepted that experts can provide answers to problems that average people can not e.g. evidence given in court can be accepted or rejected on the basis of expertise, relevance and reliability. In this contexts expertise is defined as knowledge beyond common sense [6].

Communities of practice are at the center of expert knowledge development. According to [4] new communities of practice develop conditions for effective local creation and communication of knowledge and therefore that knowledge is not shared with other communities until it settles. In linguistic terms this can be understood as a distinctive use of language shared among individuals members of a specific community to describe knowledge that is not shared with other communities of practice. According to [2] expert language in medicine consists in shared specific terms formalized in medical term banks, thesauri and lexicons. Patients get familiar with that established medical terminology relevant to their own health conditions medical language after exposure to treatments, personal doctor visits, etc.

#### 1.3.1 Language technologies supporting the identification of expertise

As life long learners do not have common learning goals nor common educational backgrounds as it is the case in traditional learning settings, long life learning educational providers need to rely on available written materials produced by individual learner to identify their degree of expertise within areas of knowledge that are relevant to study programs in offer.

Given this scenario there is a need for tools that can provide support in determining the learner's degree of expertise or position by means of state of the art language technologies. Those technologies should be capable of identify linguistics features that reflect the degree of learner's expertise by analyzing learners' text repositories.

But, will learners affected by the use of such technologies (e.g. LSA) be happy to hear about a expert system suggestion that implies the need of studying a microbiology course on account of a low cosine similarity value between the learners essays and texts produced by microbiology experts? It is well known that users are more inclined to trust an expert system when such tool can give some reasons for its judgement.

Here, in addition to use LSA for comparing similarities between learner written text and expert texts, we present the use suffix array analysis for characterizing text in a way that users of that technologies can understand .

Grammar and language constructions can be used to identify language that is characteristic of people that is not familiarized with a relevant communities of practice e.g. microbiology . This linguistic feedback could be combined with a tentative suggestion that taking a microbiology course would be one way to acquire these linguistic conventions that tend to correlate with knowledge of microbiology.

Our work focus on the use analysis of distinctive phrases for identifying the degree of expertise of it author in a specific domain where that expertise may be expressed by linguistics features that is not be evident at surface level. This approach determine that degree of expertise as the probability of language misuse in relation to average language use as sampled from expert written texts.

To give an example from and available, consider the terms "prescription charts" and "drug charts." These terms apparently mean the same thing, but it turns out that "prescription charts" occurs predominately in texts rated excellent or good, and "drug charts" occurs predominately in texts rated fair or poor. Suppose that the hypothesis that doctors either consciously or unconsciously prefer the term "prescription charts." Then if a learner uses the term "drug charts," this usage could be tagged as less favored terminology, and a tentative suggestion could possibly be made that the learner could take a pharmacology course. If the learner ultimately rejects this suggestion, then at least this learner wouldn't go home empty handed. The learner would at least have received some linguistic feedback that would be unlikely to come from a human evaluator.

## 2 Latent Semantic Analysis

In recent years, LSA has been proposed as a suitable language technology for the automatic determining the degree of expertise of a specific text's author [10]. Although, singular value decomposition (SVD) of word co-occurrence frequencies matrices has been successfully used in the context of language technologies enhanced learning (e.g. automatic assessment of student essays), learner positioning presents new challenges that expose the limitation of such an approach. In particular, learners produce text repositories containing few samples of text, many of them of small size and using language that is rich in non domain-specific expressions. Such repositories may also be generated by individuals from different backgrounds in informal learning environments (chats, online forums, etc.). In addition, those texts are generated in contexts where learners feel encouraged to hide their poor usage of language by articulating redundant expressions and making extensive use keywords. Moreover, linking the semantics of high level descriptions of learning goals and domain specific terminology used by learners in a non-formal context restrict the usability of LSA as word usage may not be very stable across corpora.

While semantic spaces approaches such as the Vector Space Model (VSM) captures surface features of a semantic space such as first or second order word co-occurrence, the semantic similarity theory behind LSA is based on a model that captures different higher

orders of word co-occurrence (i.e. third order co-occurrence and higher) by means of using singular value decomposition. LSA uses SVD to project the semantic space over a lower dimensional space. LSA supporters claim that by reducing the space dimensionality, the semantic space that models the human cognitive process becomes more accurate due to the reduction of the noise that is added during the process of generating language.

LSA belongs to the family of semantic space models and is capable of inferring relations between words that are not explicit at surface level. LSA is popular with psychologists since the dimensionality reduction can be interpreted as reducing the word space to concept space. But ultimately these "concepts" are loadings on word counts, which are not easily interpreted or explained.

LSA as well the other semantic spaces approaches represents the semantics of words, sentences and paragraphs using word co-occurrence information that does not take in consideration the position of words within the sentence. However, according to a general conception of semantics, syntax plays a significant role in representing meaning of sentences. Thus, intuitively to using information about the order words occupy within expressions in addition to their frequency of occurrence seems to be a theoretically sounded approach to improve LSA performance that is in line with mayor trends in language acquisition theories that stress the significance that syntactic structure play in the comprehension of language.

Although, semantic spaces approaches (e.g. Latent Semantic Analysis) have been successful in reasoning about ambiguity and semantic similarity when analysing texts at the level of words (linguistic units), sentences (grammatical units) and paragraphs (discourse units) they are yet not capable of reasoning beyond the analysis of a bag of words.

Typically, LSA projects a semantic space implemented as a sparse matrix into a dimensionally reduced VSM generating the best statistical approximation to the original model. The dimensionally reduced model results in a matrix with non 0 values that allows the computation of similarity between matrix columns or rows that were orthogonal in the original model.

LSA theory assumes that many words have a similar probabilistic distribution that results in a comparatively lower number of concepts.

As the variation of word choice, introduce noise to the text, SVD, the algorithm behind LSA bridges the gaps in words by conflating word used into word senses . Furthermore, LSA facilitates matrix manipulation in terms of hardware operative memory by reducing the dimensionality of the VSM.

# 3 Phrases

Traditionally, text categorization by means of LSA has relied on a bag-of-words model. It seems, in some sense, obvious that a model based on phrases should be better. But it turns out that this is not necessarily the case. Recently, Bekkerman & Allan [3] reviewed the literature on text categorization and found no general improvement when unigram models were replaced with bigram models. The problem is that using bigrams contributes heavily to the data sparseness problem.

Bekkerman & Allan have, however, compared two rather extreme positions. Our idea is to extract phrases of any length from the the training corpus, as long as the phrases are distinctive (occurring predominately in particular categories of documents). It may well be that the most distinctive phrases are generally phrases of length one (concurring with the bag-of-words model), but if there are phrases of other lengths that are more distinctive, then there seems to be no reason not to use these phrases.

To give an idea of the approach, consider the word *side*. In the medical discussions in our corpus, this word almost always occurs as part of the phrase *side effect(s)*. In a few cases, *side* occurs in a unique context or as part of another phrase, such as *flip side*. In this case, the distinctive phrase is apparently *side effect*, and the other occurrences are just noise. These noise phrases are not only unhelpful for text categorization, they are are also unhelpful for generating explanations that would be useful for learners and examiners.

The example above raises some interesting counting issues. But first we need to specify more precisely what it means for a phrase to be distinctive.

## 3.1 Distinctiveness

In general, phrases that are evenly distributed across document categories are not very distinctive, whereas phrases that tend to cluster in one particular category are distinctive. This general principle must be applied carefully, however, since with small numbers, clustering may occur due to chance.

A common measure of distinctiveness used for weighting in vector space models is *tf-idf* (term frequency multiplied by inverse document frequency) [9]. It is unclear, however, that this is the best measure for picking out which phrases to consider and which phrases to ignore. It is problematic, for example, that *idf* simply prefers terms that cluster in a small number of documents, regardless of the classifications. Given the ordinal classification of our data as *excellent*, *good*, *fair* and *poor*, we are not interested, for example, in terms that cluster in the *excellent* and *poor* texts.

So a distinctive term should be one that occurs predominately in *excellent* and *good* texts or predominately in *fair* and *poor* texts. Consider, for example, the bullet point , with occurrence vector $\langle 31, 5, 0, 0\rangle$.[1] The interpretation is that there are 31 occurrences in *excellent* documents, 5 occurrences in *good* documents and no occurrences in either of the poorer texts, this term appears be very distinctive of better texts. But if we count instead the number of different documents the bullet point occurs $\langle 4, 1, 0, 0\rangle$, we see a very different picture. The bullet point does occur in higher rated texts, but it is very bursty (cite Church) and is therefore not very useful for categorization.

There are various approaches in the literature for dealing with burstiness. Since this is not our primary

[1] Optionally the tokenizer could be set to eliminate such punctuation marks. The bullet point makes a good example here, however, due to its burstiness.

concern here, we deal with the problem by counting the number of texts containing a term rather than the total number of occurrences of the term. Thus, for the bullet point, we use the vector $\langle 4, 1, 0, 0\rangle$.

To rate a term such as the bullet point, we need some measure of goodness for the vector $\langle 4, 1, 0, 0\rangle$. There is clearly no objective measure that can be used here. As a fairly reasonable score, we simply assign 1 point for every *excellent* text, 0.8 points for every *good* text and 0.2 points for every *fair* text. So, the bullet point receives a score of 4.8. This appears to be a good score, but what is the probability that a randomly chosen term appearing in 5 texts would have a higher or equally high score? We can generate random vectors as in (3.1) (where $e$, $g$, $f$ and $p$ are the total numbers of *excellent*, *good*, *fair* and *poor* texts, respectively.

$$X_i = \begin{cases} \langle 1,0,0,0\rangle & \text{with probability } \frac{e}{e+g+f+p} \\ \langle 0,1,0,0\rangle & \text{with probability } \frac{g}{e+g+f+p} \\ \langle 0,0,1,0\rangle & \text{with probability } \frac{f}{e+g+f+p} \\ \langle 0,0,0,1\rangle & \text{with probability } \frac{p}{e+g+f+p} \end{cases}$$

A vector for a random term occurring in $n$ texts is then $\sum_{i=0}^{n} X_i$. So for a good score such as 4.8, the idea is to see what proportion of randomly generated vectors have an equally high or higher score. And for a low score, the opposite idea is to count the proportion of randomly generated scores that are equal or lower.

## 3.2 Phrase Extraction

In principle, the distinctness measure given above can be used with phrases of any length. If longer phrases can be found that are more distinct than single words, then there is no reason not to use the longer phrase. The problem is that the simulation-based distinctness test is very expensive, and it is certainly not possible to run this test for ngrams of every length in a text. The solution to this problem comes from Yamamoto & Church [12], who show suffix arrays can be used to put the large number of ngrams into a much smaller number of equivalence classes. Using suffix arrays, it is very easy to pick out just the phrases that are repeated $n$ times for some $n$, and it is very easy to extend phrases to the right: if *mumbo jumbo* repeatedly occurs together as a phrase, then it makes no sense to count *mumbo* by itself. Yamamoto & Church's suffix array program will put these two phrases into an equivalence class, so that that statistics can be calculated for the class as a whole rather than individually for all the members of the class.

Since the time of Yamamoto & Church's paper, suffix arrays have been an active area of research, primerily in bioinformatics. One of the weaknesses of the suffix array approach used by Yamamoto & Church is that extensions to the left are difficult to discover. So it is difficult to discover, for example, that *jumbo* always combines to the left to form the phrase *mumbo jumbo*. Simply stated, the problem is that suffixes are extensions of phrases to the right, so it is hard to look to the left. This problem was solved, however, by Abouelhoda et al [1], who added a BurrowsWheeler transform table to their extended suffix array data structure, giving this this data structure properties of suffix trees.

One weakness of Abouelhoda et al's approach, however, is that it does not adapt well to large alphabets. This is, of course, a serious weakness for use in text processing, where one wants at least to work with some subset of Unicode, or even worse, to treat each tokenized word as an alphabet symbol. Fortunately, the restriction to small alphabet size has recently been eliminated in the approach of Kim et al [7], who deal with the large alphabet by using binary trees, which are linearly encoded using the *child table* of Abouelhoda et al along with a *longest common prefix* table (lcp).

Using extended suffix arrays makes it possible to count different kinds of occurrences of phrases in different ways. To begin with, we are only interested in counting phrases that repeat. In the text $S$ = *to be or not to be*, the occurrence of the phrase *to be* at $S[1, 2]$ is said to be a *repeat* since the same sequence of tokens occurs at $S[5, 6]$.[2] An occurrence of a phrase $S[i, j]$ is *left maximal* is *left maximal* if the longer phrase $S[i-1, j]$ is not a *repeat*. Thus, for example, the phrase *to* at $S[1, 1]$ is *left maximal* since the phrase at $S[0, 1]$ is not a *repeat*.[3] Similarly, an occurrence of a phrase at $S[i, j]$ is right maximal if $S[i, j + 1]$ is not a repeat. If an occurrence of a phrase is both left and right maximal, then the occurrence is said to be *maximal*. Note that the occurrence of the phrase *or not* at $S[3, 4]$ is *maximal*, though it is not a repeat. Since non-repeats are rarely of interest, we generally assume that we are talking about repeats unless otherwise stated.

A phrase is also said to be *maximal* in a text if there exists a maximal occurrence of the phrase in the text. For example, in the text *mining engineering*, tokenized by characters, the phrase *in* is maximal since there are maximal occurrences at $S[2, 3]$ and $S[11, 12]$. But the longer phrase *ing* is also maximal since it occurs maximally at $S[4, 6]$ and $S[16, 18]$. So the occurrence of *in* at $S[16, 17]$ is a non-maximal occurrence of a maximal phrase. A maximal repeated phrase that is not a subsequence of a longer maximal repeated phrase is said to be *supermaximal*. Thus the phrase *ing* is supermaximal in this text.

Generally, we are only interested in counting occurrences of maximal phrases since a phrase that never occurs maximally is unlikely to be of interest. But what kind of occurrences should we count? Should we count all occurrences, or only the left maximal, right maximal or maximal occurrences? The answer is that we don't need to decide ahead of time. We can simply test each of these four cases for distinctness, and chose the most distinct case. Take, for example the word *side*, which is a maximal phrase in our texts. Should we count all instances of this phrase? Or should we perhaps restrict the count to right maximal occurrences so as to avoid counting those instances that are extended to the right to create the longer phrase *side effect*? Or maybe left maximal occurrences to avoid the longer phrase *flip side*? Or perhaps we should restrict

[2] This definition and the following definitions are similar to those found in Abouelhoda et al [1]. The difference is that Abouelhoda et al apply the terms to a pair of occurrences, whereas we apply the terms to a single occurrence.

[3] We assume here that the text is padded with unique *beginning of string* and *end of string* sentinels so that indexing at 0 or 7 makes sense.

in both directions to avoid either kind of extension. Since it is not generally possible to predict which is best, the reasonable approach is to try all possibilities to see what works best.

One counterintuitive feature of our approach is that it also makes sense to count 0-grams. A left maximal occurrence of a 0-gram, for example, must have a hapax legomena to its left, and a maximal occurrence of a 0-gram must have hapax legomena on both sides. These sequences of two hapax legomena may well be distinctive, since they often are an indication of a named entity or a foreign phrases. Counting all occurrences of the empty sequence is, of course, equivalent to counting the text length, which may well also be a distinctive feature.

# 4 Permutation test for prototyping chat texts

Our approach to categorization is based on similarity to prototypical documents of each category. Here again, we are concerned with finding an approach of identifying good prototypes in a way that is suitable for use with a small data set. We note that each candidate prototype induces a ranking of the remaining documents from "most similar" to "least similar." For a good prototype, this ranking should be significantly different from a random permutation. To test this hypothesis, we can use a standard nonparametric test, known (appropriately enough) as the "permutation test." The permutation test is a nonparametric method for testing whether two distributions are the same. The test is "exact," meaning that it is not based on large sample theory approximations [11]. Suppose that $X_1, ..., X_m \sim F_X$ and $Y_1, ..., Y_n \sim F_Y$ are two independent samples and $H_0$ is the hypothesis that the two samples are identically distributed. This is the type of hypothesis we would consider when testing whether a treatment differs from a placebo. More precisely we are testing $H_0 : F_X = F_Y$ versus $H_1 : F_X \neq F_Y$. Let $T(X_1..., X_m, Y_1, ..., Y_n) = |\overline{X}_n - \overline{Y}_n|$ and let $N = m + n$ and consider forming all $N!$ permutations of the data $X_1..., X_m, Y_1, ..., Y_n$. For each permutation, compute the test statistic $T$. Denote these values by $T_1, ..., T_{N!}$. Under the null hypothesis, each of these values is equally likely. Let $t_{obs}$ be the observed value of the test statistic. We reject the hypothesis when $T$ is large.

Usually, it is not practical to evaluate all $N!$ permutatioons. We can approximate the $p - value$ by sampling randomly from the set of permutations. The fraction of times $T_j > t_{obs}$ among these samples approximates the $p - value$. In general If $T$ is smaller than some significance level $a$ , the results are significant at that level. In our case we are not concern about a particular significant level as we are looking to the most significant texts on certain topic and with certain grade that can be used use as gold standards. Table 1 shows a toy example for the cosine similarity vector $(X_1, X_2, Y_1) = (0.1, 0.5, 0.8)$ where, $|\overline{X}_n - \overline{Y}_n| = 0.50$.

The methodology described in this section can be used as a reasonable approach for measuring how significant (representative) chat texts are for a particular category and grade. The estimation is calculated on the basis of cosine similarity between the texts and does not assume any particular distribution for those values.

| permutation | value of T | probability |
|---|---|---|
| ( 0.1, 0.5, 0.8) | 0.50 | 1/6 |
| ( 0.1, 0.8, 0.5) | 0.05 | 1/6 |
| ( 0.5, 0.1, 0.8) | 0.50 | 1/6 |
| ( 0.5, 0.8, 0.1) | 0.55 | 1/6 |
| ( 0.8, 0.1, 0.5) | 0.05 | 1/6 |
| ( 0.8, 0.5, 0.1) | 0.55 | 1/6 |

**Table 1:** *Permutation example*

# 5 Experiments

The experiments described in this section compare the performance of the traditional bag of words LSA configuration against an alternative configuration that uses maximal phrases as unit of analysis.

The alternative LSA configuration starts with a vector space model that (instead of using words counts) uses counts of highly distinctive phrases that occur at least one time as maximal phrase within the text collection under analysis.

The use of a "bag of phrases" model instead of a "bag of words" model is motivated by the small size of our sample annotated of chat texts. Since our information source is scarce, we cannot afford lose any of it. If a medical text contains the phrase "side effect," for example, it is significant information that these two words occur as a phrase, which would be lost in the traditional bag-of-words approach.

Our ongoing work in word co-occurrence models for learner positioning extends existent LSA based approaches and is aimed at analyzing and then scoring texts posted on an online medical student discussion forum where students discuss issues related to the safe prescribing of medicines, guided by clearly defined expected learning outcomes associated to six subtopics. The 504 postings have been annotated by experts with four grades (i.e. 109 poor, 200 fair, 142 good, 50 excellent) and one of six topics (i.e. 42 of topic a , 50 of b, 130 of c, 22 of d , 247 of e and 13 of f). Each grade is based on the individual posting's textual contribution to a series of expected learning outcomes. Highly scored postings can then be used as evidence of learner proficiency in the corresponding topic.

## 5.1 Building the bag of words and phrases based semantic spaces

As explained in 3.2 to identify and extract the maximal phrases we analyze suffix arrays using an extended version of the the Yamamoto and Church algorithm to generate all ngrams from a text and avoiding the combinatorial explosion by grouping these ngrams into equivalence classes. This approach, however, has the disadvantage of generating also "uninteresting" ngrams, which are neither maximal nor supermaximal. To overcome this deficiency, we employ the Burrows and Wheeler transformation table.

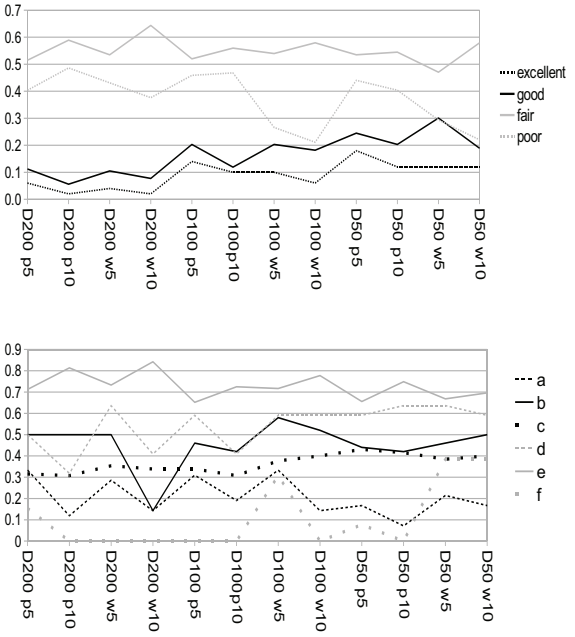



**Fig. 1:** *kNN results for neighborhood sizes 5 and 10 and semantic spaces built from phrases (p) and bags of words (w) using 5, 100 and 200 singular values*

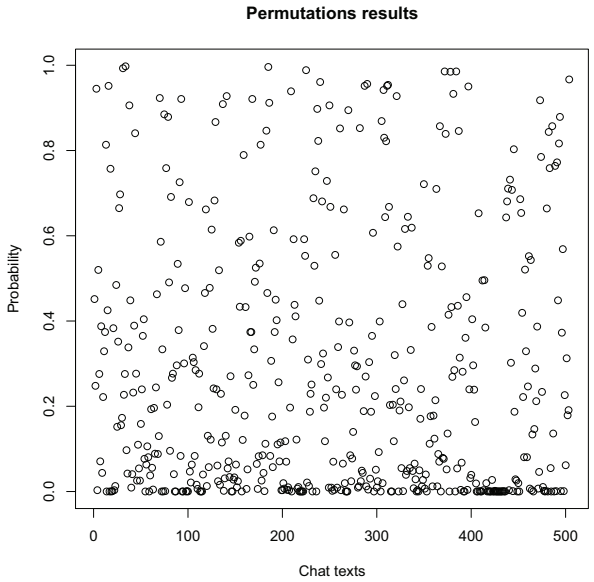



**Fig. 2:** *Permutation test results for each chat text*

Each phrase has been counted in one of 4 ways: all instances, left-maximal, right-maximal and maximal. To avoid an unmanageable level of sparseness we include in the analysis all instances of all phrases that occurs at least one time as maximal. Phrases are sorted by their scores absolute values.

We then built a 19730 phrases to 504 chat texts matrix that contains the frequency of occurrence of each phrase in each texts. We then weighted the matrix using the $tf - idf$ weighting scheme. We then generate three LSA semantics spaces by reducing the SVD resulting matrix singular values to 50, 100, and 200 respectively.

In addition we created another set of 3 bag of words based LSA semantic spaces using the same weighting scheme and respective number singular values. In this case using a 6320 tokens to 504 chat texts matrix. The number of token used is the results of choosing the tokens that occurs at least two times within the chat texts collection.

## 5.2 k Nearest Neighbor algorithm based classification

The k Nearest Neighbors algorithm (kNN) [5] is a learning algorithm that classifies texts on the basis of a measure of distance (e.g. cosine) between them. The algorithm classifies each text by looking at a k number of its nearest neighbors and then assigning it to the most common category represented by those neighbors. If no class is associated to a majority of neighbors, the text is assigned to the category represented by texts with higher cosine similarity. We arbitrarily used a low k value (i.e. k=5) as we expect that noise from the semantic space will be reduced by means of LSA. A common criticism of kNN is that, since it doesn't make any generalizations from the data, it is prone to overfitting. We assume that this criticism should not apply completely to semantic spaces generated by means of LSA as the SVD dimensional reduction smoothes and therefore reduces the effect of over fitting that is usually present in kNN based classification.

### 5.2.1 kNN results

Experimental results showed in Figure 1 demonstrate that for some topics and grades using maximal phrases as units of analysis can improve the performance of LSA. In fact the best results for two of the four grades (e.g. excellent and poor) were yielded by semantic spaces build from phrases. Different results are produced by kNN classification for topics where semantic spaces built from bags of words produced the best results clearly for at least four of the six topics.

## 5.3 Prototyping

We run the permutation test described in section 4 using the already available semantic space built from a 19730 phrases to 504 chat texts matrix and singular values reduced to 200. We then evaluate the significance of each text for its annotated grade and topic category. Figure 2 shows that the majority of texts

have low probability as the majority of them may not be a good representative of their class.

# 6 Conclusions

For particular grades and topics phrase based LSA (i.e. using semantic spaces built from phrases occurring at least one time as maximal) appears to improve over LSA results that have been obtained with the traditional bags of words approach. These results are encouraging and therefore we plan to test alternative semantic space configurations in particular using more distinctive phrases (e.g. all maximal, left maximal and right maximal ). We expect that as we collect a larger text sample we will be able to afford the use of those phrases without facing unmanageable levels of sparseness in detrimental of results already obtained. In addition, we want to stress the fact that the suffix array analysis presented in this paper is independent from its LSA application as it can be used to characterize learners and experts use of language. In addition, as our approach to categorization is based on similarity to prototypical texts for each class we presented here an non parametric test (permutation test) for identifying those prototypes.

# References


[1] M. I. Abouelhoda, S. Kurtz, and E. Ohlebusch. Replacing suffix trees with enhanced suffix arrays. *J. of Discrete Algorithms*, 2(1):53–86, 2004.

[2] H. Åhlfeldt, L. Borin, P. Daumke, N. Grabar, C. Hallett, D. Hardcastle, D. Kokkinakis, C. Mancini, K. Markó, M. Merkel, C. Pietsch, R. Power, D. Scott, A. Silvervarg, M. T. Gronostaj, S. Williams, and A. Willis. Literature review on patient-friendly documentation systems. Technical Report 2006/04, Centre for Research in Computing, The Open University, Milton Keynes, UK, May 2006. ISSN: 1744-1986.

[3] R. Bekkerman and J. Allan. Using bigrams in text categorization. Technical Report IR-408, Center of Intelligent Information Retrieval, UMass Amherst, 2004.

[4] J. S. Brown and P. Duguid. Knowledge and organization: A social-practice perspective. *Organization Science*, 12(2):198–213, March 2001.

[5] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[6] A. Davies and C. Elder, editors. *The Handbook of Applied Linguistics*. Blackwell Handbooks in Linguistics. Wiley-Blackwell, Malden, 2006.

[7] D. K. Kim, M. Kim, and H. Park. Linearized suffix tree: an efficient index data structure with the capabilities of suffix trees and suffix arrays. *Algorithmica*, 52(3):350–377, 2008.

[8] T. K. Landauer and S. T. Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, April 1997.

[9] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. In *Information Processing and Management*, pages 513–523, 1988.

[10] J. van Bruggen, P. Sloep, P. van Rosmalen, F. Brouns, H. Vogten, R. Koper, and C. Tattersall. Latent semantic analysis as a tool for learner positioning in learning networks for lifelong learning. In *British Journal of Educational Technology*, number 6, pages 729–738, 2004.

[11] L. Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer, September 2004.

[12] M. Yamamoto and K. W. Church. Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus. *Comput. Linguist.*, 27(1):1–30, 2001.

# LEXIE - an Experiment in Lexical Information Extraction

John J. Camilleri
Dept. Intelligent Computer Systems
University of Malta
MSD2080 Msida, Malta
*john@johnjcamilleri.com*

Michael Rosner
Dept. Intelligent Computer Systems
University of Malta
MSD2080 Msida, Malta
*mike.rosner@um.edu.mt*

## Abstract

This document investigates the possibility of extracting lexical information automatically from the pages of a printed dictionary of Maltese. An experiment was carried out on a small sample of dictionary entries using hand-crafted rules to parse the entries. Although the results obtained were quite promising, a major problem turned out to errors introduced by OCR and the inconsistent style adopted for writing dictionary entries.

## Keywords

lexicon extraction, lexical information, lexicon, semitic

## 1 Introduction

This paper describes an experiment carried out within the general context of the Maltilex project [6], the objective of which is the development of a computational lexicon for the Maltese language. The compilation of such a body of lexical information for an entire language in machine readable form is a formidible task. Intuitively, we are aiming for a lexicon that comprises a set of entries under which all the information relating to a particular word is stored.

It is particularly difficult for Maltese because the Semitic component of the language is morphologically rich, so the set of valid, naturally occurring word forms is, in principle, larger than the set of lexical entries. Hence, certain issues about the role of morphological analysis and lexical organisation must be resolved if the set of entries is to be effectively delimited.

Fortunately, a lot of the of the work has already been carried out by the compiler of what still remains the most comprehensive printed Maltese-English dictionary, Joseph Aquilina [2]. The question, therefore, is whether the information, and to some extent, the organisation already present in that dictionary can be exploited.

The more general issue under investigation in the context of the present workshop is the re-use of expensively produced paper lexical resources by translation into a more useable electronic form.

The approach is not entirely new. Some work along these lines was carried out for the Longman Dictionary of English by Boguraev et al [3] and we owe much to the general approach adopted there. However, the two main differences we see between their approach and ours are that (a) Boguraev's work was oriented rather heavily towards the GPSG grammatical framework and (b) they were able to access the original source tapes of the dictionary. The input they used was therefore faithful to the orginal.

In our case we had to rely upon OCR input as described further in section 4. The inherent errors caused certain problems.

One of the aims of this work is to establish a viable method for extracting lexical entries for lesser-studied languages lacking the usual battery of language resources in machine readable form. Most languages of the world fall into this category. The availability of a paper dictionary, is, however, fairly widespread, even for exotic languages so that the methods being proposed could provide a robust alternative to dictionary extraction for a relatively large number of languages.

In terms of the types of information to be extracted from these pages, the aim of this experiment was to produce, with a reasonable level of accuracy, the following: (i) a correct list of headwords appearing in the dictionary, and (ii) associated lexical information for each headword in this list.

Additionally, to facilitate the interoperability of the extracted information with any other NLP applications which may make use of it, the format chosen for the final output lexicon was to conform to the evolving Lexical Markup Framework (LMF) [5] ISO standard.

This paper is structured as follows. Sections 2 and 3 respectively describe the formats of printed dictionary and LMF output. Section 4 describes the data used for the experiment. The main part the paper is in section 5 which explains the pipeline architecture used to process the input. The paper concludes with sections 6, 7, 8 and 9, describing results, limitations and future work.

## 2 Aquilina's Dictionary Entries

Figure 1 is a scan of part of a typical dictionary page. We can discern two main entries, *SKORĊ|A* and *SKORD|ATURA* where the vertical bar divides the stem on the left from an affix on the right. In subsequent parts of the entry, occurrences of tilde are replaced with the stem. The reader should also note the presence of an alternate spelling *SKURDATURA*



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 19–26

**SKORĊ|A** [Bon], n.f. (pl. ~*i*) **1.** Scum; impurities that rise to surface of liquid especially in boiling or fermentation; floating film; ~ *ta' ġerħa*, scab of a wound; ~ *tal-ħalib*, scum of milk. **2.** Crust (of s.th. solid); ~ *tar-raba'*, hard surface of soil; [Diz Tek p. 255] ~ *tal-ħadid*, scale. **3.** Thickening of skin, face, etc. **4.** (masonry) The hard detritus underlying superficial soil, an aggregate of broken or loose fragments. [<Sic. *scorcia*/It. *scorza*]

**SKORD|ATURA/[ESI] SKURDATURA,** n.f. (pl. ~*i*) **1.** Music out of tune. **2.** (mus.) Deliberate mistuning of a string to produce an unusual effect. [Ps] ~**|A,** v.t. (imperf. +*a*, pp. ~*at*/[Bus] *skurdat*, vn. ~*ar*) **1.** To put out of tune ([id.] *il-pjanu tiegħi* ~*at*, my piano is out of tune). **2.** To disconcert, to worry (*kemm jien* ~*ata llum!*, I feel very out of sorts today). [<It. *scord-atura*; *-are*]

**Fig. 1:** *Sample dictionary entries from Aquilina*

for the second entry. In this case it is clear that the two spellings are part of the same entry, but this is not always the case. The numbered subentries refer to alternate word senses. Items in square brackets give the etymology of the word. We are mostly interested in the morpho-syntactic information which is represented by various abbreviated forms. A full list of such abbreviations is supplied with the dictionary, and the most frequent ones are are shown in Table 1.

## 3 LMF

In order to fullfil out goal of creating a viable method for the extraction of lexical information, the output format has to be defined.

Paper lexicons contain quite a variety of information, as revealed from a rudimentary glance at the scanned entries of figure 1. Besides spelling and morpho-syntactic information, there is also definition, translation, and semantic field information. Aquilina's dictionary also contains references to other entries as shown by the entry for *skorfnott* in figure 2 which refers to the entry for *SKORFNA*.

Our choice of format has been heavily influenced by a desire to have the potential to capture all these different kinds of information and at the same time to remain theory independent.

There are many potential formats that could be adopted. However, many of these are to some extent theory specific, or else overly biased towards particular kinds of information.

Lexical Markup Framework (LMF) is an evolving ISO[1] standard now in its 16th revision whose main goals are "to provide a common model for the creation and use of lexical resources, to manage the exchange of data between and among these resources, and to enable the merging of large number of individual electronic resources to form extensive global electronic resources" (see Francopoulo-et-al [4]).

[1] ISO-24613:2008

| PARTS OF SPEECH | |
|---|---|
| a./adj | adjective |
| a.ag. | adjective of agent |
| adv. | adverb |
| ap. | active participle |
| conj. | conjunction |
| interj | interjection |
| n. | noun |
| n.ag. | noun of agent |
| nom. | nominal |
| n.u. | noun of unity |
| pp. | past participle |
| pron. | pronoun |
| pr.n. | proper noun |
| v. | verb |
| vn. | verbal noun |
| vn.u. | verbal noun of unity |
| TENSE/ASPECT/MOOD | |
| fut. | future |
| pres. | present (tense) |
| imperat. | imperative |
| imperf. | imperfect |
| perf. | perfect (tense) |
| pass. | passive |
| VERB TYPE | |
| intr. | intransitive |
| t./trans. | transitive |
| refl. | reflexive |
| GENDER/CASE | |
| c. | common |
| f./fem | feminine |
| m./masc. | masculine |
| obj. | object |
| voc. | vocative |
| NUMBER | |
| du. | dual |
| pl. | plural |
| s./sing. | singular |

**Table 1:** *Common Abbreviations*

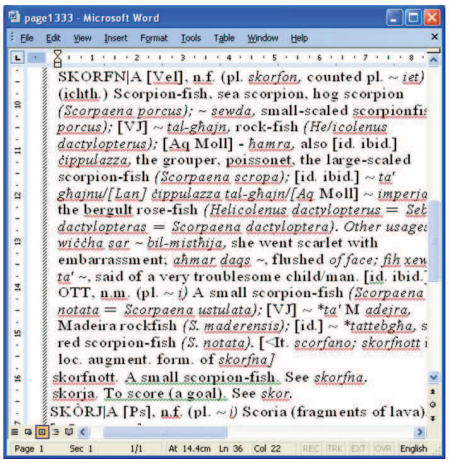



**Fig. 2:** *References to other entries*

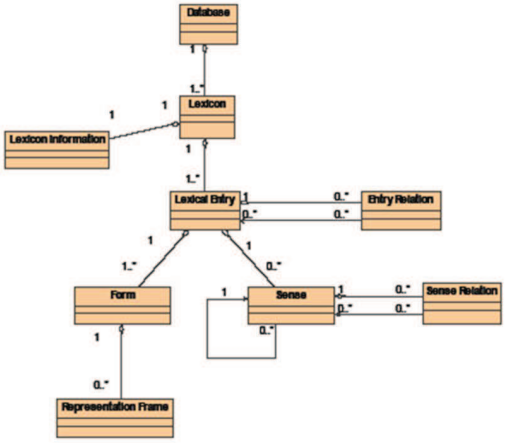



**Fig. 3:** *LMF framework [4]*

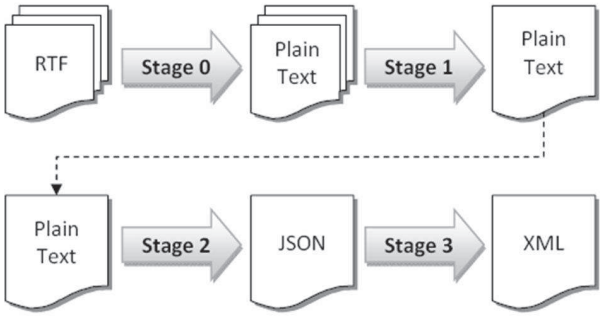



**Fig. 4:** *Processing Overview*

The model for lexical resources consists of a *core package*, which specifies how a lexical entry fits into the structure of a generalised lexical database, together with a set of extensions for handling various types of linguistic information such as morphology, syntax, semantics etc.

The diagram of figure 3 shows the organisation of the core package. Here we can see that a model is a Database which comprises one or more entities of type Lexicon, and a Lexicon comprises one or more entities of type Lexical Entry. The main interest is in the structure of the lexical entry.

# 4 Data

As part of a separate project being carried out at the University of Arizona on lexical perception by Ussishkin and Twist [1], the entire Aquilina dictionary has been scanned and run through Optical Character Recognition (OCR) software and saved as RTF files. Subsets of these files have been proof read and were kindly provided for experimental use in LEXIE.

The experiment reported here was carried out on two datasets. The first was used for development of the system and comprised pages 1333-1347 of the dictionary, containing approximately 360 entries. The second dataset, pages 1480-1490, was "unseen", and used for subsequent validation as described further in section 6

# 5 Processing

The different processing tasks required in this project were split into four distinct stages as shown in figure 4.

## 5.1 Stage 0 Pre-Processing: RTF-UTF8

The scanned dictionary pages provided for this task were saved in rich text format (RTF), with one document supplied per page. A part of the page for the entry shown is illustrated in figure 5.

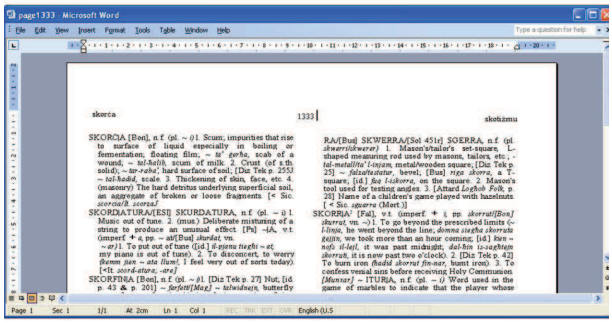


**Fig. 5:** *RTF Format*

Thus the first task was to convert these files in a format which could be more easily read by subsequent stages of the program. The initial task was correct encoding of characters into Unicode (UTF8).

At first, the idea of manually parsing out all the RTF controls from the sample pages was considered. However it soon became clear that writing a separate tool for converting RTF files into plaintext was beyond the scope of this project.

A number of freeware third-party tools available over the internet were tried but none of them were able to to correctly translate the Maltese characters into Unicode UTF8. For this reason a macro was recorded for Microsoft word.

## 5.2 Stage 1: Separation of Entries

The result of conversion to UTF8 is shown in figure 6.

While the dictionary pages were now in plain text form, they still contained a large number of irregularities which needed to be cleaned up before the parsing of the pages on a per-entry level could begin. The purpose of this processing stage was to bring all of the entries from the plain text pages together in a single file, neatly separated by newline characters.

However, as a result of the OCR quality of the sam-

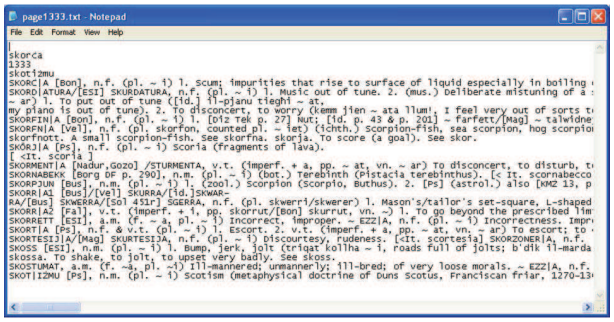



**Fig. 6:** *UTF8 Format*

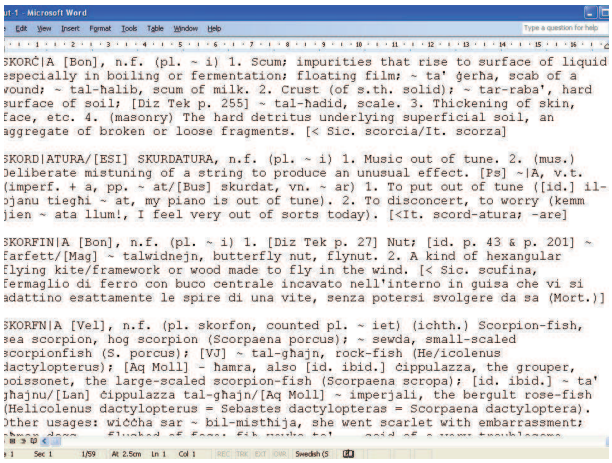



**Fig. 7:** *Separated Entries, one per line*

ple pages, considerable work was required to get the entries into this format. The major issues encountered at this stage were the following:

- inclusion of extra page information such as page numbers and first and last headwords. These can clearly be seen in the first three lines of figure 6.
- Distribution of single entries over multiple lines. The line starting "my piano is out of tune" is an an example, being part of the entry for *SKORDAR*, one of the senses of the entry whose stem is *SKORD*.
- multiple entries combined together into single line.

  Clearly, a key issue is the individuation of distinct entries, and in particular how to determine whether a given line represents the beginning of a new entry.

  The following criteria were employed. If all these are met, then the line is deemed to be the start of a new entry (this does not exclude the possibility that other entries might be appended to it:

  – The first character is a letter (euphonic i is ignored).
  – It is followed by a punctuation character.
  – Last character of previous entry is not a hyphen.
  – The first letters of the line are in ascending alphabetic order with respect to the previous entries.

  The selection of these criteria proved to be quite problematic, as for every feature which a group of headwords seem to share, there were inevitably be some which did not follow the pattern. The objective success of the criteria adopted is reflected in the results reported in section 6.
- Finally, there are a number of OCR errors. Some of these were common enough to warrant automatic correction. For example, the string "l." at the beginning of a word sense should be replaced with the string "1."

The output of this phase, with one entry per line, is shown in figure 7.

## 5.3 Stage 2: Parsing of Entries

Now that the text entries have been neatly separated, the next step was to parse each of them and extract all necessary pieces of information which would be needed in the third and final output stage.

The basic order of operations is as follows:

```
For each line
    - Get first word
    - If uppercase:
        - Set type = "ENTRY"
        - Get number after headword
        - Get headword, stem & root
        - Get etymological info, eg "[< Eng./It. Stalatt
        - Get & format alternate spellings
        - List any subentries
        - Get senses for each subentry

    - If lowercase:
        - Set type = "CROSS-REF"
        - Get & format alternate spellings
        - List cross-references with variation types
    Add to entries list
Encode entries list using JSON and write to file.
```

While the output from stage one was a simply plain-text file, the output for this stage needed to have a lot more structure to it, now that each entry has been broken down into its sub-parts. As per the system design, it is only in stage three where the entries are converted into the final LMF output.

Thus the internal output from the current stage of processing must be independent of the final LMF output format. One option for encoding these entries was to save them in a generalised XML format, independent of the LMF output in stage three. While viable, a faster and equally as robust method was to output the structured entries in JSON format.

JSON (JavaScript Object Notation)[2] is a very simple text-based data-interchange format that is com-

[2] www.json.org

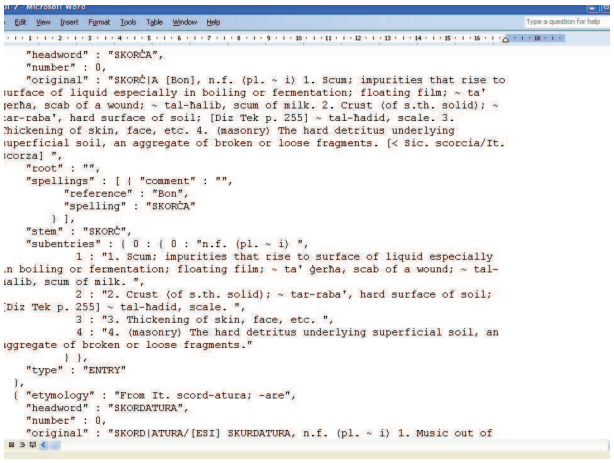



**Fig. 8:** *Output after parsing entries*


**Fig. 9:** *Output after conversion to LMF*

pletely language independent but uses conventions that are familiar to programmers of the C-family of languages. Essentially, JSON is built on two structures: (i) a collection of name/value pairs, and (ii) an ordered list of values, realized in most languages as an array, vector, list, or sequence.

The actual translation was done using the open source `demjson` Python module (7). With just one line of code, the entire entries list was converted into JSON format and saved to file, to be fed as input into the final stage of processing. Figure 8 shows the result of this process.

### 5.4 Stage 3: Output LMF

With the entries now individually separated and broken down into their sub-parts, the final stage of the process was to generate the output lexicon containing the parsed entries in an LMF-compliant XML format as diagrammed in figure 10.

To further separate the processing from the presentation aspects of the code, it was decided to use a number of templates in the generation of the final LMF output. This is not to say that the code of stage three is completely output-independent, however the use of templates definitely helped to promote this separation.

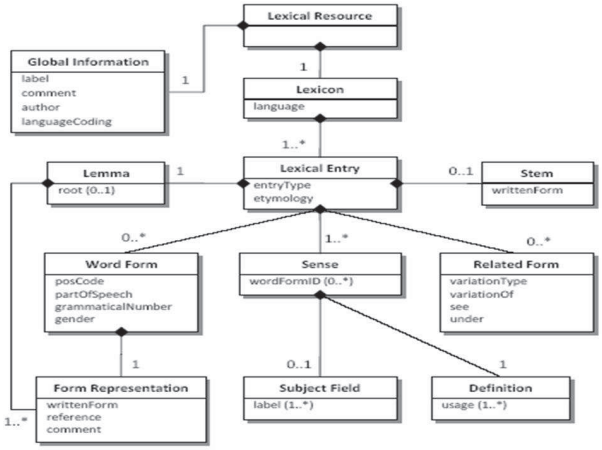



**Fig. 10:** *XML output format*

The templates used are listed below:

- `LexicalResource.xml` Represents the entire lexicon and root element for the output XML.
- `LexicalEntry.xml` A single entry in the lexicon (including cross-references).
- `FormRepresentation.xml` Used to represent alternative spellings of the same word.
- `WordForm.xml` Used to represent different forms of a word, e.g. through conjugation.
- `Sense.xml` Represents a single sense (definition and usage) or a word/word form.
- `RelatedForm.xml` Used in cross-references to indicate the referenced entry.
- `feat.xml` A simple attribute/value pair.

The output of stage 3 is the final LMF-compliant lexicon XML file as shown in figure 9.

### 5.5 Format Validation

To ensure that the generated XML lexicon was well structured and consistent with the LMF standard, a final step of validation was performed on the file. Firstly, by opening the newly created file with an XML parser, the file contents was parsed and implicitly checked to be a valid XML document.

The successfully parsed XML was then validated against the LMF DTD file to check conformity to the LMF standard. Both these tasks are achieved using the `lxml` Python library.[3]

## 6 Results and Evaluation

### 6.1 LMF Conformity

When validating against the official LMF Rev 16 DTD the generated lexicon did not pass because (i)

[3] The `lxml` library is available at `http://codespeak.net/lxml/validation.html`

the `WordForm` element had no attribute id, (ii) the `RelatedForm` element had no attribute id and (iii) in many cases the `RelatedForm` elements targets attribute contains nonexistent IDs.

The first 2 points are genuine non-conformities to the official LMF standard. However, the inclusion of a simple id attribute is only a very minor infringement, and for the purposes of the project was deemed a fair one to make. In order to accommodate the addition of these new attributes, a modified version of the original LMF DTD was created and used for subsequent validation.

In the third case, the issue is that as this lexicon only covers a sample of pages from the entire dictionary, some of the cross-referenced entries do not appear in the same file. This is quite understandable. To get around this issue and continue validation, all cross-referenced entries were temporarily removed from the lexicon. Once this was done, the output successfully passed DTD validation suggesting that if the file were to contain the entire dictionary it should also comply with the LMF standard.

### 6.2 Method of Evaluation

Once the output lexicon had been generated and validated, the next important step was to evaluate its accuracy against the original dictionary. This was achieved by manually comparing the output of the program with the original OCRd dictionary pages, and enumerating the number of correct, partial, incorrect, and missing entries.

First a more human-friendly version of the original XML lexicon was generated for evaluation purposes using PHP. Two HTML documents were generated and printed out for the evaluator to manually check and mark against for each entry.

### 6.3 Evaluation Criteria

Each entry extracted and placed into the output lexicon was given one of the following designations:

- **Correct**: the entire entry was correctly extracted.
- **Partial**: the headword and some parts of the entry are correct; however some parts are in error.
- **Incorrect**: the headword is seriously corrupted or not a headword at all.

In addition, for each extracted entry a count of any missing entries not picked up by the parser was also kept. This information was then used in the calculation of the programs accuracy, as explained in the following section.

### 6.4 Equations Used

The equations used in the calculation of the accuracy score are given below.

- Strict Score $= \frac{\text{Correct}}{\text{Total+Missed}}$
- Lax Score $= \frac{\text{Correct+Partial}}{\text{Total+Missed}}$

| | known | unknown |
|---|---|---|
| Page Range | 1333-1347 | 1480 - 1490 |
| Total Entries | 360 | 370 |
| Correct | 290 | 261 |
| Partial | 64 | 84 |
| Incorrect | 6 | 25 |
| Missed | 34 | 47 |
| Strict Score % | 73.6 | 62.59 |
| Lax Score % | 89.85 | 82.73 |

**Table 2:** *Evaluation data*

### 6.5 Known and Unknown Pages

In the development of this project, the same subset of dictionary pages was used throughout. This would certainly have introduced a certain bias of the program to perform better on these pages than it would on the dictionary as a whole. To test this, the programs accuracy was evaluated and analyzed on two subsets of dictionary pages one which was used throughout development ("known"), and one which has never been shown to the program (or developer) before ("unknown"). The results of both cases are presented in the next section.

## 7 Discussion

### 7.1 Results

Although the 62.59% for unknown pages leaves plenty of room for improvement, as discussed further below, these results are quite promising. This percentage represents a sufficently high level of accurate results to warrant further investigation of methods which can further reduce the human effort required to filter out incorrect results.

### 7.2 OCR Problems

The primary difficulty encountered in this project was the quality and consistency of the sample dictionary pages provided. Although passed through OCR software and supposedly checked by hand, the accuracy of the provided pages was far from ideal.

Apart from oft-mistaken character sequences such as "]"for "J" and "|" for "I", the major issue encountered was that of inconsistent entry formatting. This in particular included entries split across multiple paragraphs, multiple entries collapsed into a single line, and incorrect block indentation. While noticeable to human readers, these issues presented a major hurdle for the extraction process, and at least half of all the effort put into this project was devoted to correcting these OCR-related errors.

### 7.3 Variation of Notation in Dictionary

Another source of difficulty encountered was the notational variation present in the source dictionary. This was especially true for multiple word forms or definitions within an entry.

While in some entries they are listed in one format, in others they may be listed in a different format. It should be noted that these inconsistencies have been created by the author of the dictionary. Though the author may have had his reasons for such variations, they are neither obvious nor fully documented. As a result, a number of errors found in the output of this program can be attributed to these inconsistencies.

Another case of inconsistency is the use of the tilde character as a back-reference. Most of the time it refers to the previously-established headword stem, but sometimes it refers to the entire headword. Once again, what it refers to in each case is not always obvious to the reader, let alone a computer, and this ambiguity contributed to a substantial number of word form errors generated by the program.

# 8 Limitations

## 8.1 Lossiness RTF Conversion

The first stage in this project involved converting the RTF sample pages into plain text equivalents. While this provided many benefits it terms of ease of development, it also inevitably presented its own set of limitations. One of these is the loss of all formatting information, such as bold and italic text. As such formatting may contain additional information about the entry (e.g. examples of use are written in italics), it would have been preferred if these could have been retained and used during the extraction process.

## 8.2 Cross-Reference IDs

In the submitted program, whenever a cross-reference entry is processed, the existence of the main entries referred to are not explicitly checked. Instead, they are simply transcribed from the source, which means that cross-references may exist in the output lexicon with invalid reference IDs. As only a handful of pages were processed for the purposes of this project, the verification of these IDs would be somewhat futile. However in a complete version of the lexicon, these ID references would need to be verified.

## 8.3 Entry Definitions

Most of the effort carried out in this project is devoted to extracting headword variations and different word forms. Less focus however was placed on the parsing of the word definitions themselves, and in many cases the information placed in each Sense element is simply copied verbatim from the dictionary. In particular,the following issues were not addressed:

- Non-textual characters are not removed.
- Word definitions are not separated from examples of usage.
- Abbreviations and back-references are not replaced with their full equivalents.

  We do not anticipate that addressing any of these points would introduce major structural changes to the program.

# 9 Future Work

## 9.1 Scaling Up

This experiment was carried out on a total of approximately 700 lexical entries taken from 20 dictionary pages. Although results are promising, the extent to which they generalise is not clear and for this reason an absolute priority is to repeat the experiment on a much larger dataset.

## 9.2 More Thorough Error Correction

While a substantial amount of work in this project was devoted to error correction, the techniques used are far from complete. Many of the OCR errors found in the sample pages are not easily correctable with basic pattern-matching techniques, and require deeper analysis as to how they occur and can be removed. With a more dedicated effort devoted to the correction of these errors, the accuracy of the system could undoubtedly be pushed significantly higher.

## 9.3 Use of Statistical Methods

The level of accuracy achieved in this project was achieved through the use of standard string pattern-matching with regular expressions. Whilst these methods are highly effective when used in the correct contect, one major limitation is that such methods do not exploit the statistical regularities inherent in the language of dictionary entries.

A possible way forward would be to develop statistical models for language of dictionary entries, and to use these models to error correct the dictionary entries obtained by OCR. Inspection of dictionary entries reveals that a dictionary entry is composed of several parts not all of which share the same language. Hence, there is scope for investigating the sublanguages that make up dictionary entries and developing statistical models for each.

# References


[1] U. A. and A. Twist. Auditory and visual lexical decision in maltese. In C. B., R. Fabri, E. Hume, M. Mifsud, T. Stolz, and M. Vanhove, editors, *Introducing Maltese Linguistics, Selected papers from the 1st International Conference on Maltese Linguistics*, pages 233–249. John Benjmins Publiching Company, 2007.

[2] J. Aquilina. *Concise Maltese Dictionary*. Midsea Books, Valletta, 2006.

[3] B. Boguraev, T. Briscoe, J. Carroll, D. Carter, and C. Grover. The derivation of a grammatically indexed lexicon from the longman dictionary of contemporary english. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, pages 193–200, Stanford, California, USA, July 1987. Association for Computational Linguistics.

[4] G. Francopoulo, M. George, N. Calzolari, M. Monachini, N. Bel, M. Pet, and C. Soria. Lexical markup framework (lmf). In *Proc. LREC 2006*, pages 233–236. ELRA, 2006.

[5] ISO/TC37/SC4. *Language Resource Management, Lexical Markup Framework (LMF)*. International Organisation for Standardisation, 24613:2006 edition, 2006.

[6] M. Rosner, J. Caruana, and R. Fabri. Maltilex: A computational lexicon for maltese. In *In Proceedings of the Workshop on Computational Aspects of Semitic Languages, ACL/COLING98*, pages 97–105, 1998.

# Adapting NLP and Corpus Analysis Techniques to Structured Imagery Analysis in Classical Chinese Poetry

Alex Chengyu Fang
Dept of Chinese, Translation and Linguistics
City University of Hong Kong
Kowloon, Hong Kong SAR
acfang@cityu.edu.hk

Fengju Lo
Dept of Chinese Language and Literature
Yuan Ze University
Taoyuan, Taiwan
gefjulo@saturn.yzu.edu.tw

Cheuk Kit Chinn
Dept of Chinese, Translation and Linguistics
City University of Hong Kong
Kowloon, Hong Kong SAR
manigo@cityu.edu.hk

## Abstract

This paper describes some pioneering work as a joint research project between City University of Hong Kong and Yuan Ze University in Taiwan to adapt language resources and technologies in order to set up a computational framework for the study of the creative language employed in classical Chinese poetry.

In particular, it will first of all describe an existing ontology of imageries found in poems written during the Tang and the Song dynasties ($7^{th}$ –$14^{th}$ century AD). It will then propose the augmentation of such imageries into primary, complex, extended and textual imageries. A rationale of such a structured approach is that while poets may use a common dichotomy of primary imageries, creative language use is to be found in the creation of complex and compound imageries. This approach will not only support analysis of inter-poets stylistic similarities and differences but will also effectively reveal intra-poet stylistic characteristics. This article will then describe a syntactic parser designed to produce parse trees that will eventually enable the automatic identification of possible imageries and their subsequent structural analysis and classification. Finally, a case study will be presented that investigated the syntactic properties found in two lyrics written by two stylistically different lyric writers in the Song Dynasty.

**Keywords**
Imagery, ontology, parsing, lyrics, poetry, Chinese, stylistic analysis, Su Shi, Liu Yong

## 1. Introduction

The language of poetry is different from that employed in other categories of writing. "Defined from a linguistic perspective, poetry represents a variant form of language, different from speech and common writing, unique in its own way as a linguistic system." (Yuan 1989:2 [12]). The difference of poetic language from other types of writing typically exists in its intentionally polysemous readings through the creative use of imageries as part of the poet's artistic conception. Classical Chinese poetry, because of its formal restrictions in terms of syllables, tonal variations and rhyming patterns, commands a language system that appears to be particularly concise, finely rich, highly rhetorical, and thus linguistically complex, requiring a high degree of creativity in writing it, sophisticated interpretation in reading it and often a significant level of difficulty in understanding it.

This article addresses the issue of machine-aided analysis and understanding of classical Chinese poetry in general and attempts to establish a computational framework (cf. Figure 1) within which both inter- and intra-poet stylistic differences and similarities can be usefully investigated with firm grounding in textual analysis from a linguistic perspective. In particular, we believe that the creative language of poetry can be effectively investigated through its manipulation of imageries and through the range of linguistic devices that help to achieve the poetic articulation of the intended artistic ambience.

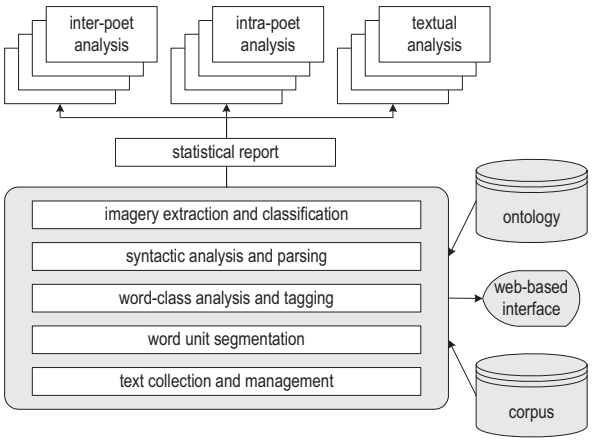



**Figure 1. A computational framework for the computer-aided analysis and understanding of classical Chinese poems.**



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 27–34

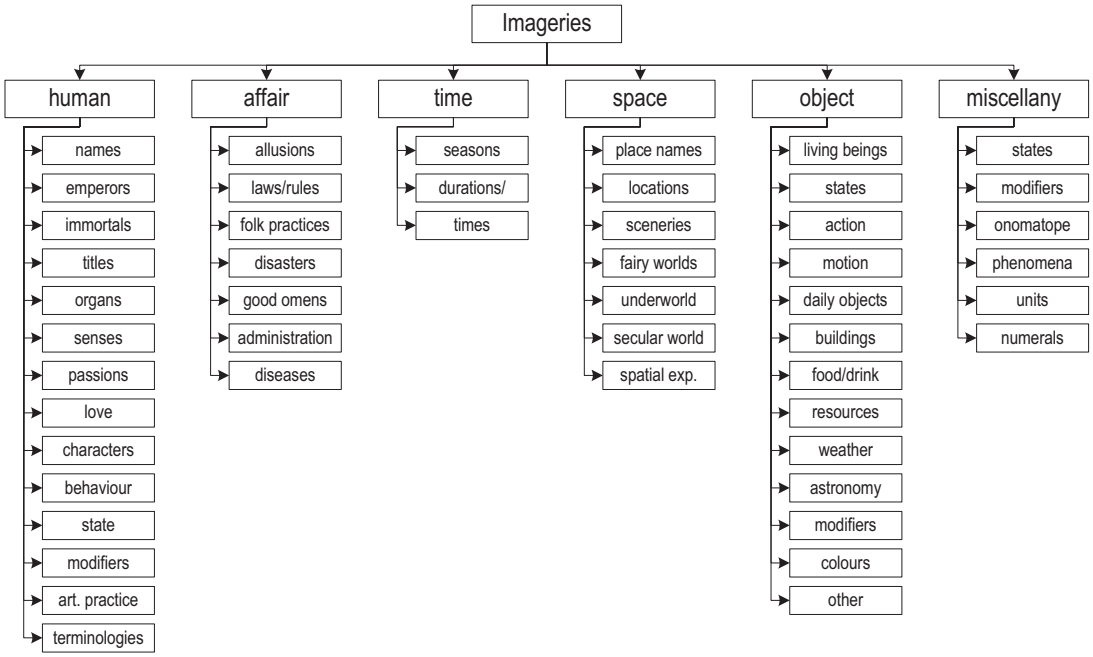



**Figure 2. An ontology of imageries in classical Chinese poems**

This article thus will first of all describe an ontology of imageries that has been created for the poems written during the Tang and the Song dynasties, ranging from the seventh century AD to the thirteenth century AD. It will then propose a new, structured approach towards the extraction and classification of imageries, according to which poetic imageries can be categorised into primary, complex, extended and textual sub-types. Since the automatic processing of imageries in this fashion requires syntactic analysis, we shall then move on to the description of a syntactic parser that provides a structured description of the syntactic constituents for classical Chinese poetry. We finally present a preliminary syntactic analysis of two contemporary poets from the Song dynasty in support of a syntax-based approach to the processing and understanding of classical Chinese poetry.

## 2. An Ontology of Imageries

Lo (2008 [8]) describes an ontology of imageries designed for the study of classical Chinese poetry. The complete collection of the poems from the Tang dynasty was processed at the lexical level. The collection comprises 51,170 poems by 2,821 poets totaling 4,767,979 characters. Individual characters were segmented into meaningful word units (WUs) before WUs were indexed according to their semantic class. A synset was created for synonymous WUs and each synset was described by a keyword. For example, "一年" (one year) is the key word for the following five variant synonymous WUs forming the synset:

| | |
|---|---|
| 一年 | 一載、一歲、一春、一年、一秋 |

Six classes are constructed: human, affair, time, space, object, and miscellany. See Figure 2. Each item in Figure 2 is categorised further into subcategories which will eventually include the actual words and expressions found in a poem. Human/characters, for example, is subdivided into positive moral characters and negative moral characters. As another example, object/astronomy is subdivided into sun, moon, star, sky, etc. Thus, in addition to the six pandects, the system notes 54 subclasses with a further 372 subdivisions before reaching the terminal classes comprising the actual WUs indexed from the complete collection of poems. The system is now available from http://cls.hs.yzu.edu.tw/tang/Database/index.html.

As an example, a search for imageries involving the use of the concept "冬" (winter) in the seasons category would yield 221 records, representing 214 lines from 193 poems by 107 poets. They contain 31 different WUs. Table 1 shows a list of such expressions sorted according to frequency in descending order.

**Table 1. Poetic expressions found in the Complete Collection of Tang Poems involving the imagery *winter*.**

| *Frequency* | *Chinese* | *English* |
|---|---|---|
| 32 | 三冬 | three winters |
| 22 | 冬春 | winter spring |
| 22 | 嚴冬 | harsh winter |
| 20 | 窮冬 | impoverished winter |
| 20 | 經冬 | enduring winter |
| 13 | 無冬 | no winter |
| 6 | 冬夏 | winter summer |
| 6 | 冬雪 | winter snow |
| 6 | 清冬 | clear winter |
| 5 | 冬天 | winter day |
| 5 | 冬盡 | winter end |
| 5 | 冬衣 | winter clothes |
| 5 | 去冬 | previous winter |
| 4 | 一冬 | one winter |
| 4 | 冬來 | winter arriving |
| 4 | 冬寒 | winter cold |
| 4 | 冬景 | winter scene |
| 4 | 凌冬 | cold winter |
| 4 | 過冬 | spend winter |
| 3 | 冬令 | winter season |
| 3 | 冬秋 | winter autumn |
| 3 | 寒冬 | cold winter |
| 2 | 入冬 | entering winter |
| 2 | 冬去 | winter departing |
| 2 | 冬深 | winter deep |
| 2 | 初冬 | early winter |
| 2 | 見冬 | showing winter |
| 1 | 冬初 | winter start |
| 1 | 早冬 | early winter |
| 1 | 殘冬 | remnant winter |
| 1 | 逢冬 | meeting winter |

## 3. A Structured Approach to the Analysis of Imageries

In this article, we propose a more structured approach to imageries than what was described in the previous section. While the ontology remains more or less sufficient for descriptions of classical Chinese poems, the imageries themselves need to be reprocessed to reveal their inner structures. For instance, the imageries involving "冬" (winter) in Table 1 can be analysed into the following according to the role of *winter* within the phrase structure. Consider Table 2.

**Table 2. A structured analysis of winter imageries.**

| $Function_{winter}$ | $Function_{collocate}$ | *Collocate* | *F* |
|---|---|---|---|
| head | modifier | 嚴, 窮, 清, 去, 凌, 寒, 初, 殘, 早 | 64 |
| head | determiner | 三, 無, 一 | 42 |
| complement | verbal | 經, 過, 見, 深, 去, 入, 逢 | 33 |
| modifier | head | 雪, 衣, 天, 景, 寒, 令, 初 | 28 |
| head | coordination | 春, 秋 | 25 |
| subject | verbal | 盡, 來 | 9 |

Table 2 has four columns. $Function_{winter}$ shows the phrase internal function of *winter*, the imagery in question. $Function_{collocate}$ indicates the phrase internal function of the collocates co-occurring with *winter*. *Collocate* lists the actual collocates and *F* the number of occurrence of $Function_{winter}$. The rows are arranged according to *F* in descending order. As can be seen, of the 216 poetic expressions in the Complete Collection of Tang Poems involving the imagery winter, there are seven types of structural analysis, of which modifier+headwinter is the most frequent, occurring 64 times. This structure also has nine different types of instantiations of the modifier as collocates with winter, namely, "嚴" (harsh), "窮" (impoverished), "清" (clear), "去" (previous), "凌" (cold), "寒" (cold), "初" (early), "殘" (remnant), and "早" (early). It is apparent that modifier+headwinter is not only the most significant in frequential terms but also in terms of the variety of its collocates.

It is thus evident that the poetic expressions involving winter could have a more structured and therefore refined representation than what is currently available. We thus propose to distinguish the following types of imageries: primary, complex, extended, and textual.

Primary imageries refer to those head nouns that may have an imagery potential. Winter, for example, is a primary imagery.

Complex imageries refer to primary imageries that have either a premodifier or a determiner. By this definition, "嚴冬" (harsh winter, modifier + head) is a complex imagery and so is "三冬" (three winters, determiner + head).

Extended imageries are defined to include those complex imageries that either serve clausal functions with

an overt subject-verb-object structure or with other syntactic constructs that function as predicates.[1]

Finally, textual imageries are represented in the poem as a system of extended imageries, carefully intended and designed by the poet as part of the artistic conception and articulation.

The four types of imageries thus correspond to four levels of linguistic analysis schematised in Figure 3.

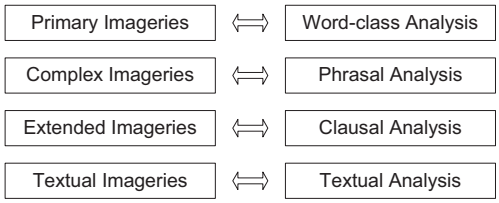



**Figure 3. A schematized correspondence between structured imagery analysis and different levels of linguistic analysis**

The neat correspondence between structured imagery analysis and different levels of linguistic analysis shows that linguistic analysis can be deployed as a stepping stone between poems as raw texts and an ontology of structured imageries derived from the nominal groups. Computation can be performed on the expressions of imageries according to the clausal structure to derive extended imageries. If necessary, techniques for textual analysis can be applied to extended imageries to represent the raw text as a system of interrelated extended imageries.

The structured approach towards the analysis of imageries described in this section will have two immediate applications. The first has to do with the automatic extraction, analysis and classification of imageries, which practically means that ontology generation can be fully automated. The second application lies in the actual analysis of classical Chinese poems. The idea that there is an intrinsic structure within imageries makes it possible to stratify and hence better analyse imageries from lexical, grammatical, syntactic and textual perspectives (cf Figure 3). In the analysis of two poets for inter-poet stylistic differences and similarities, it might be possible that the two poets both make use of a similar set of primary imageries measured in terms of lexical use and semantic grouping. Their creative use of language, which marks them as two different poets or even two distinctive stylistic schools, comes from the creative manipulation of such primary imageries by way of complex imageries, extended imageries and textual imageries.

[1] In contemporary as well as classical Chinese, grammatical categories can be used interchangeably. The syntactic function of a predicate, for instance, can be performed variously by a verb phrase, an adjective phrase, or often a prepositional phrase. See Section 5 for additional information.

# 4. A Syntactic Parser for Classical Chinese Poems

The transfer of nominals to a structured representation of imageries thus requires a syntactic parser of classical Chinese poems. This section describes a parser that represents perhaps the very first effort to analyse poetic lines in a syntactic way.

The current version of the parser is driven by a phrase structure grammar (PSG) for the generation of syntactic trees. Written in Java, it takes two input files, one as a collection of PSG rules and the other containing a poetic line where each component character is tagged with a part-of-speech symbol.[2] It produces all of the possible syntactic analysis for the poetic line permissible by the grammar. Consider the following line from a song lyric (詞, *Ci*) written by Liu Yong (柳永) in the Song Dynasty.

> 殘日下，漁人鳴榔歸去。
> (Under the setting sun, fisher men bang on the boat and leave for home.)

Each character in the above text is POS tagged to yield the following input text where a POS tag is assigned and associated with the character by an underscore:

```
殘_adj 日_n 下_marker ，_punc 漁_n 人_n 鳴_v 榔_n 歸去_v 。_punc
```

A PSG grammar is written in the following manner:

```
S -> PP NP VP
AJP -> adj
NP -> NP n
NP -> AJP n
NP -> n
PP -> NP marker punc
VP -> VP VP
VP -> v NP
VP -> v punc
```

The parser then produces a syntactic tree shown in Figure 4.

[2] The parser is adapted from an earlier version written by Mr Norman Goalby as part of his MSc thesis (Goalby 2004 [6]) supervised by the first author of this article while lecturing at the Computer Science Department, University College London.

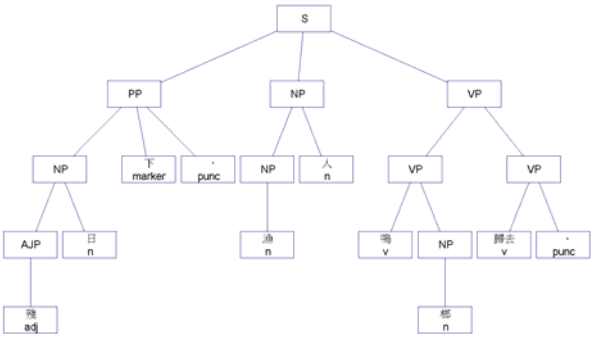



**Figure 4: The syntactic tree automatically produced by the parser for the poetic line "殘日下，漁人鳴榔歸去。".**

Such a tree structure would allow the identification of a set of useful units summarized in Table 3.

**Table 3. Useful imagery units extracted from the parse tree in Figure 4.**

| *Imagery Units* | *Lexical Units* |
|---|---|
| NPs | 殘日 (remnant sun), 漁人 (fisher men), 榔 (boat) |
| Primary imageries | 日 (sun), 漁 (fisher), 人 (men), 榔 (boat) |
| Complex imageries | 殘日 (setting sun), 漁人 (fisher men) |
| Extended imageries | 漁人鳴榔 (fisher men bang on the boat) |

Of course, as is also evident from the sample parse tree, syntactic functions are inferred implicitly. The pre-VP NP is interpreted as the subject of the sentence and the post-VP NP as the direct object. The pre-head NP "漁" (fisher) is interpreted as an NP that functions as a premodifer of the head "人" (men) because of its position within the NP proper. If two nouns are juxtaposed or coordinated within the same NP with or without an overt coordinator, the first conjoin will not be promoted to a NP in its own right. It is thus possible to distinguish between NP as a premodifier and a noun as part of a coordinated construction.

While syntactic relations can be implicitly inferred without much problem for this particular example, we envisage the further development of the current parser into one that not only produces a tree structure labelled by phrasal categories such as NP and VP but will also explicitly indicate the syntactic functions of such phrasal categories such as subject and direct object. A good example can be found in the Survey Parser, which was used to complete the one-million-word International Corpus of English (Fang 1996 [2] and 2000 [53]; Greenbaum 1996 [7]) with a grammar of fine granularity (Fang 2005 [4]).

# 5. A Syntactic Study of Two Lyrics Written in the Song Dynasty

As was mentioned at the end of Section 4, the creative use of language by two poets, which marks them as two different poets or even as belonging to two distinctive stylistic schools, comes from the creative manipulation of primary imageries by way of complex imageries, extended imageries and textual imageries. Naturally, a range of linguistic devices, within the paradigm of syntax, will have to be exploited in order to achieve the intended poetic articulation. For this matter, it is also of great benefit to perform syntactic analysis for extractions of the syntactic relations preferred or typically used by individual poets. This section describes a case study.

Two lyrics (詞, *Ci*) were selected. They are *Yong Yu Yue* (永遇樂. See Appendix A for original text in Chinese) and *Ye Ban Yue* (夜半樂. See Appendix B for original text in Chinese). The two lyrics were written in the Song Dynasty respectively by Su Shi (蘇軾) and Liu Yong (柳永) as representatives of the two stylistically distinctive schools, namely, the *Hao-fang* School (豪放派) and the *Wan-yue* School (婉約派).[3] Su Shi was selected also for a second reason: he also wrote substantially in the wan-yue style. It is thus possible to formulate a comparative framework whereby Shu Shi and Liu Young can be compared in the first instance for their differences and similarities in the manipulation of imageries as representatives of two contrastive stylistic schools. The same procedure can be applied to Shu Shi as a single poet for differences and similarities in his use of imageries across the two stylistically different groups of poems.

Table 4 summarises the two lyrics in terms of tokens, types, and type-token ratios (TTR).

**Table 4. Basic statistics about the two lyrics.**

| *Poet* | *Title* | *Token* | *Type* | *TTR* |
|---|---|---|---|---|
| Su Shi | *Yong Yu Le* | 127 | 87 | 68.5% |
| Liu Yong | *Ye Ban Le* | 167 | 130 | 77.8% |

The TTR for Su Shi is 68.5% while the TTR for Liu Yong is 77.8%, a difference of nearly 10%. The difference seems to indicate different degrees of lexical density but since the two poems have a different number of characters, interpretations here remain inconclusive. Our focus of study will be described in Sections 5.1 and 5.2.

[3] To quote Owen (1996: 582 [9]), "[t]he 'masculine' style was called *hao-fang*, loosely translated as 'bold and extravagant'; the 'feminine' style was called *wan-yue*, something like 'having a delicate sensibility'." The two terms are also translated as *the heroic style* and *the devious-evocative* in Yeh (1998 [10]).

## 5.1 A grammatical analysis of the two lyrics

Both lyrics were segmented and duly analysed by hand at grammatical, phrasal and clausal levels according to a fine-grained formal grammar noting both category types and syntactic functions. Table 5 is a summary of the two lyrics in terms of the four open classes, namely, adjectives, adverbs, nouns and verbs. Column F lists the raw frequency of occurrence for the classes and Column % the proportion of the four classes in all the word tokens.

**Table 5. Summary statistics about the two lyrics' use of adjectives, adverbs, nouns and verbs.**

| | Su Shi | | Liu Yong | |
|---|---|---|---|---|
| *POS* | *F* | *%* | *F* | *%* |
| Adjectives | 21 | 16.5 | 20 | 11.9 |
| Adverbs | 9 | 7.1 | 17 | 6.6 |
| Noun | 41 | 32.3 | 55 | 32.9 |
| Verbs | 13 | 10.2 | 21 | 16.2 |

Grammatically speaking, according to Table 5, the two lyric writers seem to be similar in their use of nouns, which account for 32.3% and 32.9% respectively for the two. Adverbs also seem to show a good degree of similarity between the two poets.

Significant differences between the two writers arise from their use of verbs and adjectives. Liu Yong employs a much higher proportion of verbs, which accounts for 16.2% of his total use of the word tokens. Su Shi, in contrast, has a much smaller proportion of verbs of only 10.2%, 6% lower than the other writer. Regarding adjectives, Su Shi has a more frequent use of adjectives (16.5%) than Liu Yong, whose adjective use has a lower proportion of 11.9%. That Liu Yong has a relative higher use of verbs but lower use of adjectives seems to suggest that this poet is perhaps stylistically more colloquial and more intended for freer vocal rendition by those who liked him. The relatively lower verb proportion and higher adjective use by Su Shi, on the other hand, seems to relate his writing style to one that is more intended for the reading eye and thus more formal, densely expressed and hence scholarly.[4]

## 5.2 A syntactic analysis of the two lyrics

Syntactic properties for the two poets are summarised in Table 6, which lists six syntactic functions: subject, predicate, object, premodifer, adverbial and complement. Each property is described by its raw frequency F and its associated proportion % for the two poets.

**Table 6. Summary statistics about the two lyrics' use of syntactic constructions.**

| | Su Shi | | Liu Yong | |
|---|---|---|---|---|
| *Function* | *F* | *%* | *F* | *%* |
| Subject | 15 | 11.8 | 15 | 8.90 |
| Predicate | 19 | 14.9 | 31 | 18.6 |
| Object | 4 | 3.1 | 1 | 0.59 |
| Premodifer | 18 | 14.2 | 30 | 17.9 |
| Adverbial | 16 | 12.6 | 17 | 10.2 |
| Complement | 1 | 0.79 | 0 | 0.0 |

According to Table 6, Su Shi outperforms Liu Yong in his use of subjects, objects, adverbials and complement. Conversely, Liu Yong outperforms Su Shi in his use of predicates and premodifiers. The pattern of differences here as evidenced by Table 6 seems to suggest that Su Shi deploys a wider range of syntactic devices than Liu Yong, which is also supported by our observations based on grammatical properties summerised in Table 5. Again, a higher proportion of predicates in Liu Yong's writing seems to suggest a more casual, colloquial style while the lower proportion of predicates by Su Shi is accompanied by a relatively higher proportion of objects, thus indicating a preference for SVO constructions, a feature typically found in formal prose. Su Shi's more frequent use of adverbials also seems to confirm this as adverbials have been found to correlate with degrees of formality albeit reported for contemporary British writing. See Fang (2006 [5]) for a detailed report of the empirical study.

That Liu Yong's works tend to be more casual and colloquial is also pointed out by literary critics. As Yeh (2000:1–12 [11]) points out, Liu Yong was a popular song writer at the time and 'every common person by the community water well could sing his songs'. Liu Yong as a lyric writer is innovative in that he abandoned the poetic convention and adopted a more life-like tone and voice in the description and presentation of women in a more realistic style (ibid [11]).

Although only a case study based on very limited evidence, the analysis results presented in this section already speak strongly and favourably for contrastive stylistic analysis of different poets within a framework where linguistic properties at grammatical and syntactic levels can be usefully retrieved and computed. The authors believe that such a computational framework represents a powerful instrument in the automatic analysis of literary texts in general and classical Chinese poems in particular.

# 6. Conclusion

This paper described a computational framework for the analysis of classical Chinese poems. In particular, it presented an ontology of imageries that has been empirically generated from the complete collection of

[4] This is partially evidenced by a recent study reported in Cao and Fang (2009 [1]), which reveals that adjectives tend to occur more frequently in formal academic writing than in informal casual speech though the primary data for the analysis comes from contemporary English.

poems written in the Tang Dynasty. The ontology is now accessible on the Internet and has been a major instrument for the analysis of imageries.

The article then argued for a structured analysis of imageries and proposed a system of imageries sub-categorised as primary, complex, extended and textual. Such a system usefully relates linguistic analysis at lexical, grammatical, syntactic and textual levels to the analysis and evaluation of imageries.

We then described an automatic parser of classical Chinese poems and demonstrated that linguistic analysis could be effectively automated to enable the automatic generation of structured imageries for poetic studies.

Finally, a case study was described that investigated the grammatical and syntactic properties in two lyric poems written by two different poets in the Song Dynasty. The study demonstrated that the two poets as seen in the two lyrics made use of different linguistic devices and that a frequential account of such devices seemed to support literary as well as linguistic interpretations of the two distinctive composing styles.

We conclude by reiterating that such a computational framework represents a useful and insightful instrument in the automatic analysis of literary texts in general and classical Chinese poems in particular.

## 7. Acknowledgments

Research reported in this article was supported in part by research grants (Nos 7002190, 7200120 and 7002190) from the City University of Hong Kong. The authors acknowledge the support received from Prof Shi-wen Yu (俞士汶) at the Institute of Computational Linguistics, Peking University, PR China. The authors also thank Dr Chunshen Zhu at City University of Hong Kong for comments on an earlier draft of the article.

## 8. References

[1] J. Cao and Alex C. Fang. 2009. Investigating Variations in Adjective Use across Different Text Categories. In *Advances in Computational Linguistics, Journal of Research In Computing Science Vol 41*. pp 207-216.

[2] A.C. Fang. 1996. The Survey Parser: Design and Development. In *Comparing English World Wide: The International Corpus of English*, ed. by Sidney Greenbaum. Oxford: Oxford University Press. pp 142-160.

[3] A.C. Fang. 2000. From Cases to Rules and Vice Versa: Robust Practical Parsing with Analogy. In *Proceedings of the Sixth International Workshop on Parsing Technologies*, 23-25 February 2000, Trento, Italy. pp 77-88.

[4] A.C. Fang. 2005. Evaluating the Performance of the Survey Parser with the NIST Scheme. In *Lecture Notes in Computer Science 3878: Computational Linguistics and Intelligent Text Processing*, ed. by A. Gelbukh. Berlin Heidelberg: Springer-Verlag. pp. 168-179.

[5] A.C. Fang. 2006. A Corpus-Based Empirical Account of Adverbial Clauses across Speech and Writing in Contemporary British English. In *LNCS 4139/2006: Advances in Natural Language Processing.* Berlin and Heidelberg: Springer. pp 32-43.

[6] N. Goalby. 2004. *Using Instance-Based Learning to Supplement Early Parsing*. MSc thesis, Computer Science Department, University College London.

[7] S. Greenbaum. 1996. *Comparing English World Wide: The International Corpus of English*. Oxford: Oxford University Press.

[8] F. Lo. 2008. The Research of Building a Semantic Cetegory System Based on the Language Characteristic of Chinese Poetry. In *Proceedings of the 9th Cross-Strait Symposium on Library Information Science*, 3–6 July 2008, Wuhan University, China.

[9] S. Owen. 1996. *An Anthology of Chinese Literature: Beginnings to 1911*. New York and London: W.W. Norton & Company, Inc.

[10] C.-Y. Yeh. 1998. Ambiguity in the Song Lyric: The Influence of the Female Voice in Huajian Songs. In *Studies of Chinese Poetry*, ed. by James Hightower and Chia-ying Yeh. Harvard: Harvard University Asian Center.

[11] C.-Y. Yeh. 2000. *Liu Yong and Zhou Bang-yan*. Taiwan: Daan Press.

[12] X.-P. Yuan. 1989. *Research in the Art of Poetry*. Taiwan: Wunan Book Publishing Co.

## Appendix A. *Yong Yu Yue* by Su Shi

蘇軾《永遇樂》

徐州夜夢覺，北登燕子樓作。

明月如霜,
好風如水,
清景無限。
曲港跳魚,
圓荷瀉露,
寂寞無人見。
紈如三鼓,
鏗然一葉,
黯黯夢雲驚斷。
夜茫茫,
重尋無處,
覺來小園行遍。

天涯倦客,
山中歸路,
望斷故園心眼。
燕子樓空,
佳人何在,

空鎖樓中燕。
古今如夢,
何曾夢覺,
但有舊歡新怨。
異時對、
黃樓夜景,
為余浩歎。

## Appendix B. *Ye Ban Yue* by Liu Yong

柳永《夜半樂》

凍雲黯淡天氣,
扁舟一葉,
乘興離江渚。
渡萬壑千巖,
越溪深處。
怒濤漸息,
樵風乍起,
更聞商旅相呼。
片帆高舉。
泛畫鷁、
翩翩過南浦。
望中酒旆閃閃,
一簇煙村,
數行霜樹。
殘日下,
漁人鳴榔歸去。
敗荷零落,
衰楊掩映,
岸邊兩兩三三,
浣沙遊女。
避行客、
含羞笑相語。

到此因念,
繡閣輕拋,
浪萍難駐。
歎後約丁寧竟何據。
慘離懷,
空恨歲晚歸期阻。
凝淚眼、
杳杳神京路。
斷鴻聲遠長天暮。

# Cross-lingual Adaptation as a Baseline: Adapting Maximum Entropy Models to Bulgarian

Georgi Georgiev
Ontotext AD
135 Tsarigradsko Chaussee
1784, Sofia
Bulgaria
*georgi.georgiev@ontotext.com*

Preslav Nakov
Department of Computer Science
National University of Singapore
13 Computing Drive
Singapore 117417
*nakov@comp.nus.edu.sg*

Petya Osenova and Kiril Simov
Linguistic Modelling Laboratory
Institute for Parallel Processing
Bulgarian Academy of Sciences
25A Acad. G. Bonchev St., 1113 Sofia, Bulgaria
{*petya,kivs*}@*bultreebank.org*

## Abstract

We describe our efforts in adapting five basic natural language processing components to Bulgarian: sentence splitter, tokenizer, part-of-speech tagger, chunker, and syntactic parser. The components were originally developed for English within OpenNLP, an open source maximum entropy based machine learning toolkit, and were retrained based on manually annotated training data from the BulTreeBank. The evaluation results show an $F_1$ score of 92.54% for the sentence splitter, 98.49% for the tokenizer, 94.43% for the part-of-speech tagger, 84.60% for the chunker, and 77.56% for the syntactic parser, which should be interpreted as baseline for Bulgarian.

## Keywords

Part-of-speech tagging, syntactic parsing, shallow parsing, chunking, tokenization, sentence splitting, maximum entropy.

## 1 Introduction

Nowadays, the dominant approach in natural language processing (NLP) is to acquire linguistic knowledge using machine learning methods. Other approaches, e.g., using manual rules, have proven to be both time-consuming and error-prone. Still, using machine learning has one major limitation: it requires manually annotated corpora as training data, which can be quite costly to create. Fortunately, for Bulgarian such a rich resource already exists – the BulTreeBank[1], an HPSG-based Syntactic Treebank with rich annotations at various linguistic levels. The existence of such a resource makes it possible to adapt to Bulgarian various NLP tools that have been originally developed for other languages, e.g., English, and that have been trained on similar kinds of resources, e.g., the Penn Treebank [4].

In this paper, we further stipulate that language adaption should be no harder than domain adaptation [2]. Similarly to Buyko & al. [2], we experiment with the OpenNLP tools[2] since they are open source and contain several platform-independent Java implementations of important NLP components. Moreover, these tools are based on a single machine learning algorithm, maximum entropy (ME) [1], as implemented in the OpenNLP MaxEnt[3] Java package. In our experiments below, we focus on five basic components from the OpenNLP tools: sentence detection, tokenization, part-of-speech (POS) tagging, chunking, and parsing.

Maximum entropy models search for a distribution $p(x|y)$ that is consistent with the empirical observations about a particular feature $f(x, y)$, computed from a set of training examples $\mathcal{T} = \{x, y\}$, e.g., a sentence $x$ and its labeling $y$; see [6] for details. From all such distributions, the one with the highest entropy is chosen [1]. It can be shown that the resulting distribution will have the following form:

$$p_w(y|x) \propto \exp(w \cdot f(x, y)) \quad (1)$$

The features used in the OpenNLP framework combine heterogeneous contextual information such as words around the end of a sentence for the English sentence splitter, or word, character $n$-grams and part-of-speech tag alone and in various combinations for the English chunker. These features are based on the publications of Sha and Pereira [7] for the chunker, and on the dissertation of Ratnaparkhi [6] for the POS tagger and the syntactic parser.

The remainder of this paper is organized as follows: Section 2 describes the process of converting the BulTreeBank XML data to Penn Treebank-style bracketing, Section 3 describes the experiments and discusses the results, and Section 4 concludes and suggests directions for future work.

[1] Created at the Linguistic Modelling Laboratory (LML), Institute for Parallel Processing, Bulgarian Academy of Sciences. See `http://www.bultreebank.org` for details.

[2] `http://opennlp.sourceforge.net`

[3] `http://maxent.sourceforge.net`



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 35–38

## 2 Converting the BulTreeBank XML to Penn Treebank-style bracketing

Converting the BulTreeBank XML [9, 5] to Penn Treebank-style bracketing format is straightforward, with some exceptions, for which we define custom tools. Consider, for example, the following sentence:

Всички на някакъв етап от живота си сме изправени пред проблеми и предизвикателства .

‘All at some point of life itself we face to problems and challenges.’

We all at some point of our lives face problems and challenges.

It has the following XML structure in BulTreeBank:

```
<S>
 <VPA>
  <VPS>
   <Pron>
    <w ana="Pce-op">Всички</w>
   </Pron>
   <PP>
    <Prep>
     <w ana="R">на</w>
    </Prep>
   </PP>
   <NPA>
    <NPA>
     <A>
      <w ana="Pfa--s-m">някакъв</w>
     </A>
     <N>
      <w ana="Ncmsi">етап</w>
     </N>
    </NPA>
    <PP>
     <Prep>
      <w ana="R">от</w>
     </Prep>
     <N>
      <w ana="Ncmsh">живота</w>
     </N>
     <Pron idref="id1">
      <w ana="Psxto">си</w>
     </Pron>
    </PP>
   </NPA>
   </PP>
   <VPC>
    <V>
     <w ana="Vxitf-r1p">сме</w>
    </V>
    <Participle>
     <w ana="Vpptcv--p-i">изправени</w>
    </Participle>
    <PP>
     <Prep>
      <w ana="R">пред</w>
     </Prep>
     <CoordP>
      <ConjArg>
       <N>
        <w ana="Ncmpi">проблеми</w>
       </N>
      </ConjArg>
      <Conj>
       <C>
        <w ana="Cp">и</w>
       </C>
      </Conj>
      <ConjArg>
       <N>
        <w ana="Ncnpi">предизвикателства</w>
       </N>
      </ConjArg>
     </CoordP>
    </PP>
   </VPC>
  </VPS>
 </VPA>
 <pt>.</pt>
</S>
```

We transform the above XML into the following Penn Treebank-style bracketing structure:

```
((S
 (NP
  (PRP Всички)
 )
 (VP
  (PP
   (IN на)
   (NP
    (NP
     (PRP някакъв)
     (NN етап)
    )
    (PP
     (IN от)
     (NN живота)
     (PP$ си)
    )
   )
  )
  (VP
   (VB сме)
   (VB изправени)
   (PP
    (IN пред)
    (NP
     (NP
      (NNS проблеми)
     )
     (CC и)
     (NP
      (NNS предизвикателства)
     )
    )
   )
  )
 )
 (. .)
```

In the process of transformation, we further apply some simple rules for the coordinations in the BulTreeBank, e.g., "CoordP" and "ConjArg" typically become "NP", and "Conj" becomes a "CC" phrase; see [5] for further details on the syntactic phrase naming conventions of the BulTreeBank and [4] for those of the Penn Treebank. We also remove the outer verb phrases in the BulTreeBank, e.g., the phrases "VPA" and "VPS" in the above example, as required by the Penn Treebank bracketing structure. We further define specific rules for pronouns. For example, the "ana" tag in the BulTreeBank [10] is very important for pronoun phrases of the following kind:

```
<Pron><w ana="P...">....
```

First, in case the fourth position is filled by a "t" and the tag starts with "Ps" as in "Ps*t*"[4], this is a *possessive form* and is part of the NP phrase in the transformation structure. For example:

хубавата ми кола

```
'beautiful-the my-clitic car'

my beautiful car
```

Or:

майка ми

```
'mother my-clitic'

my mother
```

Second, if the tag does not start with "Ps" then the pronoun is part of the verb phrase because it is a personal pronoun.

Finally, if there is no "t" on position four, but there is "l" or "-" instead, we annotate this as an NP (see the example above).

We further reduced the original BulTreeBank tagset [10] to a much smaller one with just 95 tags. In most cases, this meant losing some of the surface morphological forms. For example, in the example sentence above, the word "си" ("our own-clitic") was originally annotated with the tag "Psxt" (pronoun, possessive, reflexive, short form), which was transformed to "PP$" (pronoun, possessive). Similarly, the last word in the example sentence, "предизвикателства" (challenges), had the tag "Ncnpi" (noun, common, neuter, plural, indefinite), which was collapsed to "NNS" (noun, plural). In other cases, the tags were directly transformed, e.g., the word "от" (from) with tag "R" (preposition) was transformed to the tag "IN" (preposition or subordinating conjunction).

# 3 Experiments and evaluation

In our experiments, we used the training and the testing sections of the BulTreeBank [5, 9] without further modifications in order to retrain for Bulgarian the sentence splitter, the tokenizer, the POS tagger, the chunker (shallow parser), and the syntactic parser from the OpenNLP tools. The results are shown in Table 1. The sentence splitter achieved an $F_1$ score of 92.54%. The false positives constituted most of the errors and appeared in complicated sentences rather than at abbreviations of organization names as we expected. For example, the following chunk was recognized as a sentence:

Кой беше този човек?, би запитал той. Така че...

```
Who was that guy?, he would ask. So ...
```

However the actual sentence should have been:

Кой беше този човек?, би запитал той.

```
Who was that guy?, he would ask.
```

Some errors appeared in sentences annotated with direct speech, e.g., the sentence tagger annotated the following piece of text as a sentence:

Наведен напред, Той впери поглед в мрака
и рече: - Да бъде светлина.

```
Inclined forward, he took a look in the dark
 and said: - Let there be light.

Inclined forward, he stared at the gloom
 and said: - Let there be light.
```

However, the actual sentence in the BulTreeBank was the following one:

Наведен напред, Той впери поглед в мрака
и рече:

```
Inclined forward, he took a look in the dark
and said:

Inclined forward, he stared at the gloom
and said:
```

**The tokenizer** achieved an $F_1$ score of 98.49%. The majority of the errors appeared in sparse abbreviations. For example, the abbreviation for kilograms, "кг.", was frequently tokenized as "кг". The abbreviation for vehicle "horse power", "к.с.", was wrongly tokenized as "к.с". Some errors involved words that contained no space, e.g., "предсказание" (prediction), which were wrongly tokenized as "казание" (an old Bulgarian word that means "statement"). There were also some rare errors in names of people and locations, e.g., the name "Лазаръс" (Lazarus) was tokenized as the Bulgarian name "Лазар" (Lazar), and the city "Казанлък" (Kazanlak) was tokenized as "Казан" (Kazan).

**The POS tagger** achieved an $F_1$ score of 90.34% on the full morpho-syntactic tagset of the BulTreeBank. We made use of the tagger's ability to employ a tag dictionary that has been automatically generated from the training data and used internally to enumerate over a fixed set of possible tags for each word as opposed to allowing all possible tags; this severely limited

[4] Here * stands for a character that is not important for the discussion.

| Sentence splitter | Tokenizer | POS Tagger | Chunker | Parser |
|---|---|---|---|---|
| 92.54 | 98.49 | 94.43 | 84.60 | 77.56 |

**Table 1:** ***The $F_1$ scores (in %) of the OpenNLP components discussed in the study.***

the number of decision options available per word. The majority of the erroneous morpho-syntactic tags were not entirely wrong; rather, only some of their components were incorrect. For example, "чорбаджиите" was wrongly annotated as "noun, common, feminine, plural, definite" while the correct tag was "noun, common, masculine, plural, definite".

```
wrong annotation
‘‘чорбаджиите’’ (gaffer) with POS=Ncfpd

correct annotation
‘‘чорбаджиите’’ with POS=Ncmpd
```

Another common error was the wrong annotation of proper nouns as common nouns. Here is an example for the person name "Странджата" (*Strandzhata*, a literary character):

```
wrong annotation
‘‘Странджата’’ with POS=Ncfsd

correct annotation
‘‘Странджата’’ with POS=Npfsd
```

Another location example was the word "Балканът" (*Balkanut*, i.e., the Balkan mountain):

```
wrong annotation
‘‘Балканът’’ with POS=Ncmsd

correct annotation
‘‘Балканът’’ with POS=Npmsd
```

We further collapsed the original BulTreeBank tagset, which contained 680 morpho-syntactic tags, to a much smaller one, with 95 tags only; see Section 2 for details. The POS tagger with that reduced tagset achieved an $F_1$ score of 94.43% and was used in the experiments on chunking and syntactic parsing described below. For comparison, the best results for English on the Penn Treebank are 97.33% [8], but using a different learning algorithm: bidirectional perceptron.

**The chunker** achieved an $F_1$ score of 84.60%. This result is much lower than the best $F_1$ score for English reported at the CoNLL-2000 chunking competition: 94.13%. However, this comparison should be treated with caution since we did no special adaptation of the features to Bulgarian. We should also note that the ChunkLink[5] script, used at CoNLL-2000, was tailored to the Penn Treebank tagset, and was thus not very suitable to our collapsed BulTreeBank tagset.

**The syntactic parser:** Unfortunately, we were unable to evaluate our parser with the full morpho-syntactic tagset of the BulTreeBank; this would have required coding efforts for some parts of speech, e.g., nouns, that go beyond simple adaptation. On our collapsed tagset, we achieved an $F_1$ score of 77.56%. For comparison, the best parser for English that only uses Penn Treebank data achieves $F_1$=91.4% [3].

[5] `http://ilk.kub.nl/~sabine/chunklink`

# 4 Conclusions and future work

We have described our efforts in adapting five basic NLP components from English to Bulgarian using manually annotated training data from the BulTreeBank. We have further presented the first systematic evaluation of NLP components for Bulgarian on the same dataset and within the same machine learning framework: maximum entropy. The evaluation results have shown an $F_1$ score of 92.54% for the sentence splitter, 98.49% for the tokenizer, 94.43% for the part-of-speech tagger, 84.60% for the chunker, and 77.56% for the syntactic parser, which should be interpreted as baseline for Bulgarian.

In future work, we will improve on the above baseline results, e.g., by adding language-specific features. We further plan to adapt two other maximum entropy based OpenNLP components to Bulgarian: for named entity recognition and for co-reference resolution.

# Acknowledgments

The work reported in the paper is partially supported by the EU FP7-212578 project LTfLL – Language Technologies for Life Long Learning.

# References

[1] A. Berger, S. D. Pietra, and V. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71, 1996.

[2] E. Buyko, J. Wermter, M. Poprat, and U. Hahn. Automatically adapting an NLP core engine to the biology domain. In *Joint BioLINK-Bio-Ontologies Meeting*, pages 65–68, 2006.

[3] E. Charniak and M. Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *ACL'05: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180, Ann Arbor, Michigan, June 2005.

[4] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19, 1993.

[5] P. Osenova and K. Simov. BTB-TR05: BulTreeBank Stylebook. Technical report, BulTreeBank Project Technical Report, 2004.

[6] A. Ratnaparkhi. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. PhD thesis, University of Pennsylvania, 1998.

[7] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *NAACL'03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 134–141, 2003.

[8] L. Shen, G. Satta, and A. Joshi. Guided learning for bidirectional sequence classification. In *ACL'07: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 760–767, Prague, Czech Republic, June 2007.

[9] K. Simov. BTB-TR01: BulTreeBank Project Overview. Technical report, BulTreeBank Project Technical Report, 2004.

[10] K. Simov, P. Osenova, and M. Slavcheva. BTB-TR03: BulTreeBank Morphosyntactic Tagset. Technical report, BulTreeBank Project Technical Report, 2004.

# Mix and Match Replacement Rules

Dale Gerdemann
Department of Linguistics
Universität Tübingen
72074 Tübingen, Germany
*dg@sfs.uni-tuebingen.de*

**Abstract**

A flexible construction kit is presented compiling various forms of finite state replacement rules. The approach is simpler and more declarative than algorithms in the tradition of Kaplan & Kay. Simple constraints can be combined to achieve complex effects, including effects based on Optimality Theory.

## 1 Introduction

Traditional finite state algorithms for compiling replacement rules have a very procedural flavor. The general approach originated by Kaplan & Kay [8] involves steps for introducing markers, constraining the markers in various ways, making the actual replacement, constraining the markers again after the replacement and finally removing the markers. Several variants of Kaplan & Kay's algorithm have been presented ([9], [12], [4]) with the goals being either to improve efficiency or to provide slightly different semantics. In this paper, I present an alternative, more declarative approach. Rather than being based on a sequence of steps that are composed together, this approach is based on intersection. A basic, unconstrained replacement is represented by a transducer, which can then be constrained by intersecting this transducer with automata representing a variety of constraints. The obvious objection to this approach is that transducers are not closed under intersection.[1] It turns out, however, that a rather limited form of intersection is sufficient for implementing the various forms of replacement rules that have appeared in the literature, along with a wide variety of replacement rules that have not appeared in the literature.

One of the advantages of finite state technology is reusability. An implementation designed for one finite state toolbox can easily be redeployed with a different toolbox. At least that is the theory. In practice, toolboxes tend to provide a number of incompatible features. Replacement rules, in particular, occur in many variations: optional/obligatory, left-to-right/right-to-left/simultaneous, longest-matching/shortest-matching, etc. Sometimes a rather small variation has required introducing some new syntax. The widely used Xfst system [1], for example, has special syntax (dotted brackets) used to enforce a constraint against repeated epenthesis. While this is certainly a useful option, it is also an impediment to users who wish to transfer their Xfst implementation to another toolbox. Clearly such users would benefit more from a kind of replacement rule construction kit, allowing such variant replacement rules to be put together as needed. But the procedural style of Kaplan & Kay-inspired algorithms makes the development of such a construction kit very difficult. Given the declarative approach developed in this paper, the construction kit approach can be implemented in any standard finite-state toolkit.

The declarative approach to replacement rules is very flexible, allowing for example, variants of finite-state Optimality Theory ([2] [5]) to be incorporated into the rules. In some sense, this is not new, as Karttunen's approach to longest match replacement [9], was already a kind of optimization. The declarative, mix-and-match approach, however, allows for the easy incorporation of various optimality constraints. For example, a constraint against repeated epenthesis could be loosened to allow the minimal amount of epenthesis consistent with some other constraints.[2]

## 2 Flat Representation for Transducers using Types

The general form for a finite-state rewrite rule is as in 1:

$$x \rightarrow T(x)/\lambda \_ \rho \tag{1}$$

This is understood as a rule for replacing the string $x$ with the transduction $T(x)$ in the specified context, with some unspecified mode of operation such as obligatory left-to-right. Traditionally $T$ is specified as a cross-product of regular languages, but as discussed in Gerdemann & van Noord [4], this is by no means necessary.

In order to develop an approach based on intersection, a rule such as 1 (ignoring context, and other constraints) will first be transformed into a finite state acceptor (FSA). There is of course a standard trick of using same-length transducers, which goes back to the early days of finite-state morphology [11]. A same-length transducer can be encoded as an FSA with

[1] More precisely, regular relations are not closed under intersection. It is convenient, however, to systematically confuse finite state transducers (respectively, finite state acceptors) with the regular relations (respectively, regular languages) that they encode.

[2] This paper concentrates on some fairly simple examples that are good for expository purposes. For further examples, see: www.sfs.uni-tuebingen.de/∼dg/mixmatch.



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 39–47

transitions labeled with pairs of symbols, or equivalently by an FSA in which odd-numbered transitions (from the start state) count as inputs, and even numbered transitions count as outputs. This is, however, an inconvenient representation for two reasons. First, we often work with transducers that do not have the same-length property. And second, we would like our flattened transducer representations to contain other symbols for markup which count as neither input nor output. So rather than force things into a same-length mold, we use the following very general definition for a flat representation of a transducer:[3]

**Definition 2.1** *A finite state automaton* ***A*** *is a* ***flat representation*** *of a transducer* ***T*** *with respect to transducers* ***extract_input*** *and* ***extract_output*** *if:*

$$T = extract\_input^{-1} \circ identity(A) \circ extract\_output$$

It should be understood here that the equals sign means that the transducers defined on either side represent the same regular relation,

This flat representation is the central feature of the approach. This approach allows *extract_input* and *extract_output* to be defined in many different ways. To instantiate these extraction transducers in a useful and flexible way, I start by introducing a type system with types that these transducers can refer to.

## 2.1 Types

The goal of introducing types is to make it easy to define *extract_input* and *extract_output*. If some symbols are of the input type, for example, then these are the ones that should be extracted by *extract_input*. A simple idea would be to use different alphabets for these two types. In some cases, this is a convenient idea. For example, if the input alphabet is Cyrillic and the output alphabet is Latin, then it is easy to distinguish input and output symbols. In most cases, however, this is not practical. Therefore an alternative approach is adopted: every untyped symbol is followed by a *symbol type indicator*, which specifies the type of its predecessor.

At a minimum, there need to be two symbol type indicators: one for input symbols and one for output symbols. More usefully, there should also be indicators for symbols that are neither input nor output (marker symbols) and also for symbols that are both input and output (identity symbols). Possibly, for some algorithms, it would be useful to have a several types of marker symbols, but for now four symbols will be enough.

- define input a;
- define output b;
- define identity c;
- define marker x;
- define symbolTypeIndicator
  [input ∪ output ∪ identity ∪ marker];

[3] It is nevertheless possible using a two-level approach to work out ideas similar to those in this paper. See Yli-Jyrä [15].

Note that the self-explanatory syntax here is as in the Foma system (Hulden [6]). The particular choice of symbols used here is arbitrary and can easily be changed,

Extended alphabet symbols thus consist of a sequence of two symbols, where the second symbol determines the type of the first symbol. With the preceding conventions, for example, the sequence [z a] (= [z input]) represents 'z' as an input symbol, and [< x] (= [< marker]) represents the angle bracket '<' used as a marker symbol. For the purpose of illustration with a running example, a small test alphabet is defined here (even though Foma does not require the alphabet to be specified). In these definitions, *xsig* should be understood as "extended sigma."

- define sigma [a ∪ b ∪ c ∪ x]; # small test alphabet
- define xsig [sigma symbolTypeIndicator];

An approach using sequences of two symbols as the basic unit can be awkward to work with and painful to debug. Thus it is essential to provide a set of tools that are specialized to work with this approach.

### 2.1.1 Matchers, Constructors and Accessors

Three basic tools for working with an extended alphabet are *matchers*, *constructors* and *accessors*. Generally, these tools are designed to hide the symbol type indicators from the user.

**Matchers** Matchers are a set of defined regular expressions for matching the various types of symbols.

- define in [sigma [input ∪ identity]];
- define inx [sigma input];
- define out [sigma [output ∪ identity]];
- define outx [sigma output];
- define ident [sigma identity];

Note that the 'x' in *inx* and *outx* should be read as "except identity." Also note that these definitions use the toy alphabet *sigma*. For general use, one should replace this with Σ, which is the Foma symbol for an open alphabet.

**Constructors** Constructors are used to turn an ordinary alphabet symbols into typed symbols of the extended alphabet. For this purpose, parameterized definitions are used.

- define in(x) [x [input ∪ identity]];
- define inx(x) [x input];
- define out(x) [x [output ∪ identity]];
- define outx(x) [x output];
- define ident(x) [x [identity]];

**Accessors** Accessors are used to extract either the input side or the output side. By extracting both sides, a definition can be given for *unflatten*. The definitions rely upon a definition of term complement which is specialized for use with the extended alphabet.[4] The first step in *extractInput* eliminates all non-input symbols. This is composed with a transducer that eliminates the symbol type indicators.

- define tcomp(e) xsig-e; # term complement
- define extractInput
  [tcomp(in):$\epsilon$ $\cup$ in]* $\circ$ [$\Sigma$ $\Sigma$:$\epsilon$]*;
- define extractOutput
  [tcomp(out):$\epsilon$ $\cup$ out]* $\circ$ [$\Sigma$ $\Sigma$:$\epsilon$]*;
- define unflatten($\phi$)
  extractInput$^{-1}$ $\circ$ $\phi$ $\circ$ extractOutput;

### 2.1.2 Pretty Printers

Just as in any programming language, it is important to have concise, understandable print representations of complex data structures. Most finite state toolboxes include a graphical interface, which is very helpful for debugging. It is certainly helpful to have such an interface, but experience has shown that networks quickly become way too complicated to be understood visually. Effective debugging is an acquired skill that requires a great deal of ingenuity in breaking problems into units that can be understood as networks. Since the approach of using a typed alphabet doubles the size of the network, it must also at least double the cognitive complexity of understanding the network.[5]

What is needed is a graphical interface that shows the types of symbols in a concise way without using sequences of two symbols, and without using the symbol type indicator. An implementation might, for example, use different colors for the different types. But since such an implementation does not exist, I simply define here an ad hoc pretty printing approach, specialized for use with the small toy alphabet. The idea is that an input 'a' is transduced to 'a:', an output 'a' is transduced to ':a', and an identity 'a' is transduced to 'a'. In all cases, the symbol type indicator is removed.[6]

- define ppInput
  [inx(a):a%: $\cup$ inx(b):b%: $\cup$ inx(c):c%:
  $\cup$ inx(x):x%:];
- define ppOutput
  [outx(a):%:a $\cup$ outx(b):%:b $\cup$ outx(c):%:c
  $\cup$ outx(x):%:x];
- define ppIdentity
  [ident(a):a $\cup$ ident(b):b $\cup$ ident(c):c
  $\cup$ ident(x):x];
- ppIO [ppInput $\cup$ ppOutput $\cup$ ppIdentity];
- define pp(X) [X $\circ$ ppIO*]$_2$;

Further utilities and pretty printers will be introduced as needed. First, however, we should start to see how the type system can be useful for our compilation.

### 2.1.3 Assertions and Boolean Tests

An important invariant of the typed approach is that every sequence is of even length and every symbol in an even position is a symbol type indicator. Consistent use of matchers, constructors and accessors will help to ensure that this invariant holds. Nevertheless, things can go wrong, so it is important to use boolean tests as assertions. Boolean types were introduced into the regular expression calculus in van Noord & Gerdemann [14], though the details are worked out differently here. The central idea is similar to conventions in a variety of programming languages, where certain designated values are understood as, or coerced to boolean values. In the finite state domain, it is convenient let $\epsilon$ be `true`, and $\emptyset$ be `false`.

- define true $\epsilon$;
- define false $\emptyset$;

Note that with these conventions, `true` is a single-state FSA where the one state is both initial and final, and `false` is a single-state FSA where the one state is non-final.[7]

Given these conventions, the boolean connectives *and*, *or* and *not* can be defined as concatenation, union and complementation, respectively.

- and(B1, B2) B1 B2;
- or(B1, B2) B1 $\cup$ B2;
- not(B) $\neg$B $\cap$ true;

[4] The expression *tcomp(in):$\epsilon$ $\cup$ in* could be written more explicitly as *tcomp(in):$\epsilon$ $\cup$ identity(in)*. It is, however, standard in the literature to let language expressions be coerced to identity relation expressions when the context requires this interpretation.

[5] Besides raising the cognitive complexity, the typed approach also clearly increases the computational complexity. But this is unlikely to be a practical concern. The complexity of finite state implementations does not, in any case, derive from the complexity of individual replacement rules. Normally, complexity derives from composing a long sequence of such replacement rules.

[6] Note that the percent sign is used in Foma to escape special characters. So 'a%:' is the two-character symbol consisting of an 'a' followed by a colon. Note also that the subscript 2 in the last definition is used to extract the second projection (also known as "range" or "lower language").

[7] In Foma, it is easy to see if assertions have succeeded or not from the compiler output. For example, with a source file (assert.fom) containing the following two lines

```
define assertion1 ϵ;
define assertion2 ∅;
```

the compiler produces the following output

```
foma[0]: source assert.fom
Opening file 'assert.fom'.
defined assertion1: 136 bytes. 1 states, 0 arcs, 1 path.
defined assertion2: 136 bytes. 1 states, 0 arcs, 0 paths.
```

The information that the compiled version of *assertion1* has 1 path is sufficient to see that this assertion succeded.

For an example using a lot of assertions, see www.sfs.uni-tuebingen.de/∼dg/mixmatch/FinnOTMatching.fom. The FinnOTMatching program is also a good illustration of finite state OT in general.

With these connectives, basic decidable properties of FSA's can be defined.[8]

- empty(L) not([L:true]$_2$);
- subset(A1,A2) empty(A1 ∩ ¬A2);
- equal(A1, A2)
  and(subset(A1,A2), subset(A2,A1));

Using these tools, the following assertions can be defined.

- define evenLength(L) empty(L ∩ [[Σ Σ]* Σ]);
- define indicatorsInEvenPositions(L)
  subset([[[Σ Σ]* Σ]:ϵ Σ Σ*:ϵ]$_2$,
  symbolTypeIndicator);

# 3 Basic Unconstrained Replacement

To compile a replacement rule as in (1), we must start by flattening the transducer $T$. We have already seen the *unflatten* definition, and it is clear that *flatten* should be the inverse of this, so that for transducer $T$:

$$T \equiv unflatten(flatten(T)) \quad (2)$$

If $T$ is a cross-product transducer, then this is easy to define:

- define flattenCross($\phi$, $\psi$ )
  [$\phi$ ∘ [Σ ϵ:input]*]$_2$ [$\psi$ ∘ [Σ ϵ:output]*]$_2$;

Flattening for non-cross product transducers, although not hard, requires access to states and transitions and therefore cannot be done in a portable way.[9]

In general, the flattening step may arbitrarily intersperse input and output symbols, making it impossible to write constraints that refer at the same time to input and output symbols. The general problem is that transducers are not closed under intersection. The good news is that constraints on replacement rules rarely need to refer simultaneously to inputs and outputs.

Given a flattened transducer $T$, we can define a flattened unconstrained replacement rule with $T$ as its center. By analogy with Optimality Theory, this basic replacement rule is called *replaceGen*.

- define lb [a marker];
- define rb [b marker];
- define bracket lb ∪ rb;
- define replaceGen($\phi$) [ident ∪ [lb $\phi$ rb]]*;

Since these rules have introduced a couple of marker symbols, it is important to define corresponding pretty printing definitions.

- define ppBrackets [lb:%< ∪ rb:%>];
- define pp1(X) [X ∘ [ppIO ∪ ppBrackets]*]$_2$;

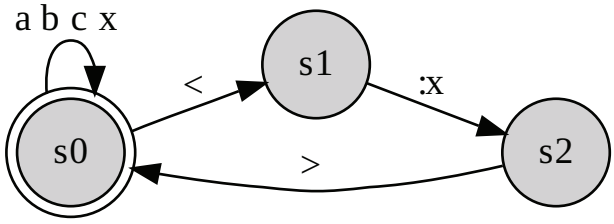



**Fig. 1:** *regex pp1(replaceGen(flattenCross(ϵ, x)))*

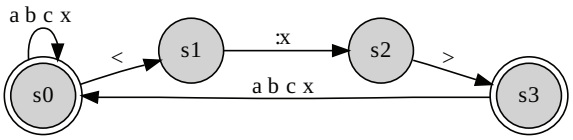



**Fig. 2:** *Non-iterated Epenthesis, flattened version*

Putting these pieces together, we see in Figure 1 the pretty-printed flattened automaton for free epenthesis of 'x'. This automaton looks superficially like an intermediate step in a traditional replacement rule compilation procedure. But in fact, it is considerably different. The brackets, for example, are not inserted as steps in a procedure. Constraints may refer to the brackets, but the brackets are neither in the input nor in the output.

# 4 Constraints

Constraints are of two basic types: strict and optimizing. Contextual constraints are, for example, traditionally treated as strict constraints, and optimizing constraints are typically limited to constraints for leftmost and longest matching (Karttunen [9]). Strict constraints, being easier, are a natural starting point.

## 4.1 Strict Constraints

As a simple example of a strict constraint, consider a constraint against iterated epenthesis. This may seem like a minor constraint, but nevertheless it was considered important enough to merit some special syntax in Xfst.[10] This is easy to define in the mix-and-match approach, using auxiliary definitions for complement and containment. The automaton is shown in Figure 2.

- define comp(E) xsig* - E;
- define contain(E) [xsig* E xsig*];
- define noIterEpenStrict
  comp(contain([lb outx* rb lb outx* rb]));
- regex pp1(replaceGen(flattenCross(ϵ, x))
  ∩
  noIterEpenStrict);

[8] For transducers, Foma provides the boolean tests: *_isfunctional* and *_isidentity*.

[9] See: www.sfs.uni-tuebingen.de/∼dg/mixmatch/flatten for a mix of Foma code and Perl code.

[10] See Beesley & Karttunen [1], p. 67.

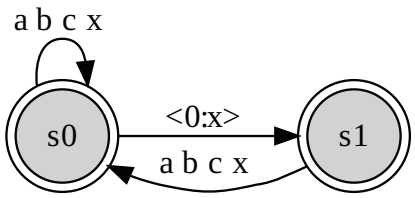



**Fig. 3:** *Non-iterated Epenthesis, unflattened version*

The unflattened version, compiled from the following code, is shown in Figure 3.

- regex unflatten(
  replaceGen(flattenCross($\epsilon$, x))
  $\cap$
  noIterEpenStrict);

# 5 Defeasible Constraints

Now let us relax the constraint against iterated epenthesis and allow iterated epenthesis only when there is no other alternative. Suppose that there is a higher ranking constraint requiring the output to have a sequence of two x's. If the input already has a sequence of two x's, then the output constraint will in any case be satisfied. If the input only has singleton x's, then epenthesis is needed. But iterated epenthesis is not needed. If, however, the input has no x's, then iterated epenthesis is needed. But it is still desirable to do no more iterated epenthesis than necessary. We start, as in OT, by putting a star after every constraint violation. To do this, we need some standard utilities from Kaplan & Kay [8].[11]

- define intro(S) [[$\Sigma$ $\Sigma$] $\cup$ $\epsilon$:S]*;
- define introx(S) [[$\Sigma$ $\Sigma$] $\cup$ $\epsilon$:S]* [$\Sigma$ $\Sigma$];
- define xintro(S) [$\Sigma$ $\Sigma$] [[$\Sigma$ $\Sigma$] $\cup$ $\epsilon$:S]*;
- define ign(L, S) [L $\circ$ intro(S)]$_2$;
- define ignx(L, S) [L $\circ$ introx(S)]$_2$;
- define xign(L, S) [L $\circ$ xintro(S)]$_2$;
- define ifPThenS(P,S) comp([P comp(S)]);
- define ifSThenP(P,S) comp([comp(P) S]);
- define PIffS(P,S) ifPThenS(P,S) $\cap$ ifSThenP(P,S);

Using these definitions, we can introduce a star, as in OT, and constrain it to occur after constraint violations.

- define star [c marker];
- define ppStar star:%*;
- define pp2(X)
  [X $\circ$ [ppIO $\cup$ ppBrackets $\cup$ ppStar]*]$_2$;
- define markViolation($\tau$, $\pi$)
  ign($\tau$, star)
  $\cap$
  PIffS([xsig* ignx($\pi$,star)], [star xsig*]);

Then we can mark the violation. The result is shown in Figure 4.

- regex pp2(markViolation(
  replaceGen(flattenCross($\epsilon$, x)),
  [lb outx* rb lb outx* rb]));

The next step involves minimizing the number of stars. We want to rule out the case that an input would be mapped to different outputs, $b$ and $w$ (for "better" and "worse"), where output $b$ contains $n$ stars, output $w$ contains $m$ stars, and $m > n$. The general idea is to construct a *worsening transducer* that turns better candidates into worse ones. Then, given a set of candidates $c$, the optimal candidates will be $c - worsen(c)$. This assumes, of course, that the worsening of a candidate is complete, all worse candidates are included and no non-worse candidates are included. In some cases, this relation can only be approximated.

There are two kinds of worsening: star-based worsening, which is the main focus of this paper, and generalized worsening. The difference is that star-based worsening works by adding stars to a candidate, whereas generalized worsening directly manipulates the output for a given input to make it worse. In either case, worsening is used to filter out bad candidates by comparison with alternative candidates for the same input. This means that GEN cannot actually change the input, GEN can only add *markup symbols* into the input, where markup symbols are disjoint from input symbols.[12]

For *replaceGen*, the obvious markup symbols are the brackets and the output symbols. But what about the identity symbols? Here, an input symbol $[\Sigma, input] = [\Sigma, a]$ is "marked up" as $[\Sigma, identity] = [\Sigma, c]$. To deal with this problem, the *changeMarkup* step is written to allow identity symbols into non-identity input symbols, and vice versa.

- define changeIdentities
  [[$\Sigma$ identity:input] $\cup$ [$\Sigma$ input:identity] $\cup$ xsig]*;
- define changeOutputs
  [outx:$\epsilon$ $\cup$ $\epsilon$:outx $\cup$ xsig]*;
- define changeBrackets
  [bracket:$\epsilon$ $\cup$ $\epsilon$:bracket $\cup$ xsig]*;
- define changeMarkup(X) [
  X
  $\circ$ changeIdentities
  $\circ$ changeOutputs
  $\circ$ changeBrackets]$_2$;

[11] As a useful convention, 'x' at the beginning means "except at the beginning," and similarly for 'x' at the end. The abbreviation *ign* is used for ignore.

[12] So GEN can only change the input indirectly by adding markup symbols that are interpreted as editing instructions. For example, if GEN brackets syllables, then unsyllabified parts of the input may be deleted in a later (phonetic) module. Other kinds of editing interpretations for markup can easily be invented.

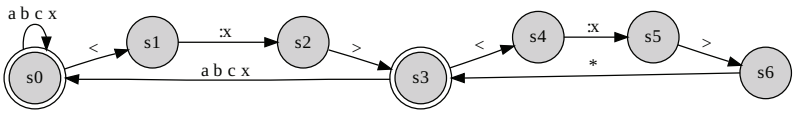



**Fig. 4:** *No iterated Epenthesis with violations starred*

Returning now to our running example, suppose that we add a higher ranked constraint which in some cases can only be satisfied by use of iterated epenthesis. Specifically, let us require that the output must contain a sequence of two instances of 'x'. Since this is a strict constraint, it can be applied by intersection. The result is shown in Figure 5.

- define containsOutputXXGen
  replaceGen(flattenCross($\epsilon$, x))
  $\cap$
  contain([out(x) tcomp(out($\Sigma$))* out(x)]);

We then mark violations of the no iterated epenthesis constraint, with the result shown in Figure 6.

- regex pp2(markViolation(
  containsOutputXXGen,
  [lb outx* rb lb outx* rb]));

The automaton in Figure 6 is fairly complicated. After changing markup, however, it is greatly simplified, since constraints on correct ordering of output are abstracted away. The result is shown in Figure 7.

- regex pp2(changeMarkup(markViolation(
  containsOutputXXGen,
  [lb outx* rb lb outx* rb])));

Figure 7 shows that every candidate has one of three things.

1. an identity x, or
2. an output-only x, or
3. a star.

If we now add in a positive number of stars, we obtain an automaton that matches every non-optimal candidate as seen in Figure 8.

- regex
  pp2(addStars(changeMarkup(markViolation(
  containsOutputXXGen,
  [lb outx* rb lb outx* rb]))));

In general, however, just adding stars is not sufficient to obtain an automaton that matches every non-optimal candidate. A simple example suffices to illustrate the problem. Suppose that there are two candidates, where the better one is $ab*c$ and the worse one is $a*bc*$. Here it is not possible to add a star to the better candidate to match the worse candidate. The issue is that stars need to be not only added but also moved around in order to match up. Here it is in general not possible using finite state methods, to obtain all possible placements of the stars. So various levels of approximation must be defined.

- define permuteStarLeft
  [$\epsilon$:star tcomp(star)* star:$\epsilon$];
- define permuteStarRight
  [star:$\epsilon$ tcomp(star)* $\epsilon$:star];
- define permuteStars
  [xsig* [permuteLeft $\cup$ permuteRight]]* xsig*;
- define permute0(X) $X_2$;
- define permute1(X) [X $\circ$ permuteStars]$_2$;
- define permute2(X)
  [X $\circ$ permuteStars $\circ$ permuteStars]$_2$;
- etc

In theory, an unbounded amount of permutation may be necessary in order to optimize all possible inputs. In practice, however, this is only the case in highly artificial examples such as the one constructed by Frank and Satta [2]. The normal case is that only *permute0* and *permute1* are needed. To test how much permutation is needed, one can use the *isFunctional*[13] test described in Gerdemann & van Noord [5].

- define isOptimized(Starred)
  _isfunctional([Unflatten(Starred)
  $\circ$
  $\Sigma$-star -> $\epsilon$]);

The idea here can be illustrated quite simply. Suppose the input *abc* results in two outputs $xa*ybc*$ and $axb*z$ where $\{x, y, z\}$ is markup. If we delete everything but the stars, then *abc* is mapped to $**$ and $*$. The *isFunctional* test is designed to rule out cases like this. Each input should be mapped to a set of candidates with a unique number of stars. If this condition holds, then the set of candidates has been optimized.

By using the *isFunctional* test, it can be determined that no permutation of stars is necessary for this case. So we can optimize in the following way, where the result is shown in Figure 9.

- define optimizeStars0(StarredCandidates) [
  StarredCandidates
  -
  permute0(
  addStars(
  changeMarkup(StarredCandidates)))];

[13] In Foma, this is called _isfunctional. The return values are `true` and `false` as described in section 2.1.3,

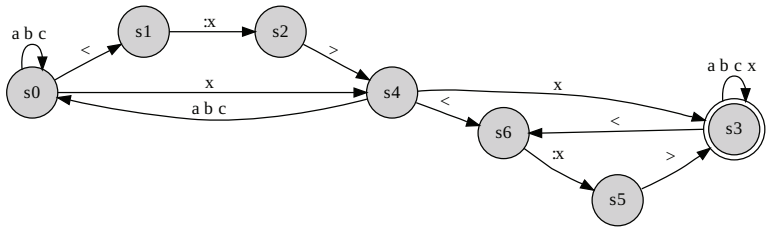



**Fig. 5:** *containsOutputXXGen*

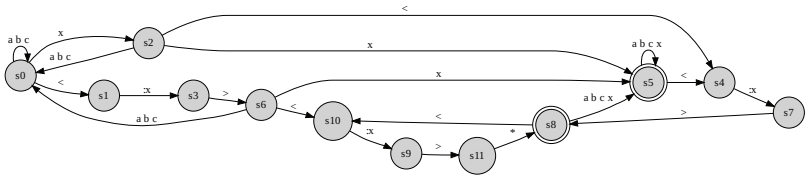



**Fig. 6:** *Starred violations of no iterated epenthesis*

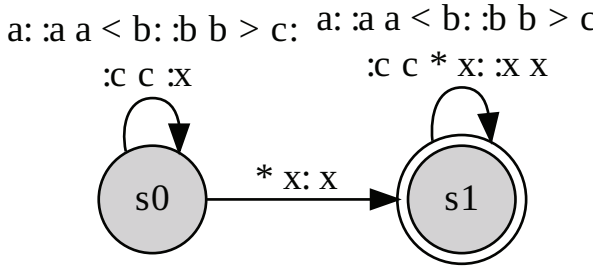



**Fig. 7:** *After changing the markup*

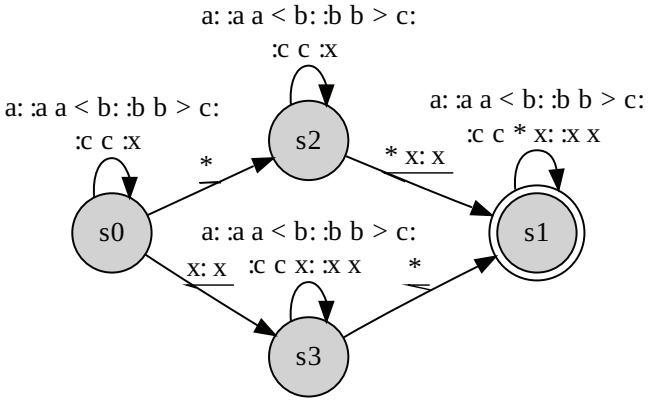



**Fig. 8:** *After adding stars*

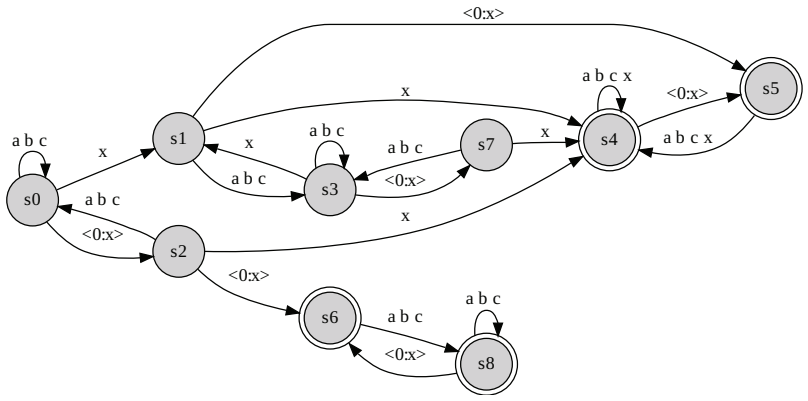



**Fig. 9:** *After optimizing*

- define opimizedRewriteRule
  unflatten(
  optimizeStars0(
  markViolation(
  containsOutputXXGen,
  [lb outx* rb lb outx* rb])));

The transducer in Figure 9 is fairly complex, so it is not obvious that it is correct. The skeptical reader should study some sample input-output such as the following.

- ab ↦ {abxx,axbxx,axxb,axxbx,xabxx,xaxbxx, xaxxb,xaxxbx,xxab,xxabx,xxaxb,xxaxbx}
- ax ↦ {axx,axxx,xaxx,xaxxx}
- axx ↦ {axx,axxx,axxxx,axxxxx,xaxx,xaxxx, xaxxxx,xaxxxxx}

# 6 Conclusion and further directions

It may seem like a lot of work has been expended in order to define a fairly simple replacement rule. But most of this work has gone into preparatory ground work. Further constraints can now be added in a fairly routine way. Certainly, constraints referring to the left and right context are necessary, and now are easy to define. And other, more exotic constraints are not difficult. The user of a toolbox is thus not bound by the particular flavors of replacement rules provided by the toolbox.

For optimizing constraints, I have introduced two kinds of worsening: star-based and generalized. Though this paper has concentrated on star-based worsening, it may well be the case that generalized worsening is even more important. The idea of generalized worsening is that markup can be directly manipulated to make candidates worse. For example, for a longest match constraint, the worsening transducer could manipulate the marked-up matches to make the matches shorter. Or a worsener for a leftmost constraint could manipulate markup to move matches further to the right. It is clear that a worsening transducer should encode an irreflexive, asymmetrical, transitive relation. Although these properties are in general undecidable [7], in practical cases it is usually clear enough that a worsening transducer is well formed.

Generalized worsening is particularly useful for describing prosodic constraints. For example, medieval Indian prosodists introduced following hierarchy of long-short syllable patterns (Singh, [13]), where long syllables count as two positions and short syllables count as one: SSSSS < LSSS < SLSS < SSLS < LLS < SSSL < LSL < SLL.[14] This looks remarkably like some versions of generalized alignment in Optimality Theory. For star-based worsening, the relevant constraint assigns stars to each long syllable, where the number of stars for a long syllable starting in position $i$ is $F_{i+1}$, the $i+1^{st}$ Fibonacci number, and short syllables do not get stars. So with stars added, the hierarchy becomes: SSSSS < L*SSS < SL**SS < SSL***S < L*L***S < SSSL***** < L*SL***** < SL**L*****. Clearly it is not possible to define a transducer to add such a pattern of stars, so the only hope here is to use generalized worsening. The details are a little tricky, but the basic principle is to treat L as a bracketed pair of S's and let worsening move the bracketing to the right or introduce additional bracketing.[15]

Although there may be tricky issues in defining a generalized worsening transducer, the optimization itself is very straightforward.

- define optimize(G, Worsener)
  G - [G ∘ Worsener]$_2$;

To summarize, then, replacement rules have been encoded in a declarative, one-level way, allowing both optimizing and non-optimizing constraints to be mixed and combined in a natural and easy way. In combining OT-style constraints with traditional Generative Phonology-style replacement rules, one may see Kisseberth [10] as a source of inspiration. Long ago, before OT was invented, Kisseberth spoke of rules conspiring to create certain surface effects. The approach presented in this paper is intended to allow such conspiratorial effects to be compiled directly into the replacement rules.

[14] For some issues concerning this pattern, see Gerdemann [3].
[15] See www.sfs.uni-tuebingen.de/∼dg/mixmatch/zeck.fom for details.

# References


[1] K. R. Beesley and L. Karttunen. *Finite State Morphology*. CSLI, Stanford, 2003.

[2] R. Frank and G. Satta. Optimality theory and the computational complexity of constraint violability. *Computational Linguistics*, 24:307–315, 1998.

[3] D. Gerdemann. Combinatorial proofs of Zeckendorf family identities. *Fibonacci Quarterly*, pages 249–262, August, 2008/2009.

[4] D. Gerdemann and G. van Noord. Transducers from rewrite rules with backreferences. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, Bergen Norway, 1999.

[5] D. Gerdemann and G. van Noord. Approximation and exactness in finite state optimality theory. In J. Eisner, L. Karttunen, and A. Thériault, editors, *SIGPHON 2000, Finite State Phonology. Proceedings of the Fifth Workshop of the ACL Special Interest Group in Computational Phonology*, Luxembourg, 2000.

[6] M. Hulden. Foma: a finite-state compiler and library. In *Proceedings of the EACL Demonstrations Session*, pages 29–32, 2009.

[7] J. H. Johnson. Rational equivalence relations. *Theoretical Computer Science*, 47(1):39–60, 1986.

[8] R. Kaplan and M. Kay. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–379, 1994.

[9] L. Karttunen. Directed replacement. In *34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz, 1996.

[10] C. Kisseberth. On the functional unity of phonological rules. *Linguistic Inquiry*, 1:291–306, 1970.

[11] K. Koskenniemi. Two level morphology: A general computational model for word-form recognition and production, 1983. Publication No. 11, Department of General Linguistics, University of Helsinki.

[12] M. Mohri and R. Sproat. An efficient compiler for weighted rewrite rules. In *34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz, 1996.

[13] P. Singh. The so-called Fibonacci numbers in ancient and medieval India. *Historia Mathematica*, 12(3):229–244, 1985.

[14] G. van Noord and D. Gerdemann. An extendible regular expression compiler for finite-state approaches in natural language processing. In O. Boldt, H. Juergensen, and L. Robbins, editors, *Workshop on Implementing Automata; WIA99 Pre-Proceedings*, Potsdam, Germany, 1999.

[15] A. Yli-Jyrä. Transducers from parallel replace rules and modes with generalized lenient composition. In *Finite-state methods and natural language processing*, 2007.

# Enabling Adaptation of Lexicalised Grammars to New Domains

Valia Kordoni & Yi Zhang
DFKI GmbH & Dept. of Computational Linguistics, Saarland University
PO Box 15 11 50, 66041 Saarbrücken, GERMANY
*kordoni,yzhang@coli.uni-sb.de*

**Abstract**

This extended abstract focuses on the main points we will be touching upon during our talk, the aim of which is to present in a concise manner our group's work on enhancing robustness of lexicalised grammars for real-life applications and thus also on enabling their adaptation to new domains in its entirety.

## 1 Introduction

At present, various wide-coverage symbolic parsing systems for different languages exist and have been integrated into real-world NLP applications, such as IE, QA, grammar checking, MT and intelligent IR. This integration, though, has reminded us of the shortcomings of symbolic systems, in particular lack of coverage, one consequence of which relates to enormous and sometimes insurmountable difficulties with porting and re-using such systems to new domains. When the hand-crafted grammars which usually lie at the heart of symbolic parsing systems are applied to naturally occurring text, we often find that they are underperforming. Typical sources of coverage deficiency include unknown words, words for which the dictionary did not contain the relevant category, Multiword Expressions (MWEs), but also more general grammatical knowledge, such as grammar rules and word ordering constraints. Currently, grammars and their accompanying lexica often need to be extended manually.

In this talk, we offer the overview to a range of machine learning-based methods which enable us to derive linguistic knowledge from corpora, for instance, in order to solve problems of coverage and efficiency deficiency of large-scale lexicalised grammars, ensuring this way their portability and re-usability and aiming at domain-independent linguistic processing. In particular, we illustrate and underline the importance of making detailed linguistic information a central part of the process of automatic acquisition of large-scale lexicons as a means for enhancing robustness and ensuring maintainability and re-usability of lexicalised grammars.

To this effect, we focus on enhancing robustness and ensuring maintainability and re-usability for two large-scale "deep" grammars, one of English (ERG; [3]) and one of German (GG; [4]), developed in the framework of Head-driven Phrase Structure Grammar (HPSG). Specifically, we show that the incorporation of detailed linguistic information into the process of automatic extension of the lexicon of such language resources enhances their performance and provides linguistically sound and more informative predictions which bring a bigger benefit for the grammars when employed in practical real-life applications.

## 2 Main Focus Points

In recent years, various techniques and resources have been developed in order to improve robustness of deep grammars for real-life applications in various domains. Nevertheless, low coverage of such grammars remains the main hindrance to their employment in open domain natural language processing. [2], as well as [6] and [7] have clearly shown that the majority of parsing failures with large-scale deep grammars are caused by missing or wrong entries in the lexica accompanying grammars like the aforementioned ones. Based on these findings, it has become clear that it is crucial to explore and come up with efficient methods for automated (Deep) Lexical Acquisition (henceforward (D)LA), the process of automatically recovering missing entries in the lexicons of deep grammars.

Recently, various high-quality DLA approaches have been proposed. [1], as well as [7] and [5] describe efficient methods towards the task of lexicon acquisition for large-scale deep grammars for English and Dutch. They treat DLA as a classification task and make use of various robust and efficient machine learning techniques to perform the acquisition process.

We use the ERG and GG grammars for the work we present in this talk, for the ERG is one of the biggest deep linguistic resources to date which has been being used in many (industrial) applications, and GG is to our knowledge one of the most thorough deep grammars of German, a language with rich morphology and free word order, which exhibits a range of interesting linguistic phenomena. Thus, the aforementioned grammars are valuable linguistic resources since they provide linguistically sound and detailed analyses of phenomena in English and German. Apart from the interesting syntactic structures, though, the lexical entries in the lexicons of the aforementioned grammars also exhibit a rich and complicated structure and contain various important linguistic constraints. Based on our claim above, in this talk we show how the information these constraints provide can be captured and used in linguistically-motivated DLA methods which we have developed. It has been shown that, comparing



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 48–49

to statistically treebank-based parsers, parsers based on these hand-written linguistic grammars have more consistent performance over changing of domains [8]. In this we prove our assumption that the linguistic information we incorporate into our DLA methods is vital for the good performance of the acquisition process and for the maintainability and re-usability of the grammars domain-independently, as well for their successful practical application.

# References


[1] T. Baldwin. Bootstrapping deep lexical resources: Resources for courses. In *Proceedings of the ACL-SIGLEX 2005 Workshop on Deep Lexical Acquisition*, pages 67–76, Ann Arbor, USA, 2005.

[2] T. Baldwin, E. M. Bender, D. Flickinger, A. Kim, and S. Oepen. Road-testing the English Resource Grammar over the British National Corpus. In *Proceedings of the Fourth Internation Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal, 2004.

[3] A. Copestake and D. Flickinger. An open-sourse grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second conference on Language Resources and Evaluation (LREC 2000)*, Athens, Greece, 2000.

[4] B. Crysmann. On the efficient implementation of German verb placement in HPSG. In *Proceedings of RANLP 2003*, pages 112–116, Borovets, Bulgaria, 2003.

[5] T. V. de Cruys. Automatically extending the lexicon for parsing. In *Proceedings of the Student Session of the European Summer School in Logic, Language and Information (ESSLLI)*, pages 180–191, Malaga, Spain, 2006.

[6] G. van Noord. Error mining for wide coverage grammar engineering. In *Proceedings of the 42nd Meeting of the Assiciation for Computational Linguistics (ACL'04), Main Volume*, pages 446–453, Barcelona, Spain, 2004.

[7] Y. Zhang and V. Kordoni. Automated deep lexical acquisition for robust open text processing. In *Proceedings of the Fifth International Conference on Language Resourses and Evaluation (LREC 2006)*, Genoa, Italy, 2006.

[8] Y. Zhang and R. Wang. Cross-domain dependency parsing using a deep linguistic grammar. In *Proceedings of ACL-IJCNLP 2009*, Singapore, 2009.

# QALL-ME needs AIR: a portability study

Constantin Orăsan, Iustin Dornescu and Natalia Ponomareva
Research Group in Computational Linguistics
University of Wolverhampton, UK
*C.Orasan, I.Dornescu2, Nata.Ponomareva@wlv.ac.uk*

## Abstract

Currently access to institutional repositories is gained using dedicated web interfaces where users can enter keywords in an attempt to express their needs. In many cases this approach is rather cumbersome for users who are required to learn a syntax specific to that particular interface. To address this problem, we propose to adapt the QALL-ME framework, a reusable framework for fast development of question answering systems, in order to allow users to access information using natural language questions. This paper describes how the web services part of the QALL-ME framework had to be adapted in order to give access to information gathered from unstructured web pages by the AIR project.

## Keywords

QALL-ME framework, web services, question answering, textual entailment

# 1 Introduction

Currently access to institutional repositories is gained using dedicated web interfaces where users can enter keywords. In many cases this approach is rather cumbersome for users who are required to learn a syntax specific to that particular interface. A solution to this problem is offered by question answering (QA), a field in computational linguistics, which develops systems that take a natural language question and provide the exact answer to it. This paper presents how the QALL-ME framework[1], a reusable framework for fast development of question answering systems, was adapted in order to allow users to access information stored in institutional repositories using natural language questions.

The AIR project [10] developed a system that extracts information about scientific publications from unstructured documents and stores this information in a database. The QALL-ME project [12] has developed a framework for implementing question answering systems for restricted domains. The first implementation of this framework was for the domain of tourism, but it is not in any particular way bound to this domain. For this reason, the QALL-ME framework can offer the ideal and most natural way of accessing the information extracted in the AIR project.

The remainder of the paper is structured as follows: Section 2 presents some background information about the QALL-ME and AIR projects. The domain in which the system is expected to run is presented in Section 3, followed by a description of the framework and how it was adapted to the new domain in Section 4. The paper finishes with conclusions.

# 2 Background information

## 2.1 The QALL-ME project

QALL-ME (Question Answering Learning technologies in a multiLingual and Multimodal Environment) is an EU-funded project with the objective of developing a shared infrastructure for multilingual and multimodal open domain Question Answering.[2] It allows users to express their information needs in the form of multilingual natural language questions using mobile phones and returns a list of ranked specific answers rather than the whole web pages. In the first phase, the tourism domain is highlighted as the domain in which the operates.

Language variability, one of the main difficulties of dealing with natural language, is addressed in QALL-ME by reformulating it as a textual entailment recognition problem. In textual entailment a text (T) is said to entail a hypothesis (H), if the meaning of H can be derived from the meaning of T. To this end, each question is treated as the text and the hypothesis is a procedure to answer the question [6]. This concept is embedded in the QALL-ME Framework [12], one of the main outputs of the project. The purpose of this framework is to provide an architecture skeleton for QA systems that extract answers from structured data sources. This framework is exploited in this paper to provide an access to data collected by the AIR project.

## 2.2 The AIR project

Manual population of institutional repositories with citation data is an extremely time- and resource-consuming process, and usually acts as a bottleneck on the fast growth and update of large repositories. The aim of the AIR project [10] was to develop a semi-automatic approach for archiving institutional repositories. To achieve this, it automatically

[1] The QALL-ME framework is available as an open source project at `http://qallme.sourceforge.net/`

[2] More information about the QALL-ME project can be found at `http://qallme.fbk.eu`



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 50–57

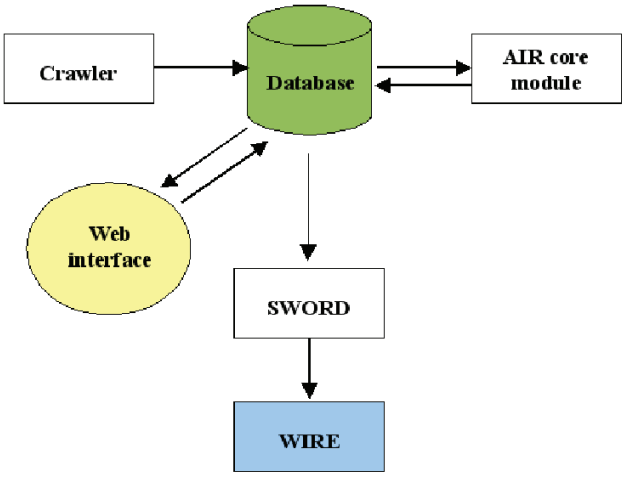



**Fig. 1:** *Overview of the AIR architecture*

discovers and extracts bibliographical data from web sites, and then interacts with users, authors or librarians, who verify and correct extracted data. The components of the AIR system are:

1. A **web crawler** which populates the database with all web pages belonging to the domain under consideration.
2. the **AIR core module** which processes unstructured data (web pages) in order to extract bibliographical references and automatically annotate them with Dublin Core Metadata tags.
3. **Web interfaces** developed for user interaction and data verification. This step was introduced to ensure the reliability of information transferred to the digital repository.
4. Data deposit is accomplished automatically using SWORD[3] protocol.

An overview of the system is presented in Figure 1.

In this section only the information extraction component is briefly presented. The other components are not relevant for the current research; more details about them can be found in [10].

Automatic extraction of bibliographical metadata from unstructured web pages is achieved in three consecutive steps using a machine learning approach:

1. A page classifier extracts web pages containing bibliographies from the whole amount of data provided by the crawler. The classifier exploits the structure of HTML format such as metadata (<keywords>, <title>, <description>) and headers (<h1>, <h2>, etc.) in order to give different weights to distinct data sources. As a machine learning method, we used and compared several methods contained in WEKA [17].
2. A record classifier selects bibliographical entries from all document records using Conditional Random Fields (CRF) [4]. As HTML-format provides some structural elements like tags, we incorporated them into the classifier for revealing enumerations or lists of equivalent records.
3. An information extraction (IE) module identifies 5 types of bibliographical metadata from Dublin Core Metadata Element Set: author, title, date, publisher and citation. As bibliographical reference represents a logical consequence of metadata tags, the use of CRF is the most appropriate.

Given that all three modules focus on similar problems we used very similar features while constructing corresponding classifiers. These features are:

- **Named Entities**, such as PERSONS, LOCATIONS, ORGANIZATIONS and DATES identified using ANNIE[4], a named entity recogniser distributed together with GATE[5] framework [2].
- **Staff names**: A list of all university members was collected and used to annotate the input text.
- **Sherpa/Romeo publishers and journals** the list of publishers and journals stored in the Sherpa/Romeo database was retrieved and used to annotate the input text.
- **Presence of year**: This feature indicates whether an element contains a year.
- **Publication triggers**: We built different lists of triggers that can indicate different types of information about a publication. Examples of such triggers are header indicators (the most frequent words that occur in <h> tags of files containing publications), publication triggers (words appearing in a bibliographical entry itself), citation triggers (words appeared in citation of an entry).
- **Orthographic features**, which capture capitalisation, digits, punctuation marks, etc. This feature was only used for implementation of the IE module.
- **Parts of speech (POS)**. Preliminary experiments revealed many cases when automatically annotated bibliographical fields ended with articles, conjunctions or prepositions. In order to correct this situation, we incorporated parts of speech information into the IE module.

All three modules of the IE component were evaluated on manually annotated data using cross-validation [10]. The page classifier obtained an f-score of 0.882 when using JRIP [1]. The record classifier obtains the best results (f-score of 0.919) with a Markov order 1. The information extraction module can recognise the author, date, title and citation with high accuracy, but perform rather poorly on identification of publisher. Detailed evaluation results can be found in [10].

[3] http://www.ukoln.ac.uk/repositories/digirep/index/SWORD_guide

[4] http://gate.ac.uk/ie/annie.html

[5] http://gate.ac.uk/

# 3 Domain description and modelling

An investigation of the domain of bibliographical references and the data extracted by the AIR project was carried out in order to define which questions can be answered by the system and to model the domain using an ontology. This section describes the domain and ontology used to represent this domain.

## 3.1 The domain

The data stored by the AIR information extraction module in the database contains information about the author(s) of a publication, its title, year of publication, publisher and the rest of the information lumped together as "citation". Using the annotation modules presented in Section 4.2 it is possible to extract from the citation the name of the journal or conference proceeding that published the article. Using automatic term extraction techniques or external databases it is possible to automatically assign keywords to each article. On the basis of this information, the following types of questions are foreseen to be answered by the system:

1. Questions about the name of an author who published in a year, in journal/conference and/or on a topic (e.g. *Which authors had a publication in <JOURNAL> in <YEAR>?*)
2. Questions about the year when an author published in a journal/conference, on a topic or with other authors (e.g. *What year did <AUTHOR> have a paper in <JOURNAL>?*)
3. Questions about the title of a publication by an author, in a journal/conference, on a topic, in a certain year and/or with other authors (e.g. *What papers about <KEYWORDS> did <AUTHOR> publish in <YEAR>?*)
4. Questions about the title of a conference/journal where an author published on a topic, in a certain year and/or with other authors (e.g. *Who published <AUTHOR> in <YEAR>?*)
5. Questions about the topic of an article published by an author in a journal/conference, in a certain year and/or with other authors (e.g. *What are the topics of the papers published in <YEAR> by <AUTHOR> and <AUTHOR>?*)

Each question can have one or several constraints, but none expects the user to specify the title of the publication. This is due to the fact that scientific articles tend to have long titles, which are unlikely to be remembered correctly and completely by a user. For this reason, even with the fuzzy matching implemented in our annotators, it is unlikely that the system can correctly guess what title a user is referring to. On the basis of the five types of questions identified above, 36 types of questions were proposed to be answered by the system.

## 3.2 The ontology

The purpose of the ontology is to provide a conceptualised description of the selected domain and to act as a link between different components of the system and different languages. The ontology developed in QALL-ME for the domain of tourism is described in [9]. Given that the scope of the AIR project consists of academic citations, we could not use this ontology and instead we had to find an ontology which:

- uses standard metadata terminologies such as Dublin Core (dc and dcterms);
- supports the entry types used by the open BibTeX reference management software or other similar schemes;
- allows arbitrary keyword indexing schemes; and
- uses dereferenceable URIs for interoperability with other systems (faceted browsing, semantic web mash-ups)

We decided to use BibTeX as it is very popular in the academic community and it is supported by many citation management systems. Moreover, the format in which AIR stores data can be easily mapped into the BibTeX format. Fields which are not explicitly identified by AIR, such as the name of the proceedings, can be easily extracted using the annotators presented in Section 4.2. In addition, by using this approach, we are not limited to using only the output of the AIR project and we can apply our QA interface to a large number of sources.

The advantage of using BibTeX as the format of the input data is that there are several ontologies that can be used (e.g. the MIT bibtex ontology[6], bibo[7], SWRC[8]. The differences between them are the vocabularies used and details such as author list representation and event representation. We chose to use a subset of the SWRC ontology [14], an ontology for modelling entities of research communities such as persons, organisations, publications (bibliographic metadata) and relationships amongst them. The main entities involved are: persons (authors and editors), organisations (publishers, research institutes, universities), publications (articles, conference papers, theses, book chapters) and collections (proceedings, journals, books, series). A relevant subset of the Dublin Core metadata terminology is used to describe the properties of the bibliographic entries. An example of a conference paper in the TURTLE syntax defined by our ontology can be seen in Listing 1.

[6] `http://zeitkunst.org/bibtex/0.1/`
[7] Bibliographic Ontology Specification `http://bibliontology.com/`
[8] Semantic Web for Research Communities `http://ontoware.org/projects/swrc/`

**Listing 1:** *Example of conference paper presented in TURTLE syntax*

```
qa2:Mitkov1998
rdf:type swrc:InProceedings;
dc:title "Robust pronoun resolution with
     limited knowledge";
terms:issued "1998";
swrc:pages "869-875";
terms:isPartOf qa2:conf/acl/2008;
dc:creator qa2:Mitkov_R_.

qa2:conf/acl/2008
rdf:type swrc:Proceedings;
dc:title "Proceedings of the 18th
    International Conference on
    Computational Linguistics (COLING
    '98)/ACL'98 Conference";
swrc:address "Montreal, Canada".
```

The SWRC terminology is also used by the DBLP[9] (Digital Bibliography & Library Project) computer science bibliography website. This makes it easy to augment the data collected by the AIR project with bibliographic information from other sources. This can be further extended by employing protocols such as Open Archives Initiative (OAI) Metadata Protocol Handler which is an interchange format that facilitates metadata harvesting from electronic repositories or the PRISM[10] protocol. Whilst AIR does not provide us with details such as affiliation relations, this information could be added from such sources, enabling the system to answer questions such as *"Scientists working in which German universities have published papers about Question Answering in 2008?"*.

Using existing software, the data collected by the AIR project in BibTeX format was converted to the RDF format defined by our ontology. For data access the SPARQL query language was used.

### 3.3 Representation of terms

As seen in Section 3.1, a large number of questions that can be asked are restricted by topics. These topics are normally represented using terms and therefore it was necessary to find a convenient way to represent terminologies and relationships between them.

Investigation of existing resources revealed that the `skos` (Simple Knowledge Organisation System) ontology [11] is appropriate as it provides a model for expressing the basic structure and content of concept schemes. A *concept scheme* is defined in the `skos` ontology as "a set of concepts, optionally including statements about semantic relationships between those concepts." (e.g. thesauri, classification schemes, terminologies, glossaries, etc.) The `skos` ontology is useful for our purposes as it encodes relations such as `skos:broader`, `skos:narrower`, `skos:broaderTransitive` and `skos:narrowerTransitive` and allows the asking of questions such as: *"What did Constantin Orasan publish about summarization"?* and the retrieval of papers which were tagged with *multi-document summarization*. A part of the terminology corresponding to computational linguistics can be found in Figure 2.

[9] `http://dblp.uni-trier.de/`
[10] `http://www.prismstandard.org/about/`
[11] (http://www.w3.org/TR/2005/WD-swbp-skos-core-guide-20050510/)

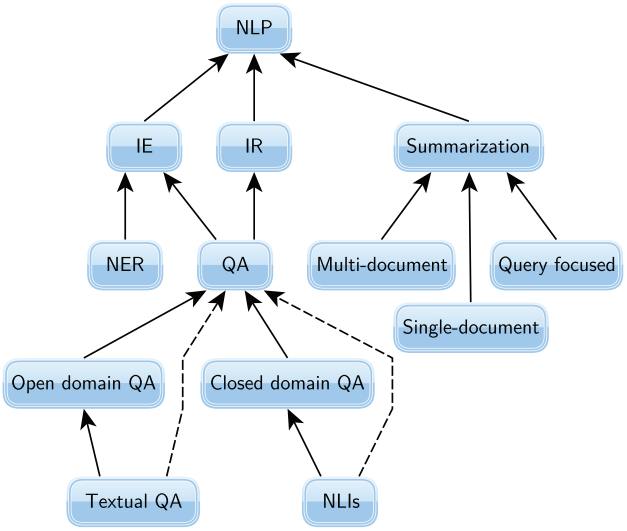



**Fig. 2:** *Test indexing terminology (full lines represent the* `skos:broader` *relation while the dotted lines represent the* `skos:broaderTransitive` *relation)*

## 4 The QALL-ME framework and its adaptation to a new domain

The QALL-ME framework is based on a Service Oriented Architecture (SOA) which, for the domain of tourism, is realised using the following web services:

1. Context providers: are used to anchor questions in space and time in this way enabling answers to temporally and spatially restricted questions
2. Annotators: Currently three types of annotators are available:
   - named entity annotators which identify names of cinemas, movies, persons, etc.
   - term annotators which identify hotel facilities, movie genres and other domain-specific terminology
   - temporal annotators that are used to recognise and normalise temporal expressions in user questions
3. Entailment engine: is used to overcome the problem of user question variability and determine whether a user question entails a retrieval procedure associated with predefined question patterns.
4. Query generator: which relies on an entailment engine to generate a query that can be used to extract the answer to a question from a database.

For the tourism demonstrator the output of this web service is a SPARQL query.

5. Answer pool: retrieves the answers from a database. In the case of the tourism demonstrator the answers are extracted from RDF encoded data using SPARQL queries.

Given that the QALL-ME framework was implemented using a modular approach, in order to adapt the system to a new domain all that was required was to re-implement or adapt some of the existing web services. This section describes each of the web services in more detail emphasising the changes that had to be made in order to adapt the system to the new domain.

## 4.1 Context provider web service

Investigation of the bibliographical domain revealed that the role of this web service is rather limited as the spatial information is not used in questions. For this reason, this web service was reduced to returning the current date and time in order to be able to answer questions with temporal restrictions such as:

*What papers were published in 2009?.*

*What papers were published **this year**?.*

The context information is used by the temporal annotator (described in Section 4.2.3) to normalise the *this year* expression to the TIMEX2 standard.

## 4.2 Annotators

The QALL-ME framework provides the possibility of annotating the input sentences with named entities, terms specific to the domain and temporal expressions. This section presents the annotator services implemented for the bibliographical domain.

### 4.2.1 Named entity and term annotators

Unlike standard Named Entity Recognition that are able to determine unknown and new entities, in restricted domains the entities are known and the annotation task is reduced to a database look-up that needs to deal with spelling mistakes, inaccurate entity references and partial matches. The ability to deal with noisy input becomes even more important for the domain of tourism where users have the possibility of asking questions using speech which means that the system needs to deal with automatic speech recognition errors.

For the domain of tourism we had to recognise named entities such as names of hotels, movies and persons, as well as terms which are multi-word expressions related to the domain such as genre of a movie (e.g. action movie) and site facilities (e.g. disabled access). Both types of expressions are identified using the same greedy algorithm that annotates expressions from a gazetteer, based on a character-based similarity distance between the tokens and an adapted TFIDF score.

The main difference between our method and other approaches, such as [5] and TagLink [13], is that it distinguishes between high probability similar tokens and tokens that are most probably distinct. Using a character-based similarity threshold, we compute a partial one to one matching and evaluate the negative impact of the unmatched tokens. The more words that are matched, the greater the confidence that the two strings represent the same entity, while a great number of mismatched words means that the two strings represent different entities. While matching tokens a certain amount of spelling variations and mistakes are allowed by using the character-based similarity measure proposed by Jaro-Winkler [16].

For the bibliographical domain the same approach was employed, but different types of entities had to be annotated. This was achieved by training the algorithm with different gazetteers. Given the nature of the domain, a large number of ambiguities were noticed. These ambiguities are discussed in the next section.

### 4.2.2 Ambiguities at the level of entity annotation

One of the main challenges we had to address when we ported the annotators to the new domain was that the names of authors and conferences can be expressed in several ways and that some entities can have several types. The former problem is referred to as **instance ambiguity** (IA) – entities having the same type and the same name, and the latter **type ambiguity** (TA) – entities of different types which have the same name (common for acronyms). For example in *What papers has Ruslan Mitkov published in computational linguistics?* the expression *computational linguistics* can be both a named entity and refer to the *Journal of Computational Linguistics*, or can be a term and refer to the field of *computational linguistics*. This represents a type ambiguity. In the same question, it is possible to refer to *Ruslan Mitkov* using *R. Mitkov* or just *Mitkov* which exemplifies an instance ambiguity.

The Entity Annotator marks spans of text from the question with the type and ID of the entities they refer to. It retrieves a list of all the entities of a given type (e.g. [*swrc:Person*]) from the RDF data along with their known aliases and creates an index which will be used to identify and rank candidate matches in questions. The Entity Annotator aggregates several distinct annotators, one for each type (e.g. persons, publishers, conferences) and applies specialised semantic rules when indexing the lists extracted from the RDF data (e.g. derive the alias "Dornescu I." from "Dornescu Iustin", or "ACL08" from "ACL 2008"). Employing such semantic rules off-line, during data pre-processing is computationally motivated. At query time, the annotator ranks all the known alternative names (synonymy) and is also able to select an ambiguous list of entities which are equally ranked (polysemy). The latter is an extension of the initial QALL-ME web services, which allowed any span of the question to refer to at most one entity.

Since the QALL-ME web services definition does not allow annotated spans to overlap, when ranking the candidates, the annotator will prefer longer spans

and fewer distinct IDs. This constrains both types of ambiguity. For example when the user inputs the following question:

> *What did I. Dornescu publish in CLEF 2008?*

the following candidates are considered:

```
1  qa2:dornescu/iustin     "Dornescu Iustin"
2  qa2:dornescu/iustin  *  "Dornescu I."
3  qa2:dornescu/iulian     "Dornescu Iulian"
4  qa2:dornescu/iulian  *  "Dornescu I."
5  qa2:i_/dornescu      *  "I. Dornescu"
6  qa2:dornescu            "Dornescu"
7  qa2:domnescu/I.         "Domnescu I."
8  qa2:conf/clef/2008   *  "CLEF 2008"
9  qa2:conf/clef        *  "CLEF"
10 qa2:lcns/clef2008    *  "CLEF 2008"
11 qa2:ev/wk/clef/08    *  "CLEF 2008"
```

has three ambiguous author matches (two of which could be duplicates), and three ambiguous acronym matches. Candidate 6 is discarded because its span overlaps a larger one which has priority (more matching words means a greater confidence); candidate 7 has a lower confidence score due to the edit distance; while candidates 1 and 3 are aliases of the highest ranked candidates. The advantage when using this scheme is that the system can inform the user that "I. Dornescu" is ambiguous and can suggest which are the alternatives. "CLEF 2008" can refer to: an event (an workshop), a series of events, or two published proceedings (the electronic Working Notes and the LCNS volume).

In contrast to the annotators used for the domain of tourism, due to the large number of ambiguities that can be present in the questions, the annotators used here return all the possible annotations of a question, leaving the rest of the components to select the correct interpretations.

#### 4.2.3 Temporal annotator

Investigation of the domain of tourism, in which the QALL-ME framework was initially developed, revealed that a large number of questions contain temporal constraints. For this reason, the framework provides the possibility of using a temporal annotator that identifies temporal expressions and normalises them using the TIMEX2 standard [3].

The temporal tagger used for the domain of tourism follows the design and methodology of the temporal tagger described in [11] that is capable of identifying both self-contained temporal expressions(TEs) and indexical/under-specified TEs. The annotator described in [11] is rule-based and tackles more cases than necessary, making it too slow for our purposes. For this reason, a simplified temporal annotator was implemented [15]. Evaluation of this simplified temporal annotator revealed that it performs with high accuracy, most of the errors being due to the reduced number of rules implemented to increase its speed and ambiguities specific to the domain of tourism.

In contrast, questions about publications features very few temporal expressions, most of them being references to years (e.g. *Which journals/conferences published [AUTHOR] in [YEAR]?*). Theoretically it is possible to use more precise temporal expressions which specify both the month and the year, or even the day, but there are very few bibliographical databases which contain enough information to allow retrieval of articles based on the precise date when they were published. In light of this, we decided to use the temporal annotator implemented for the domain of tourism without any change, knowing that it can handle without a problem both references to years and specific dates. In addition, it can deal with indexical references such as *this year* without a problem.

The output of the temporal annotator is passed to the TIMEX2SPARQL web service which converts the TIMEX2 expressions to SPARQL snippets that are used to extract the answer to questions. This web service was used directly from the QALL-ME framework.

### 4.3 Entailment engine

The most common way to answer questions in restricted domains is to take a natural language question and transform it to a standard query language such as SQL. Often this requires performing deep linguistic analysis and reconstructing the logical form of the question. Despite the extensive manual work that goes into such a method, this approach fails quite often due to the language variability which allows the same question to be expressed in numerous ways. This problem is addressed in the QALL-ME project by using an entailment module that determines whether different expressions entail the same meaning and thus can share the same retrieval procedure [6]. The retrieval procedure is a SPARQL[12] template that is instantiated by the query generator (Section 4.4).

The English prototype that works in the domain of tourism uses an entailment engine which relies on domain ontology and its alignment to WordNet [9] to calculate the similarity between two questions and determine whether there is an entailment relation between them. Before this similarity score is calculated, as a pre-processing step, the expected answer types of the two questions and the types of entities appearing in the questions are compared and if they are not compatible the entailment relation is rejected [7].

For the entailment engine used in the bibliographical domain, we could not use the existing entailment engine as we could not easily incorporate the new domain ontology and we do not have its alignment to WordNet. Instead, we adapted the expected answer type module to recognise the five types of questions presented in Section 3.1 and use the similarity metric distributed together with the framework which is language independent. This similarity metric combines Levenshtein distance, cosine similarity, Euclidean distance and Monge Elkan distance. Despite this rather simple approach, the results are good.

[12] `http://www.w3.org/2001/sw/DataAccess/`

### 4.4 Query generator

The role of the Query generator web service is to produce a valid SPARQL query which can be used to answer a given question. To achieve this, it has a pre-prepared set of question patterns together with their retrieval procedure. The query generator relies on the entailment engine to determine which question pattern is entailed by a user question and in this way determine the retrieval procedure. The retrieval procedure is a SPARQL template which contains slots that are filled in using entities from the question.

The query generator used for the domain of tourism had to be changed in order to be used for the new domain. The first change was to produce question patterns and their SPARQL templates for the new domain. Ou et. al. [8] show how these can be automatically produced starting from the ontology, but the results they report are lower than for the manually produced set. In light of this, the question patterns and their SPARQL templates were manually produced on the basis of the types presented in Section 3.1.

The second change introduced was to allow the query generator to deal with ambiguous entities. The original query generator expects that each entity has exactly one interpretation. This is not the case in the bibliographical domain and the query generator was modified to gather all the possible interpretations when calling the entailment engine. For example, in

> *What did I. Dornescu publish in CLEF 2008?*

*CLEF 2008* can be interpreted both as name of proceedings and as an event. Given that there is no question pattern referring to an event, the entailment engine is used to rule out the interpretation where *CLEF 2008* is an event.

The SPARQL query generated in this web service is used in the next step to retrieve the actual answer.

### 4.5 Answer pool

The role of the Answer Pool web service is to take the SPARQL query generated by the query generator and retrieve the results from an RDF database. The answer pool service is domain independent and was used with almost no changes. For cases where entities are ambiguous (both type and instance ambiguities) and the question can be interpreted in several ways, several SPARQLs are generated and therefore several answer sets are retrieved. This represents an improvement over the service used in the domain of tourism.

The answer pool also plays a role in dealing with ambiguities. Some SPARQL queries retrieve no results due to the interpretations of some entities (e.g. in the case of *What did I. Dornescu publish in CLEF 2008?*, *I. Dornescu* can be interpreted as *Iulian Dornescu* who is not an author in our database). In this case the ambiguity is not shown to the user.

For cases where several interpretations of the question yield answers, a presentation module is used to show the results in a user-friendly manner. This is presented in the next section.

### 4.6 Presentation module

The presentation module is domain dependent and for this reason is not part of the framework. For the domain of tourism, the output of the system is textual answers, speech, maps, images and videos and different presentation modules were used depending on where the results were displayed (i.e. computer screen or mobile phone). This are not appropriate for the bibliographical domain, where for presenting the results to the user, we currently use the Citeline, a tool which turns a publication list in BibTeX format into a visual exhibit.[13] To be able to use Citeline the RDF data retrieved by the Answer Pool web service is converted back to BibTeX format. A screenshot of the presentation module is presented in Figure 3. No language processing is performed at this stage and ambiguities present in the input question are dealt with using faceted browsing.

## 5 Conclusions

This paper presented the adaptation of the QALL-ME Framework to the bibliographical domain. This process required to create a domain ontology and adapt the web services that constitute the framework.

To model the bibliographical domain, a subset of the SWRC ontology [14], an ontology for modelling entities of research communities such as persons, organisations, publications (bibliographic metadata) and relationships amongst them was used. The terminology specific to the domain was encoded using the `skos` ontology.

The QALL-ME Framework is based on a Service Oriented Architecture (SOA) and realised using web services. Each of the web services were described with emphasise on the changes necessary due to the new domain. One of the main problems that had to be faced was the large number of ambiguities (both instance and type ambiguities) that can be present in a user question. As a result, the annotation web services had to be changed to allow multiple annotations for a text span. This in turn, determined changes in the query generator and answer pool web services, in order to allow them to deal with multiple interpretations for a question.

Currently no formal evaluation of the system was carried out. In the future, we plan to collect questions from users and perform an on-field evaluation in order to be able to assess the performance of the system.

## Acknowledgements

This work is supported by the EU-funded project QALL-ME (FP6 IST-033860).

## References

[1] W. W. Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123, 1995.

[13] `http://citeline.mit.edu/`

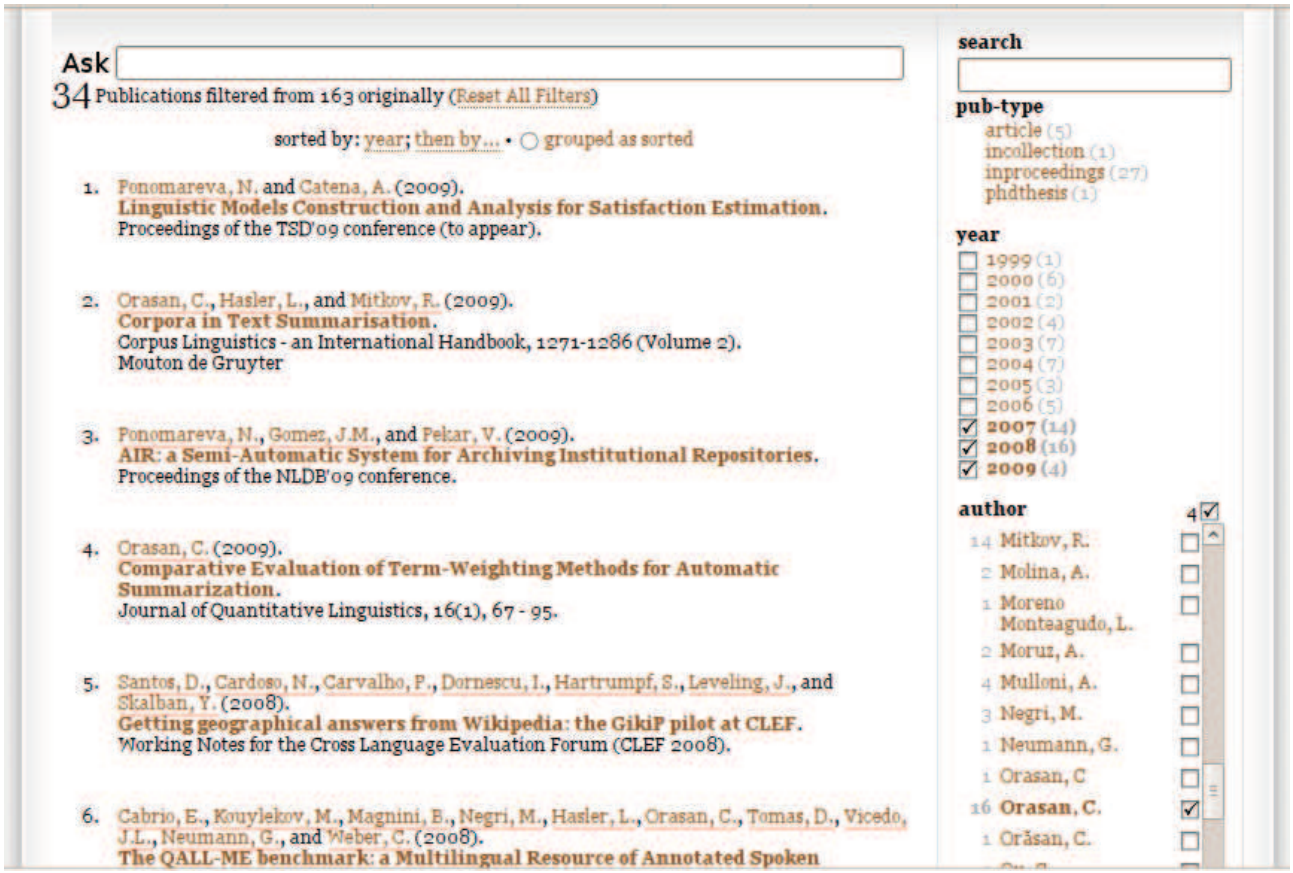



**Fig. 3:** *Faceted browsing of results using powered by MIT Citeline, Babel and Exhibit*


[2] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of ACL02*, 2002.

[3] L. Ferro, L. Gerber, I. Mani, B. Sundheim, and W. G. TIDES 2005 Standard for the Annotation of Temporal Expressions. Technical report, MITRE, April 2005.

[4] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[5] A. E. Monge and C. Elkan. The field matching problem: Algorithms and applications. In *KDD*, pages 267–270, 1996.

[6] M. Negri, B. Magnini, and M. O. Kouylekov. Detecting expected answer relations through textual entailment. In *9th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 532–543, Heidelberg, Germany, 2008. Springer.

[7] S. Ou, D. Mekhaldi, and C. Orăsan. An ontology-based question answering method with the use of textual entailment. In *Proceedings of NLPKE09*, 2009.

[8] S. Ou, C. Orăsan, D. Mekhaldi, and L. Hasler. Automatic question pattern generation for ontology-based question answering. In *Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference (FLAIRS2008)*, pages 183 – 188. Menlo Park, CA: AAAI Press, 2008.

[9] S. Ou, V. Pekar, C. Orăsan, C. Spurk, and M. Negri. Development and alignment of a domain-specific ontology for question answering. In European Language Resources Association (ELRA), editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 28 – 30 2008.

[10] N. Ponomareva, J. M. Gomez, and V. Pekar. Air: a semi-automatic system for archiving institutional repositories. In *Proceedings of the 14th International Conference on Applications of Natural Language to Information Systems (NLDB 2009)*, Saarbrucken, Germany, June 24-26 2009.

[11] G. Puşcaşu. A framework for temporal resolution. In *Proceedings of the 4th Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal, May, 26-28 2004.

[12] B. Sacaleanu, C. Orasan, C. Spurk, S. Ou, O. Ferrandez, M. Kouylekov, and M. Negri. Entailment-based question answering for structured data. In *Coling 2008: Companion volume: Posters and Demonstrations*, pages 29 – 32, Manchester, UK, August 2008.

[13] A. Salhi and H. Camacho. A string metric based on a one-to-one greedy matching algorithm. *Research in Computer Science*, number 19:171–182, 2006.

[14] Y. Sure, S. Bloehdorn, P. Haase, J. Hartmann, and D. Oberle. The swrc ontology - semantic web for research communities. In C. Bento, A. Cardoso, and G. Dias, editors, *Proceedings of the 12th Portuguese Conference on Artificial Intelligence - Progress in Artificial Intelligence (EPIA 2005)*, volume 3803 of *LNCS*, pages 218 – 231, Covilha, Portugal, DEC 2005. Springer.

[15] A. Varga, G. Puşcaşu, and C. Orăsan. Identification of temporal expressions in the domain of tourism. In *Knowledge Engineering: Principles and Techniques*, volume 1, pages 29 – 32, Cluj-Napoca, Romania, July 2 – 4 2009.

[16] W. E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Bureau of the Census, 1999.

[17] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers, 2005.

# Personal Health Information Leak Prevention in Heterogeneous Texts

Marina Sokolova[1], Khaled El Emam[1,2], Sean Rose[2], Sadrul Chowdhury[1], Emilio Neri[1], Elizabeth Jonker[1], Liam Peyton[2]

[1]Electronic Health Information Lab
Children's Hospital of Eastern Ontario
401 Smyth Rd., Ottawa, Canada, K1H 8L1

[2] University of Ottawa
800 King Edward, Ottawa, Canada, ON K1N 6N5

{*msokolova, kelemam,srose,schowdhury,eneri,ejonker*}*@ehealthinformation.ca*
*lpeyton@site.uottawa.ca*

## Abstract

We built a system which prevents leaks of personal health information inadvertently disclosed in heterogeneous text data . The system works with free-form texts. We empirically tested the system on files gathered from peer-to-peer file exchange networks. This study presents our text analysis apparatus. We discuss adaptation of lexical sources used in medical, scientific, domain for analysis of personal health information.

## Keywords

information leak prevention, personal health information

## 1 Introduction

When electronic means became the prime instrument for storage and exchange of personal health data, the risks of inadvertent disclosure of personal health information (i.e., details of the individual's health) had increased. Inadvertently disclosed personal health information facilitates criminals to commit medical identity theft, i.e., allows an imposter to obtain care or medications under someone else's identity [10]. Furthermore, PHI is an important source of identity theft [14], and has been used by terrorist organizations to target law enforcement personnel and intimidate witnesses [21]. PHI security breaches had happened in various domains. PHI has leaked from a Canadian provincial government agency [6] and from health care providers, through documents sent by employees and medical students [18]. There are several examples of the confirmed leaks on peer-to-peer file sharing networks: a chiropractor exposed his patient files on a peer-to-peer network, including notes on treatments and medications taken [20], a criminal obtained passwords for 117,000 medical records through a file sharing network [24]. In this work, we present a system which detects personal health information (PHI) in free-form heterogenous texts. It can be used to detect the inadvertent disclosure of PHI, thus, benefit information leak detection.

Texts which contain personal health information can be written by doctors, nurses, medical students or patients and can be obtained from various sources within the health care network. Hospitals provide patient health records (e.g. speech assessment, discharge summaries, nurse notes), patients write letters, notes, etc. These texts can be found on the web, within peer-to-peer file exchange networks, and on second-hand disk drives [12, 25]. Within those texts, we seek the information which refers to individual's health: disease (pneumonia)[1], treatment procedures (X-rays), prescribed drugs (aspirin), health care providers (the Apple Tree Medical Centre). Our system contributes to information leak prevention, a growing content-based part of *data leak prevention*.

There are several differences between our tool and the previous work on PHI leak prevention. Our system detects personally identifiable and health information. Previous work focussed on detection and de-identification of personally identifiable information (.e.g,, person names, phone numbers, age-related dates), but did not retrieve health information. Our system processes data of unknown content, context and structure. Whereas, previously the PHI leak prevention systems operated within a closed domain of hospital patient records, where the input data was guaranteed to contain PHI. As we mentioned, these systems were built to find and alter personally identifiable information, e.g., name, age, phone.

In our case, the input files come with unknown content. Sometimes the file content excludes a possibility of personal information, e.g., a young-adult vampire-romance novel Twilight, a research presentation Statistical Learning Theory, a song Quel Temps Fait Il A Paris.

[1] Hereinafter, this font signifies examples.



*Workshop Adaptation of Language Resources and Technology to New Domains 2009 - Borovets, Bulgaria*, pages 58–69

Sometimes, file contents may suggest holding personal information and PHI, e.g., personal correspondence, documents from lawyer or physician offices. In many other cases, files fall between these two categories. We discard files which we identify as being highly unlikely to contain PHI and concentrate on the analysis of the remaining files. In the remainder of the presentation, we define personal health information, provide examples of texts containing PHI and discuss the extent of confirmed inadvertent PHI leaks. We define pairs of possible/impossible and probable/improbable PHI containers. Our data and empirical results are presented after that. We follow with discussion of related work and motives for the adaptation of medical knowledge sources. At the end, we present plans for future work and conclusions.

## 2 Background

Our group works on prevention of inadvertent disclosure of personal health information in heterogenous text data. Personal health information (PHI) refers to ailments, treatments and other health-specific details of an individual. In Ontario, Canada, the Personal Health Information Protection Act [1] defines PHI as information that:

1. relates to the physical or mental health of the individual, including information that consists of the health history of the individual's family
2. relates to the providing of health care to the individual, including the identification of a person as a provider of health care to the individual,
3. is a plan of service within the meaning of the Long-Term Care Act for the individual
4. relates to payments or eligibility for health care, or eligibility for coverage for health care, in respect of the individual
5. relates to the donation by the individual of any body part or bodily substance of the individual or is derived from the testing or examination of any such body part or bodily substance
6. is the individual's health number
7. identifies an individual's substitute decision-maker.

It also states that "[identifying information] identifies an individual or for which it is reasonably foreseeable in the circumstances that it could be utilized, either alone or with other information, to identify an individual" [1]. Below we present samples of texts with PHI:

[THE PATIENT] ADMITTED IN TRANSFER FROM [HOSPITAL NAME] FOR MENTAL STATUS CHANGES POST FALL AT HOME AND CONTINUED HYPOTENSION AT CALVERT HOSPITAL REQUIRING DOPAMINE;

[The person]'s heart attack happened on a sidewalk in Midtown Manhattan last May. He was walking back to work along Third Avenue with two colleagues after a several-hundred-dollar sushi lunch. There was the distant rumble of heartburn, the ominous tingle of perspiration.

*I, [John Doe], want to make the following statement under Oath: On 02 May 2007, in duty hours between 1030 1100 hrs I was practicing drill exercises for our next mobilization to [Place Name, Country]. In that exercise trying to put someone under arrest the person fell down on my left knee causing me a contusion.*

The act protects the confidentiality of PHI and the privacy of individuals with respect to that information, while facilitating the effective provision of health care. Similar PHI protection acts have been enabled: US has the Health Insurance Portability and Accountability Act, often known as HIPAA [2], EU – Directive 95/46/EC , or, Data Protection Directive [3], although some details vary.

We divide PHI into two broad categories: personally identifiable information (PII), e.g., name, birth data, address, and health information (HI), e.g., diagnosis, prescribed drugs. Table 1 lists some PII and HI subcategories and their examples. Further, in this paper, we concentrate on the HI category.

| Personal information | |
|---|---|
| Info categories | Examples |
| Given names | Serge, Jasmine |
| Locations | London, Osaka |
| Addresses | 401 Smyth Rd., Empire State Bldg. |
| Dates | 02 May 2008, 05/14/07 |
| **Health information** | |
| Info categories | Examples |
| Disease names | Pneumonia, arthritis |
| Symptoms | calcium deficiency |
| Drug names | Aspirin, Fosamax |
| Health care providers | CHEO, Dr. Joe Doe |

**Table 1:** *PHI categories and their examples*

Previously, second hand disk drives and peer-to-peer file exchange networks (p2p networks) were searched for the presence of texts with PHI [12, 25]. In [12], we studied the extent of inadvertent PHI exposure on second hand computer hard drives. We purchased functional disk drives from various second-hand computer equipment vendors, then examined sixty drives using

[2] http://www.hhs.gov/ocr/privacy/index.html
[3] http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=-CELEX:31995L0046:EN:HTML

digital forensic tools. The focus was on drives with a capacity range of 10 GB to 40 GB, which were used by individual end users in desktop machines and servers. The recovered data was examined manually by two experts. PHI was found in 425 files gathered from 11(18%) disk drives. 5 disk drives contained PHI of the computer owner, 6 disks – PHI of other people. In [25], we suggested evaluation measures for automated detection of PHI files. We semi-manually examined 859 files gathered from two p2p networks. 8(1%) files contained PHI. Although the number of files was comparatively small, the personal health information contained in these files potentially was exposed to millions of on-lookers.

# 3 Methods

## 3.1 Files as Personal Health Information Containers

We consider that among heterogeneous files of unknown content, some files can be *possible* containers of personal information, whereas others cannot, i.e. *impossible* containers. Only the possible containers which contain personal information may become *probable* sources of PHI leaks. The files become actual PHI leaks if they additionally contain health information. The impossible containers are improbable to leak personal information, and consequently, improbable to leak PHI. In other words,

$$DATA = POS + POS' \quad (1)$$

$$PHI \subset PROB \subset POS \quad (2)$$

$$POS' \subset PROB' \subset PHI' \quad (3)$$

where $DATA$ denotes the data, $POS$ is a set of possible containers, $PROB$ – a set of probable containers, $PHI$ – a set of files with PHI. $'$ marks the set complement. Based on the set relations 1–3 , we apply the following rules of inference ($f$ denotes a file):

$$f \in POS' \rightarrow f \in PROB' \rightarrow f \in PHI' \quad (4)$$

$$f \in PHI \rightarrow f \in PROB \rightarrow f \in POS \quad (5)$$

To categorize files into $POS$ or $POS'$, we recall that personal information is information which identifies an individual, either by itself or jointly with other information (see Section 2). According to this definition, two types of files immediately fall into the impossible container category:

1. contents not concerned with individuals;
2. contents not able to identify individuals.

To find the first type, we look for files with content unrelated to individuals, e.g., fictions, songs . To find the second type, we look for files whose content is unreadable for end users, e.g viruses. All the other files might or might not contain personal information, and are put into the possible container category. From those, only the files that are shown to contain personal information are marked for further investigation, i.e. health information detection.

Rules 4 – 5 are used in our text analysis system. The system's work cycle has three phases. On the first two phases, it identifies impossible containers and removes them from the set. On the third phase, it works with the remaining files which we consider to be possible containers. The utilized text analysis gradually deepens:

**shallow analysis** the file titles and contents are treated as streams of characters (phase 1);

**partial content analysis** for each file, a limited keyword search is performed on a small portion of text (phase 2);

**deep content analysis** texts are mined for syntactic and semantic patterns (phase 3).

## 3.2 Creating the Corpus

For our experiments, we used files gathered from two peer-to-peer file sharing networks (p2p networks). Peer-to-peer networks allow decentralized sharing of computer resources, including those that provide the infrastructure for direct, real-time communication and collaboration between peer computers. The networks are known for hosting files with PHI information.

The usage, together with the observed security weakness, marks p2p network data as a possible source of information leaks, including PHI leaks. This assumption was confirmed by data management studies [18].

To gather data, we obtained the project approval from the Research Ethics Board of Children's Hospital of Eastern Ontario. The files were gathered from April 2008 till June 2009. The *Gnutella* and *eD2K P2P* networks were selected due to their prevalence and global popularity.[4] To automatically capture samples of p2p files, we modified the publicly available ShareAZA[5] p2p client. The tool is a software package which allows one to connect to multiple P2P networks simultaneously in order to search for and download files. Modifications to this client included changes to the search function as well as increased logging capabilities. The search function was modified to automatically search for any document file (Microsoft Word, Raw Text, Rich Text, Excel, Powerpoint, PDF, Wordpad, XML, etc) and automatically retrieve it. Automatic searches were conducted by the code at fifteen minute intervals. A semi-manual analysis of the first data sample (859 files) showed the presence of PHI on the two p2p networks [25]. In total, we have gathered 2852 files. The data was sent for processing "as is", without preliminary pre-processing: we preserve all the initial spelling, capitalization, grammar, etc.

[4] `http://www.kolabora.com/news/2004/01/09/popular_p2p_tools_and_programs.htm`, retrieved Aug 12, 2009

[5] `http://shareaza.sourceforge.net/?id=source`

| Detection of publishable and educational text | | | |
|---|---|---|---|
| Categories | Examples | Categories | Examples |
| Books | ebook, ISBN, publisher | Periodic | magazine, article, volume |
| Retail | manual, readme, copyright | Genre | biography, fiction, sci-fi, whodunit |
| Book type | dictionary, novel, cookbook | Publishable | abstract, acknowledgement, introduction |
| Education | theses, assignment, course-work | Special (NA) | Bible, dummies, Microsoft, software |
| Detection of non-personal text | | | |
| Categories | Examples | Categories | Examples |
| Music | album, ballad, song | Fictionals | Harry Porter, Scarlet O'Hara |
| Advertisement | Mrs Tiggy Winkles, Tim Hortons | Politics | Al Gore, Winston Churchill |

**Table 2:** *Categories and terms for partial content analysis*

### 3.3 Empirical Text Processing

**Shallow analysis** On this step, we aimed to remove files which were the most unlikely candidates to leak PHI. We assumed that any published text was not leaking PHI: fictions described non-existing heros, magazines and newspapers obtained person's consent on information disclosure, songs were not providing enough details, etc. Corrupted files and non-text files (images, music) were other candidates for a fast removal. On this step, we applied string matching and character-based $N$-gram modelling methods.

First, we processed only the file titles. The titles were compared with the `Amazon.com` database. 723(25.35%) files were removed after their titles were found in the database, e.g. New York New York, Abba_-The_Winner_Takes_It_All, abominable snowman were discarded. However, if there was no exact title matching, the file was passed for further processing.

On all of the following steps, we worked with whole file data, i.e., body and title. Immediately after the `Amazon.com` search, the files were passed through a text extractor. 37(1.30%) non-text files ( images, music, viruses) were removed. Then we applied a modified version of a publicly available language identifier TextCat [6]. For each file, the tool built character-based $N$-grams. The $N$-grams were compared with language models for 69 languages. The best-fitting model provided the language text tag. On that step, 724(25.39%) non-English texts were discarded (e.g., 13 de octubre 20008, canserle ilgili bilgiler).

**Partial content analysis** The goal of this phase is to identify and remove publishable, educational and non-personal texts which cannot be identified by the title string matching. #11 The Dragonfiend Pact_1, Copy of VintagePatternBook are book files with "camouflaged" titles that passed through the string matching method of the shallow filter; chriscolumbus is a student assignment which could not be detected earlier. To find such files, we define categories of terms characteristic to publishable and educational texts. One category represents local North American (NA) preferences in the files; see the upper part of Table 2 for examples.

We also look for texts with non-personal content. Music texts, discussion of popular fictional characters, current political events, and advertisements would be unlikely candidates for leaking explicit, detailed PHI. The lower part of Table 2 lists detection categories and examples of terms.

In practice, we applied the key-word search to the titles and the first 200 words of the body text. Many detected texts were educational (assignments, reports, theses), some represented small literary forms (essays, poetry, self-published books). Manuals, tech reports, articles were also detected on this step. As a result, 605(21.21%) publishable and non-personal texts were filtered out.

Another task was to find and remove multiple copies of the same file. For each pair of remaining files, we compared their sizes, titles, and first and last sentences. If all parameters were the same, we tagged two files as duplicates and kept only one for further processing.[7] 41(1.44%) files were removed on this step. The remaining p2p files are deemed susceptible to information leaks, i.e., possible containers, and passed to the deeper analysis stage; Figure 1 shows the proportional distribution of the processed files.

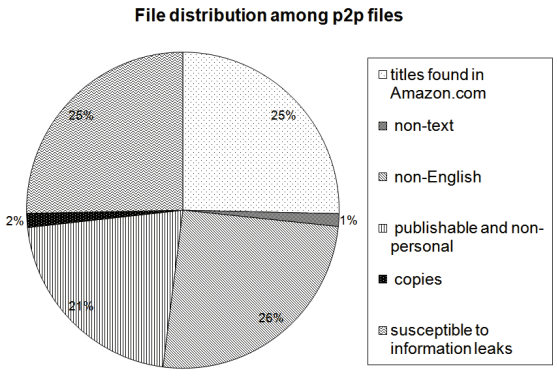



**Fig. 1:** *File content distribution found by shallow and partial content analysis*

[6] http://odur.let.rug.nl/ vannoord/TextCat/

[7] Here and everywhere, when appropriate, we used hashes and compared hash values.

**Deep content analysis** 722(25.31%) files remained after the shallow and partial content analysis phases. Potentially, those files may have personal content, thus, hold personal health information (i.e., possible containers). In processing these files, we want to identify a set of probable containers first and then work with this set only. For this, complete contents are analyzed with a combination of syntactic, and semantic methods. This phase uses external resources: dictionaries and knowledge sources.

The heterogeneity of the data makes it unrealistic to expect such file commonalities as text structure, grammar style, content word vocabulary, etc. We, instead, rely on the definition of PHI which can be expressed through a reasonably limited number of semantic categories, e.g., person and geographic names, disease names and symptoms. Sets of syntactic rules are used to identify references to individuals, locations and age-identifiable events (birth, death). We parse sentences to find preposition phrases, noun phrases and verb phrases. Soft Regular Expressions(RE) are used to extract numeric-based categories, such as phone number, street number and unit, dates, and email.

In data management and privacy protection, geographic information is shown to be the single most important category responsible for person identification [16, 3]. We implement geographic information extraction for the following categories:

**country** : all the UN-recognized countries and their capitals on all the continents (France, Paris; Liberia, Monrovia), and self-proclaimed entities (Eritrea, Abkhazia) ;

**place** : in US: state name, state capital, the largest city (Illinois, Springfield, Chicago); in Canada: province, province capital, largest cities, tourist attractions (Alberta, Edmonton, Calgary, Banff), the same – for territories; in Europe, Latin America, Asia, Africa, Australia: Alpha, Beta and Gamma world cities [8].

**code** : US' ZIP code (Massachusetts 02163, NY 10027), Canada's postal code (K1H 8L1);

**street** : for US and Canada – type (Avenue, Ch., Street, Beach), number (401 Smyth Rd.);

**landmarks** : Empire State Building, CN Tower , Niagara Falls, etc.

From other named entities, we concentrate on recognition of person names (John Smith), organizations (Nepean High School) and health care providers (Ottawa General Hospital). We considered organizations to be geographic pointers, as they can tie an individual to a certain location. The Contact Us link on the organization web site or a name part (Boston University) are strong indicators of a person's geographic affiliation. Hospital, doctor, and registered nurse information in the file makes it a strong candidate for a PHI leak. We combined a key word search and syntactic patterns to find these named entities. We did not apply look ups of health care providers, as no confirmed high quality sources are available

To reduce computationally expensive person name look-up, we first searched for patterns of family relations (My daughter, an uncle of) and self-identification (my name, sincerely). Other patterns are event-related (was born, died in). Depending on the patterns, either preceding or following capitalized words are stored in the name list. Further, when the tool checks for a person name, it will first check with the file dictionary. The pattern search is augmented with an RE-based search. The latter is combined with the person name look up. We use three proprietary dictionaries: female and male first names and last names. Our dictionaries contain formal and informal name forms (William, Bill, Billy) and non-Anglo-Saxon names (Meehai, Leila).

To be marked as probable containers, files should contain a geographic identifier (e.g., street address, place name, organization ) and two other identifiers, e.g., first name and last name, first name and another geographic identifier, last name and a phone number. The 345(12.10%) probable containers were then passed into the final phase of health information extraction where 12 PHI files were found (Figure 2). We discuss in detail PHI extraction in Section 3.4.

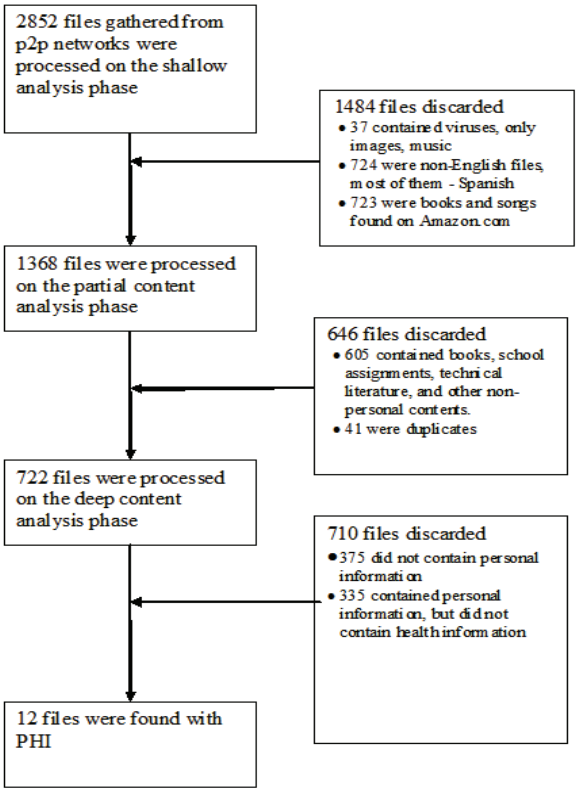



**Fig. 2:** *Gradual reduction in the number of analyzed files*

[8] http://en.wikipedia.org/wiki/Global_city#GaWC_Inventory_of_ World_Cities

## 3.4 Health Information Extraction

### 3.4.1 Ontology structure

Our ontology building works as follows:

i to identify a small number of semantic categories which correspond to the main categories of Health Information;

ii work with each category separately, identifying the information that should be analyzed;

iii apply Information Extraction methods to find the information indicators in the existing sources.

At the initial step, we form three genetic semantic categories – disease, drugs, and symptoms – as was discussed in Section 2; see Table 1 for examples. For diseases and drugs, we concentrate on extraction of their names. The category contents were derived from Webster's New World Medical Dictionary [15], the International Classification of Diseases (ICD9 codes)[9], the Medical Dictionary for Regulatory Activities (MedDRA)[10] and Canadian Drug Product Database (Active and Inactive) [11].

ICD9 codes [2] are used by health care professionals to tag and classify morbidity data from inpatient and outpatient records, physician offices, as well as most of the National Center for Health Statistics (NCHS)[12] and the Canada Institute for Health Information[13] surveys. The codes are divided into two sections: one containing diseases and injuries (ICD9CM Disease and Injury), and another containing surgical, diagnostic, and therapeutic procedures (ICD9CM Procedures). ICD9 provides the hierarchy of diseases where terms on every level relate to the individual's health. The following sample presents the complete hierarchical snapshot for cholera:

1 INFECTIOUS AND PARASITIC DISEASES (001-139)
INTESTINAL INFECTIOUS DISEASES (001-009)
Excludes: helminthiases (120.0-129)

001 Cholera

001.0 Due to Vibrio cholerae

001.1 Due to Vibrio cholerae el tor

001.9 Cholera, unspecified

This succinctness allows reduction in the source processing and simplifies information extraction steps.

The Canadian Drug Product Database (CDPD) contains product specific information on drugs approved for use in Canada. It includes human pharmaceutical and biological drugs, veterinary drugs, and disinfectant products. Additionally, a database of previously available drugs is maintained. However, an average, non-expert individual may treat drugs as consumer goods and refer to them in different ways. That is why we expect that drug names can vary from a generic, non-proprietary, name as Ibuprofen to a more specific brand name as Advil.

To accommodate extraction of various drug names, we sought information provided by Merck & Co., an international pharmaceutical company[14]. We obtained a list of generic drug names and the trade names associated with them [15].

Patient symptoms such as chest pain or headache, as well as mentioned procedures such as heart surgery, also consist of health information, as they may allow one to infer a specific medical, behavioural or psychological condition or ailment of another individual. To identify patient symptoms, we use the MedDRA dictionary which covers a wide range of terminology including symptoms and signs (i.e. visible symptoms).

However, the listed above resources leave some gaps in PHI detection. The most noticeable absentees are acronyms (ICU) and providers (therapist, surgeon), but also some condition names (blood pressure, tube fed). To fill the gaps, we manually searched the Webster's medical dictionary. Figure 3 shows the structure of our ontology.

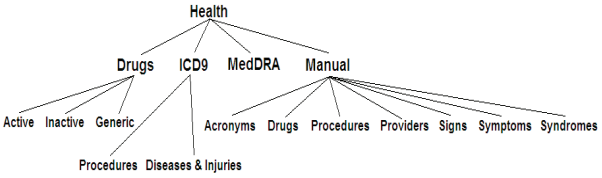



**Fig. 3:** *The structure of the knowledge source*

### 3.4.2 Terms and term units

We aimed to populate the ontology with PHI-related single terms (diabetes) and term units (Felty's syndrome). We first minimized the above mentioned resources by removing un-related categories (e.g., animal diseases, animal drugs). Then the remaining resource texts were normalized : converted to lowercase, punctuation marks and numbers were removed, and stop words (of, when) were eliminated.

We consider that a term unit is a sequence of two or more consecutive units words, that has characteristics of a syntactic and semantic unit, i.e. collocation. To identify collocations, we used a subset of 700,000 articles from the MEDLINE corpus [16], a repository of medical documents. The normalized text was processed by TEXT::NSP[17], a collocation extraction and $N$-gram building tool. We looked for $N$-grams of

[9] http://www.cdc.gov/nchs/about/otheract/icd9/abticd9.htm
[10] http://www.meddramsso.com/MSSOWeb/index.htm
[11] http://www.hc-sc.gc.ca/dhp-mps/prodpharma/databasdon/index-eng.php
[12] http://www.cdc.gov/nchs/
[13] http://secure.cihi.ca/cihiweb/dispPage.jsp?cw_page=home_e
[14] http://www.merck.com/
[15] http://www.merck.com/mmpe/appendixes/ap2/ap2a.html
[16] http://medline.cos.com/
[17] http://search.cpan.org/ tpederse/Text-NSP-1.09/lib/Text/NSP.pm

length 2 and 3. From these counts a log-likelihood statistical significance test was performed to determine if a given textual unit qualifies either as an $N$-gram or a collocation. We used the tool default settings. The sets of trigrams and bigrams were then merged into a single set of collocations which is used later in the process.

Each of the IC9CM, CDPD, Merck, MedDRA, and manually created datasets were then used to find and extract PHI entities by applying the following procedure: try to match collocations, and if a match is found mark it as a PHI indicator; if a word is not matched as part of a collocation, mark it as a single word PHI indicator. However, this inclusiveness may reduce the detection power of the terms. For example, cat and magic would be extracted from the drug base entry:

```
CAT IV - SUNBURN PROTECTANTS, LEG MAGIC
```

Thus, the list of HI indicators had to be filtered and pruned to eliminate false HI indicators like these. For filtering of these words, we used data obtained from the British National Corpus (BNC)[18] The BNC is used to filter out terms that occur over 1800 times. This threshold was chosen, as hospital occurs slightly less than 1800 times. Additionally, we eliminated content words which appear among the top frequent 5000 words in the Brown corpus [19]. No collocations were filtered out by this process, which may in part cause false PHI indicators to be included in the final list. This process resulted in the ontology with 62004 entities, including 25528 unigrams, 27641 two term collocations, and 8835 three term collocations. Figure 4 sketches the ontology building process.

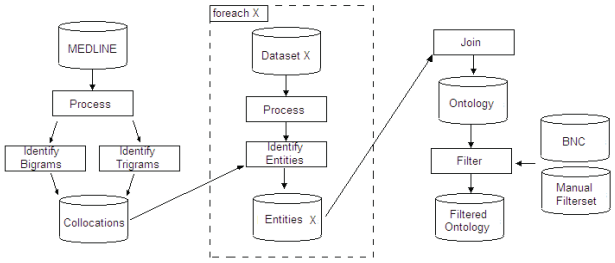



**Fig. 4:** *Health Information ontology building. X denotes an external source*

### 3.4.3 Text classification

After the ontology has been created, it can be used to classify a text as PHI or not PHI. Based on the number of trigrams $T$, bigrams $B$ and unigrams $U$ in the text, identified as $HI$ indicators, along with the length of the document $n$, we classified the text as either PHI or not PHI. We then could weight $N$-gram contributions proportionally to the number of words $N$ :

$$\frac{3T + 2B + U}{n} > H \rightarrow PHI \tag{6}$$

[18] http://www.natcorp.ox.ac.uk/
[19] http://www.edict.com.hk/textanalyser/

Texts were marked as $PHI'$ if their $N$-grams did not satisfy Eq. 6 .

If we wanted to use the ontology structure, we would augment the formula by including the term category contributions. Each category is assigned a normalized weight based upon its quality. We rate the quality of a resource based upon the percentage of entities within it that are not filtered out. So a category $C_i$ of original size (i.e., a number of initial terms) $S_i$ and filtered size (i.e., a number of remaining terms) $F_i$ has quality $Q_i$ defined as:

$$Q_i = \frac{F_i}{S_i} \tag{7}$$

We then compute the normalizing factor $M$ and the weight $W_i$, given to an entity from a given category:

$$M = \frac{1}{n} \sum_{1}^{k} Q_i \tag{8}$$

$$W_i = \frac{Q_i}{M} \tag{9}$$

The weighted formula for text classification becomes:

$$\sum_{1}^{k} W_i \frac{3T_i + 2B_i + U_i}{n} > H \rightarrow PHI \tag{10}$$

Empirical testing of several thousand documents allowed us to determine a suitable threshold value of 0.04. Yet, the threshold $H$ was chosen from a set of empirical experiments such that it optimized the precision, while keeping the recall at 100%. Thus, the chosen threshold value may be overly fit to the test data. In the future, when more sample data becomes available, experiments should be performed to try and to determine what the optimal threshold value should be. Figure 5 depicts the PHI text classification process.

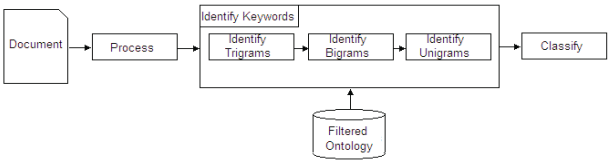



**Fig. 5:** *HI text classification*

# 4 Empirical Results

**Evaluation on screened positive examples** Evaluating the correct identification of PHI leaks presents a certain methodological difficulty. The number of PHI files is negligible even if compared with the number of probable containers. On the other hand, all of the PHI files exhibit specific characteristics: they contain personally identifiable and health information. Hence, we can apply measures that evaluate a tool's performance only on examples which satisfy pre-determined criteria [23]. The other examples

are ignored. The approach – evaluation on screened positive examples – has been shown effective and appropriate for PHI leak detection [25]. Table 3 presents the confusion matrix:

| | | Predicted HI =1 | Predicted HI =0 |
|---|---|---|---|
| Actual | PHI=1 | $n^+_{PHI}$ | ? |
| | PHI=0 | $n^+_{\overline{PHI}}$ | ? |
| | | $n^+$ | $n^-$ |

**Table 3:** *Confusion matrix for classification of screen positive examples*

We compute *True Detection Probability*($\hat{TDP}$) and *False Referral Probability*($\hat{FRP}$):

$$\hat{TDP} = \frac{n^+_{PHI}}{n^+ + n^-} \tag{11}$$

$$\hat{FRP} = \frac{n^+_{\overline{PHI}}}{n^+ + n^-}. \tag{12}$$

$TDP$ shows the proportion of files an algorithm marked as having the PII and HI indicators `and` containing PHI. $FRP$ shows the proportion of files the algorithm marked as having the PII and HI indicators but `not` containing PHI.

To put the measures in perspective, we use the ideal classification (Table 4) where all the predicted HI files are indeed the PHI files and visa-verse:

| | | Predicted HI =1 | Predicted HI =0 |
|---|---|---|---|
| Actual | PHI=1 | $n^+$ | ? |
| | PHI=0 | 0 | ? |
| | | $n^+$ | $n^-$ |

**Table 4:** *Confusion matrix for the ideal classification of screen positive examples*

Then, $\hat{TDP}_i = \frac{n^+}{n^+ + n^-}$, $\hat{FRP}_i = 0$. A method works better if $\frac{\hat{TDP}}{\hat{TDP}_i}$ is close to 1 and $\hat{FRP}$ – to 0.

**The ontology application** We tested our tool on several sets of p2p files. Here we report typical results, in terms of accuracy.

(a) 72 files were randomly obtained from a peer-to-peer file sharing network. The set contained nine files with health care information (parents' notes, letters, documents from a lawyer office). We used our HI ontology and manually examined all the labels output by the system. Table 5 shows the results.

Here, $\hat{TDP}_i = 11.11\%$. We obtained $\hat{TDP} = 11.11\%$ , thus, $\frac{\hat{TDP}}{\hat{TDP}_i} = 1$. We obtained $\hat{FRP} = 1.39\%$: HI indicators were extracted from a summary of a teen fiction which was not a PHI file.

| | $HI_o$ =1 | $HI_o$ =0 |
|---|---|---|
| PHI=1 | 8 | ? |
| PHI=0 | 1 | ? |
| | $n^+$ | $n^-$ |

**Table 5:** *HI ontology: classification of the* 72 *files*

For *per-term* extraction accuracy, we obtained Recall = 100%, i.e., all health care indicators were correctly extracted. On the relevant, true PHI, eight documents, we obtained Precision =100%, i.e. all extracted indicators were health care indicators indeed.

(b) To test the proposed system a set of 76 texts were used. This set was composed of 4 PHI texts and 72 non-PHI texts. Table 6 lists the results.

| | $HI_o$ =1 | $HI_o$ =0 |
|---|---|---|
| PHI=1 | 4 | ? |
| PHI=0 | 2 | ? |
| | $n^+$ | $n^-$ |

**Table 6:** *HI ontology: classification of the* 76 *files*

We obtained $\hat{TDP} = 5.26\% = \hat{TDP}_i$. Again, $\frac{\hat{TDP}}{\hat{TDP}_i} = 1$. We obtained $\hat{FRP} = 2.78\%$. Of the false PHI files, one was a resume of a healthcare worker, and one was an unfilled health insurance form. In both cases the falsely classified texts contained both PII and HI, yet there was no link between the two. In the future, a deeper analysis phase – perhaps, co-reference resolution – of potential PHI texts could be done, potentially increasing the precision of the method as a whole.

**Medical Subjects Heading application** Medical Subjects Heading (MeSH), a controlled vocabulary thesaurus, is produced by the National Library of Medicine.[20]. Its hierarchical and categorized structure is often used in analysis of medical texts [7]. In our case, however, MeSH would require a considerable adjustment before it can be used for PHI leak detection. For example, the top hierarchical terms are too general: Anatomy, Endocrine system (level 1) and Bladder, Work Schedule Tolerance (level 3) do not contribute much to the knowledge of a personal health situation. The bottom level terms might be informative only to experts but not to the general population, e.g Motor Cortex (level 8), Trypanosoma cruzi(level 11). In both cases, the use of these terms could considerably increase the number of false positives. On the other hand, the un-predictability of the input data makes it desirable for our tool to use terms belonging to all the categories, perhaps mapping it to only one of the assigned fields.

To support our claim, we used topical descriptors (DC= 1) represented by the Main Heading field, without selecting specific hierarchical level. We filtered out the stop words and frequent content words. This left 21470 terms (Corneal Ulcer, Fibromyalgia). We applied

[20] `http://www.nlm.nih.gov/mesh/`

them to classify a sample of 50 probable files. All of the PHI files were detected, but the number of false positive files was very high. 12 files were identified as containing PHI, whereas the correct number was 4 files (Table 7). Publisher, technology are examples of MeSH terms contributing to the file misclassification.

| | $HI_m$ =1 | $HI_m$ =0 |
|---|---|---|
| PHI=1 | 4 | ? |
| PHI=0 | 8 | ? |
| | $n^+$ | $n^-$ |

**Table 7:** *MeSH: classification of the* 50 *files*

In this case, $\hat{TDP} = 8.00\% = \hat{TDP}_i$, $\frac{\hat{TDP}}{\hat{TDP}_i} = 1$. On the other hand, $\hat{FRP} = 16.00\%$ shows over-inclusiveness of the test. A small number of examples, however, does not allow for conclusive remarks. We plan more experiments when new files will be gathered.

# 5 Related Work

**Information Leak Prevention** From a data management perspective, our problem belongs to information leak prevention, a part of *data leak prevention*. Intentional and un-intentional leaks of data have become a major issue for businesses, end users, software and network providers, etc. Many companies (*Symantec* [21], *Websense*[22], etc.) concentrate their efforts on building tools able, ideally, to prevent or, at least, minimize such leaks. These tools are based on organizational policies and identify, monitor, and protect data at rest, in motion, and in use [23]. Information leak prevention is concerned with content analysis of data. Information leak prevention tools are deployed in banks, financial companies, government organizations [4]. While processing free-form text data, some of the tools apply NLP methods to enforce safer data management [4]. Many of those tools work on specific text structure and type. Our tool, on the other hand, contributes to the solution of a specific task (i.e., prevention of PHI leaks) without constraining this solution to predefined text structure or types.

The applied research community participates in tool development for information leak prevention [13, 19]. *Microsoft Research* developed a *defensive* tool that looks for personally identifiable information in one's own documents. The tool processes digital documents, including metadata, and removes the owner's name, username,security ID, computer NetBIOS name, names of online, email, webmail servers, etc. No NLP or TDM techniques are involved: all documents are treated as flat byte streams. First, the tool collects potentially sensitive information from the computer, then searches documents for its presence. The tool is semi-automated. User intervention is required to reduce false positives, i.e., non-sensitive information wrongly labelled as PII [5]. In related efforts, academic groups mostly work on hospital record de-identification; more details follow in the next paragraph. Few teams are actively involved in health information leak prevention outside of the de-identification of hospital records. In [8], the authors propose a method which detects the inference of sensitive information in documents. The method relies on search engines to find the most frequent association of topic-based terms. In [26], the author describes a method which warns web site owners if posted personal information enables identity theft (e.g., date of birth, address, name).

We, instead, focus on leak prevention techniques able to detect information within heterogeneous texts.

**De-identification of personally identifiable information** So far, PHI detection attracted only limited attention from Text Data Mining and NLP communities. Mainly, the work has been restricted to preparation of hospital records for future use by other researchers, i.e., the secondary use of health data. In Europe and North America, the law requires removal of personally identifiable information, *data de-identification*, before permitting documents for secondary use [11]. The privacy protection requirements made de-identification popular among NLP, Information Extraction, and Machine Learning applications implemented on hospital records and other PHI texts.

Typically, a de-identification method is designed for one type of documents, e.g. discharge summaries [27, 28]. De-identification consists of the detection of patients' personally identifiable information and its subsequent transformation. De-identification tasks are restricted to detection and transformation of PII, and avoid the analysis of terminology concerning health conditions of patients. The reported systems' primary methods are look-ups of person and geographic name dictionaries. Their performance, thus, depends on comparability of the dictionaries and the input data. Table 8, adapted from [22], compares the performance of a publicly available de-identifier Stat De-id when it uses customized dictionaries (the left part) and without them (the right part). The results were obtained on the de-identification of nurse notes.

| Customized dictionaries | | | Non-customized dictionaries | | |
|---|---|---|---|---|---|
| *Fscore* | *Pr* | *R* | *Fscore* | *Pr* | *R* |
| 84.4 | 74.9 | 96.7 | 77.4 | 72.3 | 83.4 |

**Table 8:** *Classification (%) of person names, health care provider names, addresses, age-related dates.*

Unfortunately, many publications do not report on disambiguation results, e.g. person and geographic names (Washington - surname, city, or state? Sofia - name or city? Marina - name, street or area?), or

[21] http://www.symantec.com/index.jsp
[22] http://www.websense.com/content/home.aspx
[23] http://media.techtarget.com/searchFinancialSecurity/downloads/Understanding_Selecting_DLP_Solution.pdf

person and trade marks (Tim Hortons - a coffee chain or a person? Jo Malone – a private label or a person?). Withholding of disambiguation accuracy makes it difficult to correctly assess the tool's performance.

We, on other hand, focus on heterogeneous texts whose content and context is not determined before our system processes them.

**Health care information extraction** There are few publications dedicated to health care information analysis in free-form, unstructured texts. Presented work often focuses on specific, rather narrow information categories. In [17], the authors compare four commercial tools which extract medication name, route, dose (number-based), strength (number-based) , and frequency (number-based) from discharge summaries and family practice notes. In [29], the authors focus on detection of obesity-related diagnostic information. They used NLP methods to extract 16 obesity diagnoses from dictated physician documentation. We, instead, opt for detection of all the HI categories.

# 6 Discussion

In recent years, Text Data Mining and Natural Language Processing communities have concentrated their efforts on the analysis of medical, biomedical and bioinformatics texts. With educational and research medical publications rapidly increasing (for some types, the increase fitting an exponential curve [9]), machine-readable lexical and knowledge sources were built to promote mining of medical texts: MedLine[24], GENIA corpus[25], MeSH[26], to name a few.

The services of medical and allied professionals are offered through Health Care[27], the industry which provides the prevention, treatment, and management of illness and the preservation of mental and physical well-being. Although medicine and health care are closely related, the domains produce remarkably different text data. A bulk of texts containing medical information comes from articles in medical journals, magazines, professional blogs, research publications. These are formally written, well-edited, *knowledge-rich* texts. Texts containing PHI are mostly internal reports, letters and various forms of personal communication. Often, they offer a description of the individual's health and lifestyle and related information such as treatments they are receiving and drugs they are taking. The texts are written without adherence to requirements of formal editing. Sometimes they are unedited, containing grammatical and lexical irregularities. For example, hepatitis can be shortened as hep, and future actions can be described as Assess/plan. Recent publications show the increased demand for automated Health Care text processing; for example, see the Journal of the American Medical Informatics Association [28]. However, there are no readily available lexical resources that cope well with Health Care text characteristics. The existing medical resources may require adaptation. Their "as is" application may give insufficient results in terms of effectiveness (missed relevant information, whereas non-relevant information captured) and efficiency (a long processing time and extra computational resources ).

For example, the Medical Entities Dictionary (MED)[29] is an ontology containing approximately 60000 concepts, 208000 synonyms, and 84000 hierarchies. This powerful lexical and knowledge resource is designed with medical research in mind, as opposed to detection of personal health information which may require a more concise knowledge base. We borrow Figure 6 from the MED web site. [30] It shows the term Plasma Glucose Test with its relationship to other terms in the MED database. Solid lines connect it to parents in the *isa* hierarchy, broken lines are nonhierarchic semantic links.

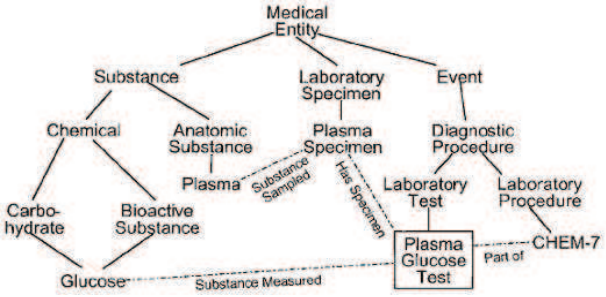



**Fig. 6:** *The term Plasma Glucose Test and relations to other terms in the MED database (adapted from the MED web site)*

Consider a text which contains the term Plasma Glucose Test. When referring to an individual, the term indicates examination for diabetes. There are two types of the test. Random Plasma Glucose Test refers to a simple blood sugar test. No fasting or glucose administration is required. Results are processed within 24 to 48 hours or faster. Fasting Plasma Glucose Test is more demanding: the patient should avoid food or drink, except water, for at least 12 hours prior to the procedure [15]. Both test types and the generic term Plasma Glucose Test relate to the physical health of a person, thus, are HI (see Section 2, point 1). CHEM-7, metabolic panel testing[31], is a more general term which indicates 7 possible tests (glucose, serum sodium, serum potassium, etc.). The term is HI, although it is less revealing than Plasma Glucose Test.

Other terms, e.g., Bioactive Substance, Plasma, Event, may or may not reveal HI, depending on the context. Searching texts for all the sixteen terms in-

[24] http://www.nlm.nih.gov/databases/databases_medline.html
[25] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/home/wiki.cgi
[26] http://www.nlm.nih.gov/mesh/
[27] http://medical-dictionary.thefreedictionary.com/health+care
[28] http://www.jamia.org/
[29] http://med.dmi.columbia.edu/
[30] http://med.dmi.columbia.edu/struc.htm
[31] http://www.nlm.nih.gov/medlineplus/ency/article/003462.htm

creases the computing time by a factor of eight (the near linear processing was confirmed in laboratory testing). The extra time might not be a problem for topic classification and text mining which process published and otherwise legitimately dispensed documents. We, on the other hand, want to limit information exposure to a reasonable minimum. The excessive terms can also increase the probability of texts being falsely tagged PHI.

We opted to build a HI lexical source from the knowledge sources used by medical and allied professionals in health care organizations. The two primary sources are the International Classification of Diseases (ICD9 codes)[32], and Canadian Drug Product Database[33].

# 7 Future Work and Conclusions

The separation of possible and impossible containers is a key factor of our tool performance. To insure high accuracy, we want to implement a more diligent testing of the file titles. In future, before being fed into the `Amazon.com` search, the titles will be pre-screened to find possible legal and health-related documents (Jane Doe letter of assessment, My affidavit). We plan to use stemming during this pre-screening. The sought after key words belong to two groups: authorized and unauthorized evidence (e.g., affidavit, permission, statement) and health records (e.g., discharge, hospital, referral). We also plan to extend geographic information analysis: (i) add more categories, e.g. population hubs such as major airports (Heathrow, Pearson), international resorts (Varna, Saltsburg), etc.; (ii) rank the found names according to their contribution for person identification; for example, New York, pop. $> 8,300,000$, can be ranked lower than Ottawa, pop. $812,000$. In future, we want to reinforce person names with statistical evidence of their use, e.g., a reverse rank on the list of popular North American names. These techniques should allow file ranking with respect to a potential risk of information leaks. In this study, we focus on contents seen by end users and do not collect the hidden file metadata. We may want to investigate the metadata impact on the scale of information leaks.

By all means, we also plan to continue testing the tool. A restricted number of PHI files can make our tool prone to data over-fitting. Hence, we continue to gather new data samples. Our future work may include analysis of the MeSH hierarchical levels, in order to reduce the number of false positive examples. We also want to use BNC to find frequent collocations, as filtering these out from the PHI indicator dictionary can, too, reduce the false positives. Other directions of future work are related to detection of rare events (e.g., a rare coma complication Lock-in). If found in text, such event can correct identify a person. However, automated extraction of rarely supplied information is difficult and may require expanding our system with a new element.

We have introduced a system which helps to prevent leaks of personal health information. Our system is able to work within the complex environment of previously unseen data types. It prevents the leakage of PHI texts in heterogeneous input, i.e. files in which context, content, and type vary unlimitedly. The uncertain content of the files contrasts our data set with homogenous data sets of known content; for example, company employee files and hospital records, where each file is pre-disposed to contain personal information, or movie script archives, where scripts are a work of fiction. To accommodate the uncertainty, we have introduced a taxonomy of files related to the possibility and probability of the files leaking personal health information.

On empirical evidence, we have shown that the medical sources are well-suited to analyze formally written, well-edited texts, often with an abundance of scientific terms and relations. For our task, however, the sources contain excessive information, making text analysis too slow, inefficient and prone to false positive identification. A series of experiments was performed on files exchanged in peer-to-peer file sharing networks with encouraging results.

**Acknowledgements** This work has been funded by the Natural Sciences and Engineering Research Council of Canada and the Ontario Centre of Excellence. We thank Terry Copeck for his assistance with text extraction. We thank anonymous reviewers for their helpful comments.

[32] `http://www.cdc.gov/nchs/about/otheract/icd9/abticd9.htm`

[33] `http://www.hc-sc.gc.ca/dhp-mps/prodpharma/databasdon/index-eng.php`

# References

[1] Personal Health Information Protection Act. Legislation of Ontario, 2004. `http://www.e-laws.-gov.on.ca/html/statutes/english/elaws_statutes_04p03_e.htm`, accessed Sept. 7, 2008.

[2] The International Classification of Diseases, 9th revision, clinical modification, 2007. National Centre for Health Statistics, Centres for Disease Control and Prevention, US Government.

[3] Review of systems for extracting and anonymizing geographic information. Technical report, Electronic Health Information Lab, CHEO, submitted to GeoConnections, 2008.

[4] Understanding and selecting a data loss prevention solution. Technical report, Securosis, L.L.C, the SANS Institute, 2009.

[5] T. Aura, T. Kuhn, and M. Roe. Scanning electronic documents for personally identifiable information. In *Proceedings of the 2006 ACM Workshop on Privacy in the Electronic Society (WPES 06)*, pages 41–50, 2006.

[6] M. Baird. Personal files were accessible for more than three weeks. The Western Star, 2008.

http://www.thewesternstar.com/index.cfm?sid-=104156&sc=23, retrieved Feb 5, 2009.

[7] L. Bouma and M. de Rijke. Specificity helps text classification. In *Proceedings of European Conference on Information Retrieval (ECIR 2006)*, pages 539–542. Springer, 2006.

[8] R. Chow, P. Golle, and J. Staddon. Detecting privacy leaks using corpus-based association rules. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 893–901, 2008.

[9] J. DeShazo, D. LaVallie, and F. Wolfe. Publication trends in the medical informatics literature: 20 years of 'medical informatics' in mesh. *BMC Medical Informatics and Decision Making*, 9(7):e13, 2009.

[10] P. Dixon. Medical identity theft: The information crime that can kill you. The World Privacy Forum, 2006. http://www.worldprivacyforum.org/medicaliden-titytheft.html, retrieved June 7, 2009.

[11] B. Elger and A. Caplan. Consent and anonymization in research involving biobanks. *European Molecular Biology Organization reports*, 7(7):661–666, 2006.

[12] K. E. Emam, E. Neri, and E. Jonker. An evaluation of personal health information remnants in second hand personal computer disk drives. *Journal of Medical Internet Research*, 9(3):e24, 2007.

[13] A. Evfimievski, R. Fagin, and D. Woodruff. Epistemic privacy. In *Proceedings of the 27th ACM Symposium on Principles of Database Systems (PODS 2008)*, pages 171–180, 2008.

[14] J. Gayer. Policing privacy: Law enforcement's response to identity theft. CALPIRG Education Fund, 2003. http://www.calpirg.org/home/reports/reportarchives, retrieved June 7, 2009.

[15] F. Hecht and W. Shiel, editors. *Webster's New World Medical Dictionary*. Wiley Publishing, second edition, 2003.

[16] T. Herzog, F. Scheuren, and W. Winkler. *Data Quality and Record Linkage Techniques*. Springer, 2007.

[17] V. Jagannathan, C. Mullett, J. Arbogast, K. Halbritter, D. Yellapragada, S. Regulapati, and P. Bandaru. Assessment of commercial NLP engines for medication information extraction from dictated clinical notes. *International Journal of Medical Informatics*, 78(4):284 – 291, 2008.

[18] E. Johnson. Data hemorrhages in the health-care sector. In *Financial Cryptography and Data Security*, 2009.

[19] R. Jones, R. Kumar, B. Pang, and A. Tomkins. Vanity fair: privacy in querylog bundles. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM08)*, pages 853–862, 2008.

[20] J. Long. *No Tech Hacking: A Guide to Social Engineering, Dumpster Diving, and Shoulder Surfing*. Syngress Press, 2008.

[21] C. McGuigan and M. Browne. Hospital leak linked to witness in lvf case. Belfast Telegraph, 2007. http://www.belfasttelegraph.co.uk/sunday-life/news/hospital-leak-linked-to-witness-in-lvf-case-13904797.html , retrieved June 7, 2009.

[22] I. Neamatullah, M. Douglass, L. Lehman, A. Reisner, M. Villarroel, W. Long, P. Szolovits, G. Moody, R. Mark, and G. Clifford. Automated de-identification of free-text medical records. *BMC Medical Informatics and Decision Making*, 8(32):e17, 2008.

[23] M. Pepe. *The Statistical Evaluation of Medical Tests for Classification and Prediction*. Oxford University Press, 2004.

[24] C. Preimesberger. Cyber-criminals use p2p tools for identity theft, security analyst warns. eWeek.com, 2006.

[25] M. Sokolova and K. El Emam. Evaluation of learning from screened positive examples. In *Proceedings of the 3rd workshop on Evaluation Methods for Machine Learning (EMML-ICML 2008)*, 2008.

[26] L. Sweeney. Protecting job seekers from identity theft. *IEEE Internet Computing*, 10(2):74–78, 2006.

[27] O. Uzuner, Y. Luo, and P. Szolovits. Evaluating the state-of-the-art in automatic de-indentification. *Journal of the American Medical Informatics Association*, 14:550–563, 2007.

[28] O. Uzuner, T. Sibanda, Y. Luo, and P. Szolovits. A de-identifier for medical discharge summaries. *Journal of Artificial Intelligence in Medicine*, 42:13–35, 2008.

[29] H. Ware, C. Mullett, and V. Jagannathan. Natural Language Processing (NLP) Framework to Assess Clinical Conditions. *Journal of the American Medical Informatics Association : JAMIA*, 16:585–589, 2009.

# Author Index