FACILE: Classifying Texts Integrating Pattern Matching and Information Extraction[1]

Fabio Ciravegna
Alberto Lavelli
Nadia Mana
ITC-irst
Loc. Pantè di Povo
38050 Trento Italy

Johannes Matiasek
ÖFAI
Schottengasse 3
1010 Vienna Austria

Luca Gilardoni
Silvia Mazza
Massimo Ferraro
Quinary SpA
Via Fara 35
20124 Milan Italy

William J. Black
Fabio Rinaldi
David Mowatt
UMIST - PO Box 88
Manchester M60 1QD
United Kingdom

Abstract

Successfully managing information means being able to find relevant new information and to correctly integrate it with pre-existing knowledge. Much information is nowadays stored as multilingual textual data; therefore advanced classification systems are currently considered as strategic components for effective knowledge management. We describe an experience integrating different innovative AI technologies such as hierarchical pattern matching and information extraction to provide flexible multilingual classification adaptable to user needs. Pattern matching produces fairly accurate and fast categorisation over a large number of classes, while information extraction provides fine-grained classification for a reduced number of classes. The resulting system was adopted by the main Italian financial news agency providing a pay-to-view service.

1 Introduction

Knowledge is nowadays the key source for competitive advantage. The success or failure of a company can depend on the ability to find the right information at the right time. The www explosion (and the increasing usage of Internet technologies as a core channel for communication) multiplies the sources of information and increases by orders of magnitude the amount of information available. However, while raising the opportunities for gaining competitive advantages, this also increases the information glut. The main value is not in the information itself, but in the capability of managing it successfully to derive knowledge that is critical to an organisation's objectives. Successfully managing information means being able to correctly integrate it with existing structured information, to facilitate communication and knowledge sharing and to support knowledge-based organisations.

The role of natural language processing and artificial intelligence is fundamental in this respect as: i) the vast majority of this information is textual and available in different languages; ii) the development of new tools for structuring textual data starting from its content represents one of the fundamental steps in successfully managing information.

This is particularly evident in the business arena, where on-line textual information from news providers has long since been available and heavily used. Recent reports from Gartner Group [Bair, 1998] explicitly mention advanced classification systems characterised by semantic technologies as the most strategically relevant element to support effective knowledge management. However, technology available on the market still mainly resorts to information retrieval-derived systems and techniques (IR). This technology does not provide adequate accuracy when coping with rich and complex classification structures. This is because IR systems do not take into account linguistic features. More linguistically oriented text classification can be achieved by the use of pattern matching (PM). PM can produce a fairly accurate and fast categorisation over a large number of classes [Hayes and Weinstein, 1991; Jacobs and Rau, 1990]. Resource development does not require linguistic expertise and can be done by trained users. But PM is still weak on the analysis of linguistic structures and cannot be used for fine-grained categorisation.

Information extraction (IE) techniques can be also used for text classification (e.g., the Text filtering subtask of the ST task in [MUC6]). IE systems perform very well in detecting texts relevant for a single class (e.g., management succession) with results ranging between 80-95 for both precision and recall. But IE cannot be performed on a large number of classes: it is an expensive technology as it requires a large amount of time of linguistically aware personnel [Grishman, 1997]. Attempts at separating linguistic knowledge (e.g., syntactic knowledge) from domain dependent knowledge (e.g., the domain patterns) [Grishman, 1997; Hobbs *et al.*,

[1] This work was partially funded by the European Union in the framework of the Language Engineering Sector, project FACILE (LE 2440). ÖFAI is supported by the Austrian BMWV.

1997] simplified the task, but did not solve the general problem. Moreover IE systems are still not efficient enough to cope with large amount of texts.

This paper proposes the integration of PM and IE as a method for providing classification highly adaptable to different user needs. Pattern-based (shallow) classification performs broad-band text filtering, comparable to that done by a human quickly skimming texts, i.e. recognising main topics without careful reading. IE is used to provide more fine-grained classification, comparable to that produced by a careful human reading. IE is used to cope with only some of the classes, i.e. those on which shallow classification results need to be refined.

Classification provided in this way is very adaptable to user needs as it provides flexibility in different directions: broad/fine-grained classification, large/restricted number of classes, high/low efficiency in the process. It is also flexible in terms of application development. PM rules take less time to develop than IE resources: a PM-based application can be provided as a first (although not complete) answer, waiting for a more complete integrated PM+IE application to be developed.

The FACILE system (Fast and Accurate Categorization of Information by Language Engineering) is based on this two-tiered approach. It is able to categorise in detail business texts such as agency news, newspaper articles and analysis reports from different sources and written in different languages (currently English, German, Italian, and - to some extent - Spanish). The system has been developed as an applied research project, bringing together researchers and end-user organisations. It was delivered as a pre-engineered prototype and evaluated by three users participating in the project (an Italian rating agency, a Spanish savings bank, and a German information broker). The system has been adopted by the main Italian financial news agency for production use. It currently provides classified news both for internal work organisation and as an external pay-to-view service.

In this paper we describe the system, focusing on architectural and evaluation issues. We believe that our approach is interesting because, although it does not propose completely new technologies, it demonstrates that innovative technologies can be profitably integrated to build applications providing high added-value services.

2 System architecture

The system architecture has been driven by the idea of providing classification highly adaptable to user needs. The first need addressed is that of providing a refinable classification strategy, resorting to information extraction only when needed. In general classification is obtained via a multi-step process, where each step adds information to the input. A control module activates the different modules in order to assign the appropriate level of classification needed by the user for each text. To do that the control module uses the information progressively augmented by the different modules to each text. It is this module that decides whether or not IE is to be performed on a specific text. The other modules in the architecture are the preprocessor, the shallow analyser, the deep analyser and a group of application-specific modules. From the point of view of flexible classification **the preprocessor** implements tasks that are to a large degree application independent, such as segmentation, normalisation of numbers, dates and abbreviations, as well as morphological analysis, part of speech tagging, preliminary unknown word guessing and named entity recognition. It does not require major adaptation for new applications. **The shallow analyser** performs a preliminary classification by exploiting techniques that can be easily tailored to new applications by trained personnel; it is based on pattern matching and assigns a set of classes to each text. **The deep analyser** performs more fine-grained classification via IE. It produces filled templates pertaining to the categories selected by the shallow analyser. It can be ported to new classes by linguistically-trained personnel only. **Application layers** are constituted by an additional group of modules that permits the adaptation of the system to different user environments. In the rest of this section we will analyse each module separately.

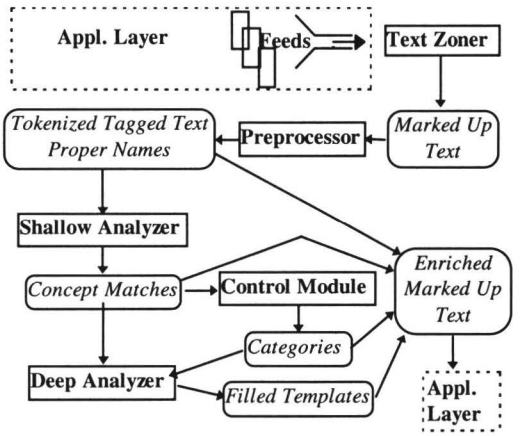

Figure 1: general architecture

2.1 Preprocessor

The preprocessor transforms a stream of characters into analysed tokens. It provides morpho-syntactic information and a disambiguated syntactic tag for each token in the input, treating proper names as single tokens and assigning them a (disambiguated) semantic tag. The key steps are the same for all the languages considered (currently English, Italian, German and Spanish) and use the same machinery with language-specific resources. The steps are: (1) tokenisation (splitting of input text, assignment of orthographic features); (2) morphological analysis (stemming and assignment of morpho-syntactic information); (3) part of speech tagging (disambiguation of multiple morpho-syntactic categories); (4) database lookup (assignment of semantic fea-

tures); (5) named entity analysis (NEA). Steps 2 and 3 integrate generic resources and machinery provided by InXight Inc. LinguisticX. Database lookup adds semantic features (e.g. person, organisation, location etc., but also auxiliary categories, such as verb-of-speech, used as clues in NEA) for (possibly multi-word) tokens in the database. The information in the database is to be regarded as provisional and necessarily incomplete. For example, while "Pearl" may be in the database as a first name, "Pearl Harbor" is not a person name. Therefore more informed methods for correctly identifying named entities are needed. NEA uses non-deterministic chart parsing. The approach used has been partially suggested by [Coates-Stevens, 1992]. NEA recognises complete names as well as numbers or time expressions. Unlike other approaches (e.g., [Mikheev *et al.*, 1998]), a high-level language is used for the definition of the patterns. NEA uses language-specific context-sensitive rules that can take into account any information placed in the chart by preceding modules. NEA employs a weighting mechanism that enables it to either return all competing analyses or to select the most plausible one [Black *et al*, 1998]. The preprocessor input is a zoned text, i.e. with parts to be analysed identified and marked up. Its output is a text where each (group of) word(s) is associated with orthographic and morphological features, normalised form and semantic labels (e.g. "John Smith" is a "proper name" with orthographic class "capitalised", feature "masculine" and semantic tag "person") .

2.2 Shallow Classifier Module

The shallow classifier (SCM) is based on pattern matching. It operates in two stages: first it recognises domain relevant concepts mentioned in texts, and then assigns categories to the text. The first step is performed by the **shallow analyser**. It is based on a refined version of the model presented in [Gilardoni *et al.*, 1994], in turn derived from seminal work by [Hayes and Weinstein, 1991]; the shallow analyser models both domain objects (e.g., shares, market sectors, balance sheet items) and domain events (e.g., joint ventures, public offers, financial transactions). Its pattern matching techniques are integrated with a knowledge representation system. Concepts are associated with a set of weighted linguistic patterns. The pattern language is similar in spirit to those implemented by search engines (e.g., use of Boolean operators, proximity, optionality), but is augmented to enable testing of lexical variants, exploiting information from the preprocessing stage. It is also tightly coupled with the knowledge representation language, allowing references to concepts in patterns, exploiting inheritance and relations to reduce complexity. The result of shallow analysis is a set of matches of text portions with concepts with scores, expressing confidence factors. Matches maintain reference to the part of text matched enabling them to be used for annotation of texts. The second step is performed by a rule-based **categorizer**. The text's main topic is determined using application-specific heuristics (based on both concept matches and context), as well as domain knowledge. The shallow analysis and categorizer use similar but distinct concept lattices, implemented in the same KR formalism. The categorizer's lattice directly reflects the end-user's perception of the application-specific conceptual space. Both are domain-dependent, but application specificity is kept confined in the category lattice, resulting in a domain model largely sharable by different applications coping with different classification structures. The patterns that characterise concepts are obviously language dependent. But they are kept associated to concepts, i.e. they are domain motivated and not linguistically motivated. Developing a multilingual application they can be handled in an homogeneous way.

The whole machinery is supported by modular declarative resources to describe concepts, patterns, categories and classification rules. The modularization of these resources allows easy development, maintenance and portability across both languages and applications. It also diminishes the major drawback of a knowledge-based approach (i.e., the complexity in building the knowledge base), while retaining the major benefits (i.e., attainable accuracy and consistency).

On the other side, alternative classification criteria or a different category lattice imply modification to the categorizer's resources (rules) but not necessarily to the domain knowledge encoded in the concepts network or in the associated patterns.

2.3 Deep Analyser Module

The Deep Analyser Module (DAM) performs information extraction to achieve fine-grained classification. It classifies texts by their content, for example by allowing the user to express a query such as "send me any texts concerning bonds issued by European-based financial institutions whose amount exceeds 1 million Euro". DAM's task is similar to the ST task in MUC conferences ([MUC, 1998]), but the kind of templates needed for classification are generally flatter than those used in MUC (see the section on evaluation for an example). DAM is based on cascades of Finite State Transducers (FSTs). The global architecture is depicted in figure 2. DAM receives as input the pre-processed and classified text and produces filled templates. The whole architecture uses the same formalism, and the same set of primitives for all FSTs. Each FST operates on string of *tokens*. Tokens are abstract representations of lexical elements that have three types of realisations accessible at any time during analysis: lexical (strings, NEs, etc.), syntactic (parse trees, Typed Feature Structures) and semantic (Quasi Logical Forms, QLFs) [Ciravegna and Lavelli, 1999]. The first module applied is the parser. It assumes that there is one and only one possible correct parse tree for each sentence. Its goal is to produce a *sufficient approximation* of such a parse tree, i.e. a tree that captures all the relations use-

ful for information extraction, while leaving the others implicit. Parsing is performed in three steps: identification of NPs, VGs and PPs (chunking), A-structure recognition (i.e., recognition of subcategorization frames of verbs, some NPs, etc.), and modifier attachment. The output of the parser is a parse tree with the relevant modifiers attached, a Typed Feature Structure representing relations among constituents in the parse tree and a QLF. The accuracy in producing the correct A-structure reached P=95%, R=71% in a blind text on 62 texts [Ciravegna and Lavelli, 1998]. Following parsing, default reasoning is applied to introduce in the QLF additional information not explicitly contained in the text, but needed for template filling (e.g., "if a person working for company X is hired by company Y, s/he is no longer employee of X").

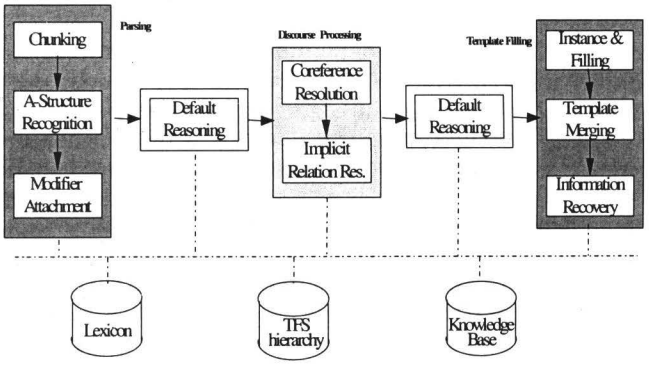

Figure 2: DAM architecture

After this, discourse processing is performed: (pro)nominal references are resolved for both objects (people, organisations, physical objects) and events (e.g., hiring/firing); implicit relations are also captured (e.g., "The Bank of Japan decided .... The **president** said ...."). High accuracy is reached in discourse processing as the amount of available information (i.e., parse tree with associated information in the Typed Feature Structures and in the QLF) allows reliable choices among candidates. Templates are finally filled by using the final QLF (as produced by an additional default reasoning step). Template merging and recovery actions cope with missing information.

Portability was a main requirement for DAM. The module can be ported to new domains by just modifying declarative resources: the lexicon, the knowledge base and the FST grammars. The grammars are sequences of cascades of FSTs. All grammars use the same formalism and primitives (15 in all). This uniformity is an important advantage, allowing new applications to be developed by a single person. The distinction between domain dependent and domain independent resources simplifies the porting; currently there are 14 FST cascades: 6 are domain independent, 7 are domain dependent and one is user dependent (i.e., the template presentation module). Porting to new applications required one to three man months, depending on the application.

2.4 Around the Kernel: Application Layer

One of the main user-needs is the integration of the system directly into the user environment. For this reason the system architecture is divided in two main blocks: the Classification Kernel (composed by the four modules mentioned above) and the application layer.

The Classification Kernel is currently deployed as a standalone Unix module, configurable within a full development environment or stripped down to a runtime executable, accessible by application layers using different communication protocols, including sockets and http. An NT version is currently in an alpha release.

The classification kernel is concerned only with text analysis, expecting as input a normalised, tagged text and providing as output an annotated text. Input and output tagging is currently based, due to historical reasons, on HTML extended to carry structural information, with work on XML support ongoing. Both formats are dynamically configurable to support different text sources . The application layer is responsible for interaction with the real world, i.e. of feeding the categorizer and of handling categorisation results to make them available to final users. Application can be extensively customised or modified to properly integrate within different Information Technology infrastructures. Four applications have been so far implemented with the system, spanning four languages and with different scopes over the classification domains. All of them are different in terms of IT integration, but all share a common model with a source and annotated text repository (based on Oracle, Fulcrum and in two cases on flat files in a shared file system) residing on either Unix or NT systems, simple feeders alimenting the kernel running on connected Sparc, and different intra/internet interfaces towards annotated text repository.

3 EVALUATION

The system was evaluated on different corpora for different languages in different application environments. It was subject to extensive albeit informal tests during development using corpora composed by (in at least one case) several thousands texts. The three analysis modules were formally evaluated against user tagged corpora by using automatic tools either acquired, such as the MUC scorer, or internally developed. In the following we will focus on the formal evaluation first, discussing each module separately. Results are not independent, as the modules operate in a chain: e.g. results of the shallow classifier discount the errors introduced by the preprocessor. However, as soon as former modules start to deliver reasonable results (e.g. in line with those reported below), chain effects become negligible and, in some case, could be recovered at a later stage. The advantage given by the integration of pattern matching and IE techniques is moreover not measurable in terms of percentage gains. Being able, for example to discriminate bond issue story depending on amount

threshold is currently completely out of reach without IE techniques - while on the other side not being able to properly drive IE has generally dramatic effect as well on both precision and recall.

3.1 Preprocessor

Formal evaluation of the preprocessor was done for named entity recognition in three languages (English, Italian, German). Two corpora of newswire messages were selected for each language, one used for building and enhancing the NE rules, the other one used as the corpus for a blind test. For Italian and German the corpus has been selected from sources in normal use at user's sites. For English, the test corpus chosen was the MUC-7 Dry-Run corpus, which gave better comparability with the procedure adopted with the other languages than the Formal Run, in which domain differed between training and testing. Annotation of these corpora was performed manually in the form of SGML markup using appropriate tools. Preprocessor results were delivered by the system in SGML-annotated form; these results were compared to the pre-annotated version using the MUC scoring scheme. See Table 3 for evaluation results.

		#words	#NE	Prec	Rec
training	English	81,704	12,922	.94	.92
	German	10,887	1,534	.97	.96
	Italian	9,200	554	.96	.96
test	English	18,084	2,716	.91	.94
	German	21,734	2,936	.94	.85
	Italian	5,700	395	.86	.92

Table 3: preprocessor evaluation

3.2 Shallow Categorizer Module

The shallow categorizer has been evaluated formally in a blind test for Italian and German, on two specific applications. For both languages a test set was selected by end users from usual text sources the system was expected to analyse (agency news and financial newspaper); the blind test was performed on 75 texts for Italian and 50 for German. An English version of the Italian application was also evaluated with comparable results, albeit in a less formal way using material from Reuters corpus (http://www.research.att.com/~lewis/) re-annotated with respect to the category structure handled by the system. The Italian application was required to classify texts against a set of about 100 categories arranged in a hierarchy. The classification ranged over several financial domain topics. Top-level elements were company news (especially company results, capital and stake operations, ratings, bond issues, management changes and meetings), sector news (refined by specific market sectors and behaviours), and stock market related news. The German application classified over a hierarchy describing country economic indicators (inflation, employment, etc.) also discriminating by country and by whether the indicator figures were mentioned as measured data or as forecast.

A first measure for precision and recall was taken, considering each category in isolation, i.e. independently of its position in the hierarchy. On this basis, whenever a superordinate category is assigned by the system, this is marked as an error for both the most specific ones (lowering the recall) and for the more general assigned (lowering the precision). The table below summarises the results obtained; please note that English figures were obtained with largely underdeveloped resources:

	# texts	Ass.	Corr.	A not C	C Not A	Prec	Rec
German	50	121	110	20	9	.83	.92
Italian	75	304	299	26	21	.91	.93
English	267	315	340	44	69	.86	.80

Table 4: shallow classification evaluation

As categories are defined hierarchically they do not induce disjoint partitions. This means that whenever the system 'takes risks' in assigning more specific categories, e.g. wrongly assigning 'annual results' whenever the correct category should have been the more general 'economic results' this is not an error with respect the more general one. Analogously, if the application 'stays conservative' and misses a more specific category, the assignment of an ancestor should count as evidence for the more specific category anyway. Taking this into account, as in [Gilardoni and Rocca, 1995], higher precision and recall than above are obtained for intermediate categories. Although less significant from the point of view of system comparison, it is however far more significant from the user point of view, as whenever the hierarchy is, as in our case, structured and deep, the user would likely exploit the structure querying at intermediate levels.

3.3 Deep Analyser Module

For Italian, one template filling was fully developed (Bond-issue) and two others have reached the level of demonstration (management succession in companies and company annual results). For English a demonstrator for the field of economic indicators was developed. Here we will focus on bond issues in Italian. The information to be extracted for each bond issue included: issuer, kind of bond, amount, currency, placement date, interest date, maturity, average duration, global rate and first rate. The corpus (used both for training and test) included 95 texts (agency news and newspaper articles, globally 10,472 words). Number of slot fills per text: 6.4. The test was performed as follows: first the directives for the annotator were developed and the texts tagged by a person; then DAM was instructed to operate in the domain (by just modifying declarative resources) and run on the corpus. Finally DAM results were compared against the user defined answers by using the MUC scorer [Douthat, 1998]. The development cycle was organised as follows: DAM resources were developed by carefully inspecting the first 30 texts of the

corpus. Then the system was compared against the whole corpus (95 texts): results were Recall=.51, Precision=.74, F(1)=.60. Then the resources were tuned on the whole corpus, focusing on the texts that did not reach sufficient results in terms of R&P. The final results on the whole corpus were P=.80, R=.72, F(1)=.76.

3.4 User Validation

Formal evaluation enables system comparison and also provides useful information for technical development. Nevertheless, real evaluation comes by user validation (is this the right system?) and by real usage. End users from financial institutions were involved at each stage in development, with feedback incorporated in prototypes released. The adoption of the system for production use by a major Italian news agency, not involved in development and moreover putting the system at work already when only a preliminary and largely incomplete prototype was available, is a clear proof of the validity and usefulness of the approach.

4 Conclusion

We have described the design and evaluation of an advanced system for fine-grained text categorisation. The system is highly modular, both in its processing filters and in its declarative resources. This has permitted it to be applied to a range of application domains in four European languages. The architecture brings together PM and IE. The system has been a considerable success story: it has been adopted by the main Italian financial news agency for providing a pay-to-view internet service. The system integrated many heterogeneous software components. It has been running continuously in the user environment since January 1998. Applications at user sites involved in the consortium are also available. We contend that this shows that state-of-the-art AI techniques are mature enough to provide real world applications. The architecture is designed in order to separate knowledge intensive resources (DAM's), from less knowledge intensive modules (SCM's), from domain independent modules (preprocessor's). This approach simplifies resource development both in terms of required expertise and in terms of task complexity. In the future we will continue working in the direction of the time-to-market reduction, in order to considerably enlarge the market for fine grained classification systems. In particular we will investigate the use of machine learning techniques for resource development.

Acknowledgements

Participants to the FACILE project were: Quinary SpA, ITC-IRST, Italrating SpA (IT), ÖFAI, Sema SAE (ES), Caja de Segovia (ES), Sis (DE), UMIST-CCL. The authors would like to thank all the project team, in particular P. Prunotto, V. Fiorentino and K. Schirmer.

References

[Bair, 1998] J. Bair. *Dimensions of KM Technology Selection*, GartnerGroup publishing, October 1998.

[Black *et al.*, 1998] W. J. Black, F. Rinaldi, and D. Mowatt. FACILE: Description of the NE System Used for MUC-7. In [MUC, 1998].

[Ciravegna and Lavelli, 1999] F. Ciravegna and A. Lavelli. Full Text Parsing using Cascades of Rules: an Information Extraction Perspective. In *Proc. of the 9th Conf. of the European Chapter of the Association for Computational Linguistics*, Bergen, Norway, 1999.

[Coates-Stevens, 1992] S. Coates-Stevens. *The Analysis and Acquisition of Proper Names for Robust Text Understanding*. PhD Thesis, City Univ., London, 1992.

[Douthat, 1998] A. Douthat. *The Message Understanding Conference Scoring Software User's Manual*. In [MUC, 1998].

[Gilardoni *et al.*, 1994] L. Gilardoni, P. Prunotto, and G. Rocca, Hierarchical Pattern Matching for Knowledge Based News Categorization. In *RIAO-94, Intelligent Multimedia Information Retrieval Systems and Management*, pages 67-82, New York, October, 1994.

[Gilardoni and Rocca, 1995] L. Gilardoni and G. Rocca. Evaluating Categorization Systems; Lessons Learned from the COBALT Experience. *2nd Language Engineering Convention* - London October 1995.

[Grishman, 1997] R. Grishman. Information Extraction: Techniques and Challenges. In: M. T. Pazienza (ed.). *Information Extraction: a multidisciplinary approach to an emerging technology*. Springer-Verlag, 1997.

[Hayes and Weinstein, 1990] P.J. Hayes and S.P. Weinstein. Construe/TIS: A System for Content Based Indexing of a DataBase of News Stories. In *Proc. of the 2nd Conf. of Innovative Applications of Artificial Intelligence*, 1990.

[Hobbs *et al.*, 1997] J.R. Hobbs, D. Appelt, J. Bear, D. Israel, M. Kameyama, M. Stickel, and M. Tyson. FASTUS: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text. In: E. Roche and Y. Schabes (eds.). *Finite-State Language Processing*. The MIT Press, 1997.

[Jacobs and Rau, 1990] P.S. Jacobs and L.F. Rau. SCISOR: Extracting Information from On-line News. *Communications of the ACM*, 33(11), 1990.

[Mikheev *et al.*, 1998] A. Mikheev, C. Grover, and M. Moens. Description of the LTG System as used for MUC-7. In [MUC, 1998].

[MUC, 1998] *Message Understanding Conference Proceedings (MUC-7)*. Av. at: http://www.muc.saic.com/