Multilingual Information Extraction

Philipp Johannes Masche

Helsinki, 15th February 2004

Department of Computer Science
UNIVERSITY OF HELSINKI

HELSINGIN YLIOPISTO - HELSINGFORS UNIVERSITET - UNIVERSITY OF HELSINKI

Tiedekunta/Osasto - Fakultet/Sektion - Faculty	Laitos - Institution - Department
Matemaattis-luonnontieteellinen	Tietojenkäsittelytieteen laitos

Tekijä - Författare - Author
Philipp Johannes Masche

Työn nimi - Arbetets titel - Title of the paper
Multilingual Information Extraction

Oppiaine - Läroämne - Subject
Tietojenkäsittelytiede - Computer Science

Työn laji - Arbetets art - Type of work	Aika - Datum - Date	Sivumäärä - Sidoantal - Number of pages
Pro gradu -tutkielma	15. 2. 2004	58

Tiivistelmä - Referat - Abstract

For over a decade attention has been focused on information extraction in English. In recent years, however, the focus has shifted on to other languages. New languages are continuously introduced to information extraction, and systems now need to handle a variety of languages. Inevitably every new language brings new challenges and possibilities.

This paper examines information extraction within a multilingual environment. The paper is in two parts; firstly the components of English information extraction are outlined. Then, these components are transferred to new languages and assembled into a multilingual information extraction framework.

The components of information extraction can be induced into new languages through adaption of methods used for English language. In a multilingual environment different methods can be utilised to transfer linguistic knowledge across language barriers. Machine translation and cross-language projection across word aligned corpora especially offer alternatives when other linguistic resources grow short.

A framework for multilingual information extraction is proposed at the end of this paper, that addresses the following issues: Multilingual information extraction must recognise language automatically and single components of information extraction need to process numerous languages. The results of information extraction need to be stored either in a language dependent or independent fashion and must be translated to other languages.

Subject classification (Computing Reviews 1998): H.3.3 Information Search and Retrieval, H.3.7 Digital Libraries, I.2.7 Natural Language Processing

Avainsanat - Nyckelord - Keywords
Multilingual information extraction

Säilytyspaikka - Förvaringställe - Storage location
Tietojenkäsittelytieteen laitoksen kirjasto, sarjanumero C-2004-

Muita tietoja - Övriga uppgifter - Other information

Contents

1 Introduction

In recent years the number of texts digitally available has increased dramatically. In the early days of the world wide web the majority of these texts were English. Recently the number of non-English texts has grown considerably. These developments have opened possibilities and created the need for tools for processing texts in other languages. Information extraction is an important text processing tool to make the information in large text collections more easy accessible to humans and computer programs.

Around ten years ago information extraction started to become an increasingly intense and successful field of research. Automatic extraction of information from texts involves deciding whether a text is relevant for a certain domain, and if so extracting a set of facts from that text. Most of the known information extraction systems have been invented for texts written in the English language. Nowadays English language information extraction systems perform nearly as well as human experts. Most publications consider problems from the point of view of English language specialities. Only recently an increasing number of papers deal with non-English information extraction [AI99, Gri+99].

In comparison to the success of English language information extraction systems, information extraction systems for other languages are not as reliable, or are still lacking essential components. Depending on the number of native speakers, prosperity of countries, and the need for natural language processing capabilities, as well as due to the complexity of certain languages, information extraction systems are differently developed for individual languages. Recent research has focused on developing information extraction systems comparable to English systems. Additionally, new methods are being invented to create information extraction components for new languages. Machine translation, for instance, can be used to transfer linguistic knowledge from English to other languages. In environments where many languages are present, monolingual information extraction systems need to be assembled together into multilingual systems. Multilingual frameworks serve as an infrastructure where text can be processed in any language.

In Chapter two I give an overview of the most common information extraction tasks and the methods used in classical information extraction for the English language. With these concepts in mind, I discuss in Chapter three how information extraction is adapted to new languages. Often the classical concepts are not satisfying because other languages lack resources which are available in English. I will therefore describe methods for overcoming the resource shortages and creating information extraction components for new languages. The difference between mono- and multilinguality is distinguished and general methods for multilingual text processing tasks are discussed. In Chapter four I present a multilingual information extraction framework utilising the previously studied methods. Summary and conclusion are presented in Chapter five.

2 Information Extraction

Information extraction is the task of extracting information from natural language text. More specifically, information extraction is expected to extract the same type of information from every text document in a text corpus, and only if the text contains such relevant information. If the text does not contain relevant information, nothing is to be extracted. Often it is anticipated that results are in the form of a data tuple which can be stored in a database. The patterns for these data tuples are also called *templates*. Thus, sometimes information extraction is also referred to as *template filling*.

At a first glance information extraction might appear to be similar to information retrieval. Information retrieval, however, presents the texts as result in ranked order, which is different to the templates filled in information extraction. Automated text summarisation also appears to be somewhat similar to information extraction. During summarisation it is important to preserve the most important information in a text. Information extraction is only concerned with one specific kind of information and widely ignores others, possibly important, but for the task irrelevant information. Text categorisation solves some of the problems of information extraction. Texts could be categorised into two classes: those containing relevant information for the information extraction task, and those which do not. However, text categorisation is mainly based on statistical analysis of word frequency, while information extraction employs various formal linguistic analysis methods to access syntactic and semantic structure of the text. Information extraction can help to improve the text processing tasks mentioned above. One could imagine, for instance, that future Internet search engines use a combination of these disciplines for text retrieval. Large text collections would be easily accessible and could be sorted by the extraction results [Gri+99].

In the early days of information extraction, research focused on extraction from English text. As a result, some components of English information extraction systems perform nowadays on a level comparable to human experts. English information extraction is far ahead of information extraction systems for other languages and in this paper English information extraction will be used as an example to explain general problems of and concepts for information extraction. Certainly it will not always be possible to exactly copy information extraction concepts from English to other languages. Nevertheless, these concepts build a good foundation and understanding them will help us to derive concepts for information extraction in other languages [Gri+99].

For a decade information extraction was driven by the conferences for message understanding (MUC). These conferences helped to formalise information extraction[1].

As example information extraction tasks, the so-called *MUC tasks*, have been specified and include evaluation criteria and text corpora for testing. The MUC tasks are widely used for evaluation of information extraction systems. For MUC-1 and MUC-2 (1989) a corpus of messages about naval operations were published. For MUC-3 (1991) and MUC-4 (1992) a collection of news articles about terrorist activities in South- and Middle-America were published. The task was to extract information about place, time, perpetrators, victims and the kind of terrorist activity. For the MUC-5-(1993) a corpus of news articles about joint ventures and micro electronics was provided. The task prepared for the MUC-6 (1995) includes a collection of news articles dealing with executive succession. The MUC-7 task is based on a corpus of news articles about space vehicle and missile launches [Che03, MUC91, MUC92, MUC93, MUC95, MUC98].

There are two important aspects of information extraction. The first is to construct an information extraction system, the second to use it. Information extraction can be divided into various subtasks. Some of these are performed to build or train an information extraction system, others are used to perform information extraction, and a few are used in both circumstances. In many cases the use of one concept rules out the use of another. Differences between information extraction systems are, for instance, the thoroughness with which analysis of syntax and semantics is carried out on the text, or what general linguistic approach is favoured for linguistic subtasks. The scientific field of computational linguistics is traditionally split into two camps. One favours stochastic methods using a prepared training corpus while the other favours handcrafted systems based on linguistic rules [JM00].

An information extraction task is usually meant to be carried out on texts belonging to one language domain. That often includes dealing with language using specific vocabulary, expressions and sentence structure which are rather rare and not necessarily known to everybody. Considering, for instance, the MUC tasks, it is easily observed how different and specific the language domains of the test corpora can be. The methods used for creation of an information extraction system are considered domain independent. Nonetheless certain components of information extraction systems are specialised to perform on one certain domain and task.

[1]The first *Message Understanding Conference* (MUC-1) was held 1987, the last conference MUC-7 was held 1997. Nowadays various conferences in the field of computational linguistics and artificial intelligence deal with information extraction or its subtasks.

The most important and promising concepts of information extraction are shown in the flow chart in Figure 1. We will discuss information extraction components focusing on how they can be used for information extraction in other languages. Only those concepts and methods which I believe are useful and relevant in the context of other languages and multilingual information extraction are described.

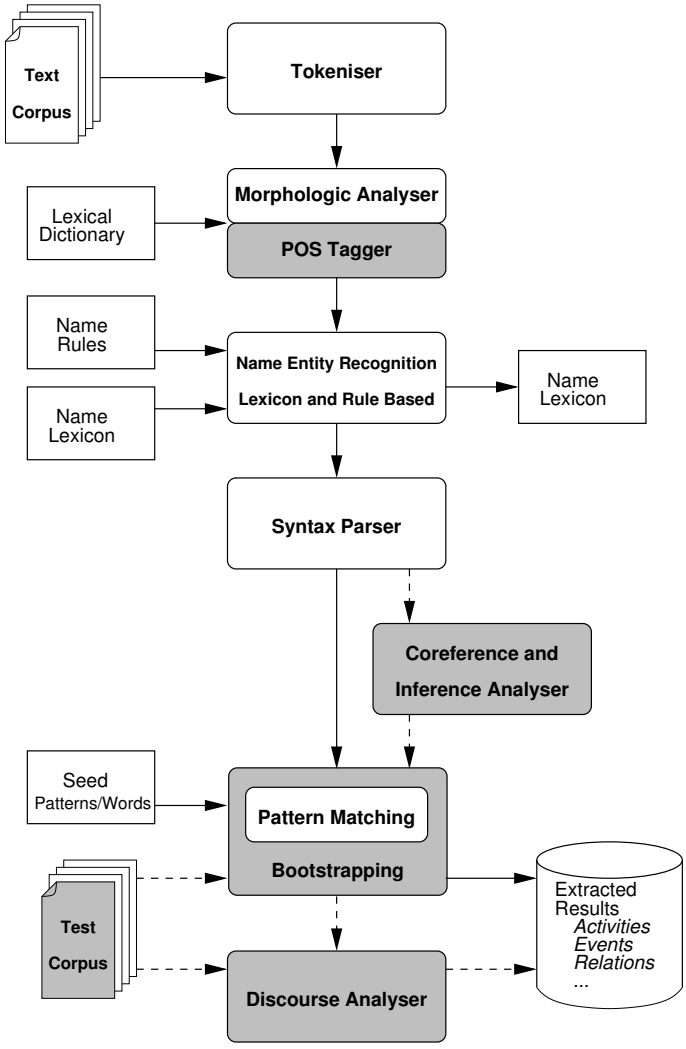

Figure 1: Outline of a generalised information extraction system.

Above, in Figure 1, each large rectangular box represents one component. The grey boxes represent components which are not used in all information extraction systems and are optional. The arrows show the work flow of information extraction, the dashed arrows depict optional paths. The smaller rectangles represent resources, that is lexica, text corpora, databases, lists of regular expressions and seed word lists.

At first the text corpus is *tokenised* into paragraphs, sentences and words[2]. After tokenisation we look up all words in a lexical dictionary and, if necessary, we analyse their morphology. Some information extraction systems apply in the same phase part-of-speech (POS) taggers, which add syntactical information about the words as tags to the text. Morphological analysis and POS tagging are strongly related and are often implemented as one component. They are described and analysed in detail in section 2.2.

The next component in line is *named entity recognition*. Named entities are definite noun phrases, naming, for instance, persons, locations and organisations. Named entities consist of one or more words and often represent the information which is to be extracted. There are several methods to recognise and extract named entities. At this point only rule and lexicon based named entity recognition is applied. We store the found named entities in a name lexicon for later use. More about named entity recognition can be found in section 2.3.

We discover further syntactical information comparable to the part-of-speech information throughout *syntax parsing* and tag it to the text. Syntax parsing does not depend on named entity recognition and we could apply it also before. Depending on the amount of syntax information required by the subsequent components, we either execute shallow syntax parsing and ignore complex syntax structures, or we use deep analysis methods to analyse the syntax fully. A more thorough introduction to parsing is given in section 2.4.

Newer information extraction systems apply explicit *analysis of coreferences* and *inferences* to produce better results. Some information extraction systems post-process the extracted results to discover relationships appearing in the discourse of the texts. Coreferences, inferences and *discourse analysis* are discussed in section 2.5.

Finding *extraction patterns* is the core task of information extraction systems. Information will be extracted using these patterns. Based on the linguistic analysis on the text during the previously applied components, extraction patterns match facts. These facts, or pieces of information, are later used to fill the slots of the result templates and are assembled to result data tuples. The basic concepts of extraction patterns are described in section 2.6.

Newer systems use various *bootstrapping* algorithms to improve the results of the pattern matching, or do unsupervised named entity recognition. Some systems re-

[2]Tokenising is a fairly simple task in languages using the Latin alphabet. In this paper I consider only such languages and hence will not discuss tokenising. However, for example Chinese, Japanese and Korean scripts are not that simple to tokenise and the recognition of word and sentence borders is an issue which must be addressed for information extraction in those languages.

quire a test corpus to evaluate the results of the pattern matching and bootstrapping process. Others use bootstrapping and evaluate the results by their appearance in the context and do not need a test corpus. The different bootstrapping algorithms are discussed in section 2.7.

After the single pieces of information have been extracted from text, they need to be merged together so that the information items belonging together are assembled together into one result data tuple. Merging is not further discussed in this paper.

2.1 Evaluation of Information Extraction

Information extraction systems may take quite diverging approaches in solving the problems at hand. A fair comparison of the results is often not directly possible. We need a comparison method in order to decide which approach works better under given circumstance. That will enable us to compare information extraction results. Additionally the performance of single components of information extraction needs to be evaluated.

Most often the used evaluation method is statistical evaluation. The results are compared against the correct solution to the problem. However, we have to keep in mind that sometimes there is not only one correct solution. In some cases even human experts disagree on which information exactly to extract. If we assume that a correct solution is available, it consists of a data tuple for each relevant text in the test corpus, containing the correct extraction set of information. An extraction result for a text is correct when the correct data tuple matches exactly the template the information extraction system filled[3]. A result is incorrect when the information extraction system extracts information that does not match, or that is irrelevant to the correct solution [AI99].

An exact match of one data tuple is called a *true positive (TP)*. A text document considered irrelevant in both the correct solution and the information extraction system output is called a *true negative (TN)*. If the information extraction system discards a text document which is not discarded in the correct solution, we call it a *false negative (FN)*. If the correct solution rejects a document but the information extraction system extracts a data tuple from it, it is a *false positive (FP)*.

[3]The definition of a pair of matching data tuples depends on the implementation of the system under evaluation. An exact string match, for example, will be too strict in the case of not normalised named entities (c.f. section 2.3).

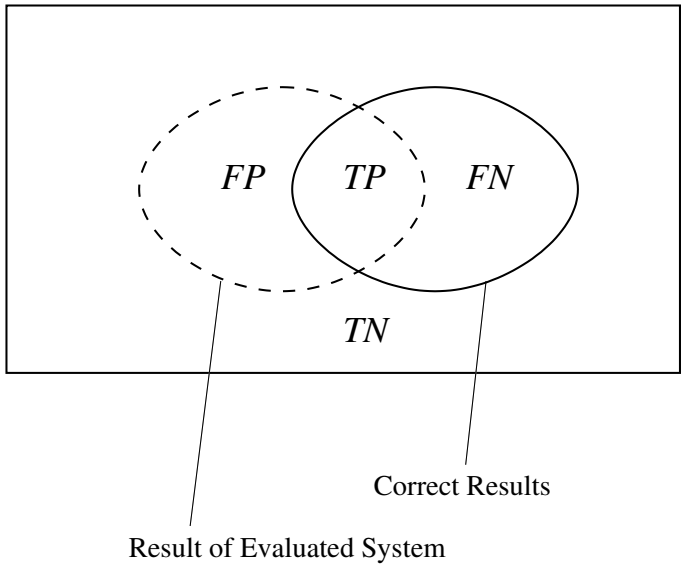

Figure 2: Precision and recall of the results.

Figure 2 depicts the TP, TN, FP and FN as sets. The result is correct when both sets, the evaluated results and the correct results, are the same and $FP = FN = \emptyset$. We need to compute more expressive values in order to get useful evaluation attributes from these statistical values. The *precision* π, given in equation (2.1), describes how many of the results the information extraction system generates are correct.

$$\pi = \frac{TP}{TP + FP} \tag{2.1}$$

The *recall* ρ, given in equation (2.2), describes how many of the correct results the information extraction system is able to find.

$$\rho = \frac{TP}{TP + FN} \tag{2.2}$$

Obviously precision and recall range in the interval $0 \leq \pi, \rho \leq 1$. Precision and recall each describe one quality of the information extraction system's performance. Systems have to maximise both. Accordingly, full evaluation of the system is only expressed by *both* values. In other words, if the precision is very high and the recall rather low, the result may be worse than an average value for both precision and recall. If we want to compare systems directly, we need one single value representing the quality of the information extraction system's performance.

One such value is the *F_β-measure*. It is calculated from precision π and recall ρ as given in equation (2.3) [Rij79].

$$F_\beta = \frac{(\beta^2 + 1)\pi\rho}{\beta^2\pi + \rho} \tag{2.3}$$

The π-weight parameter β, $0 \leq \beta \leq \infty$, is used to give the precision more or less weight than the recall. The F-measure used in many evaluations of information extraction systems, is the F_1-measure for $\beta = 1$.

$$F_{\beta=1} = \frac{2\pi\rho}{\pi + \rho} \tag{2.4}$$

Some newer information extraction systems do not need tagged training text corpora anymore, which means that test results are not available. Calculating precision, recall and F_1-measure values without a test result is not possible. Thus the comparison of such systems with others is not directly possible. Of cause such systems can be applied on a test corpus to get comparable results. That, however, overlooks the strength of these systems to perform under circumstances where no test corpus is available.

The evaluation techniques of precision, recall and F_1-measure are also employed for some of the information extraction subtasks. The results of POS taggers, named entity recognisers and syntax parsers, for example, can be measured using the same principles as we have seen above.

It would be desirable to have objective measures for the comparison of the performance of information extraction systems and their components. A fair comparison is only possible under identical testing conditions. Unfortunately these are not given for most of the concepts discussed in this paper, especially not for information extraction in other languages than English. Evaluation results are published with the use of precision, recall and F_1-measure in almost all publications on which this paper is based upon. The evaluation conditions are, however, not directly comparable in most cases. That is because different text corpora were used, preconditions were different and the evaluated language domains varied. Even though it is desirable to compare the methods described throughout this paper with each other, an evaluation based on the published evaluation values is not fair. Rather than comparing on an unfair basis, we will not make any comparison between results presented in different publications. For those interested, evaluation results are found in the cited papers.

2.2 Morphological Analysis and Part-of-Speech Tagging

Morphology is concerned with the internal structure of words, that is inflectional changes, derivations of words and compound words. In general there are two ways to deal with morphology: listing the inflections, derivations and possible compounds in a lexical dictionary, or listing only abstract descriptions (such as, infinitive, nominative) and describe the morphology with a set of inflection and derivation rules [Arn+94].

In comparison to other languages, the English language has a fairly simple morphological structure. There are, for instance, only few cases and verb inflection forms in English. Thus, morphology does not present a real problem in processing English texts. Some authors argue that listing all inflection forms in a lexicon provides less complex and fast solutions. Other researchers, on the other hand, have shown that rule based morphological analysers perform well also for English language, and utilise them for information extraction where lexicon based analysers are of less use [AI99, YGT00, Vou97, TJ97].

In order, for us to undertake morphological analysis, we need to look up each word in a lexical dictionary. We will have to do the same for part-of-speech analysis and, in a way, both are the counterparts of one and the same analysis. The part-of-speech feature is the functional part a single word plays in a sentence, for example, substantive, adjective, verb, adverb, conjunction, pronoun, preposition and determiner. In order to ascertain which part-of-speech a word represents, we will need to look it up in a lexicon. If the word is not in the dictionary, we can usually find the basic form of the word in the dictionary. We use morphological analysis in order to decide on the basic form of words. Once we have determined the part-of-speech of a word, the word receives a tag with its part-of-speech. This is in fact why the process is often referred to as part-of-speech tagging.

The example (S2.1) shows a simple sentence and its POS tags. '*VB*' marks the verb, '*DT*' the determiner and '*NN*' the singular noun of the sentence. In addition, finer grained information gained throughout morphological analysis, such as voice and tense of a verb[4], or number, case and gender of a substantive, may be added to the POS tag.

(S2.1)
VB DT NN
Book that flight.

[4]Note that verbs in the preterit tense and some other forms appear in combination with an auxiliary verb. POS taggers are not concerned with word groups and, hence, will not be able to determine the voice and tense of verb groups. This task belongs to syntactical analysis, which is covered within the following section. POS tagging is neither interested in noun phrases amounting to more than one word.

The major problem faced during POS tagging is the ambiguity of words, as the examples (S2.2) and (S2.3) depict: While *race* in the sentence (S2.2) is a verb, in sentence (S2.3) it is a noun. We need an indication from the context of the word *race* to know which POS tag is appropriate. Otherwise the part-of-speech of *race* is ambiguous. POS tags are dependent on the context in which the word appears and what it means.

(S2.2) *Secretariat is expected to* **race** *tomorrow.*

(S2.3) *People continue to enquire the reason for the* **race** *for outer space.*

[JM00]

We can resolve the ambiguity by examining the preceding word. Consider again the example above: If *race* is preceded by the determiner *the*, it is a noun, if preceded by *to*, it is a verb. Other ambiguities can be removed during syntactical analysis. Some ambiguities, however, cannot be ruled out with certainty. Nevertheless, in many cases one meaning is much more likely than the others [JM00].

Three different approaches to POS tagging have been used throughout the recent decades. The first approach is to build a rule based tagger. Rule based taggers incorporate a list of decision rules. In cases where there are ambiguities to resolve, a rule will match in the appropriate context and tag the correct part-of-speech.

The example (S2.4) below depicts such a rule that resolves the ambiguity of the word *race*. Rule based taggers typically include many such rules. The construction of these rules is a time consuming task and requires linguistic expertise. The POS tagger ENGTWOL is a more complex and well performing example of a rule based tagger [Vou97, JM00].

(S2.4) *If* ***race*** *is preceded by* ***to*** *then tag as* ***verb****.*

[JM00]

The second approach is to train stochastic taggers. Using a tagged training corpus, the tagger is learning the probability of each word, to have a certain tag. Using Hidden Markov models and the Viterbi algorithm, the tagger will find the sequence of POS tags for a whole sentence with the highest likelihood, based on the trained probabilities (see section 2.3) [JM00].

The Brill tagger represents the third approach. It uses transformation based learning. This method is a combination of probabilities and rules, where the rules are created using machine learning mechanisms. Hence, the creation of a Brill tagger requires a tagged training corpus [JM00, Bri95].

There are diverging opinions on how important POS tagging is for information extraction. The tagger used for the ExDisco information extraction system uses tags based on a grammar, heuristic rules and trained statistics. The tags contain quite rich details on each words part-of-speech [YGT00, Vou97, TJ97].

The opposing side claims that POS tagging may as well be omitted. It is clear, that in some cases, it is better to have no POS tag at all than an erroneous tag. It has been observed, that the cases in which information extraction would benefit most from POS tags are those where words contain rare senses. These are unfortunately also the five percent of cases where POS taggers fail to disambiguate [AI99].

2.3 Named Entity Recognition

A *named entity* is an individual existence such as a person, a place or an institution. Names are used to refer to named entities. Names of persons, places, institutions and so forth appear in almost every text. Names consist of definite noun phrases often including more than one word. Specific parts of the name have often specific meanings. In order to access the information in texts, it is important for us to identify those definite noun phrases that are names, as well as which named entity is described by the name. We can divide named entities into classes by their semantics. That is, for example, if the named entity is human, the name belongs to the semantic class *person*. Other classes are, for instance, locations, companies and institutions, product names, dates and numbers. Named entity recognition is to recognise named entities, to sort them into semantic classes and to tag them in the text. In information extraction, named entities are important because typically some or possibly even all slots of the result templates are to be filled with named entities. The quality of information extraction will inevitably be influenced by the quality of named entity recognition.

There are several difficulties in recognising and categorising named entities. Many names and their semantic classes can be listed in a semantic lexicon, so that we can look up them up for named entity recognition. The names of all countries, or common first and last names, for instance, can be easily listed. Yet, many names cannot be listed for several reasons. To list, for instance, more than just the largest cities of the world is tedious.

In other cases we cannot predict the whole number of all valid names. Product and company names, for example, are invented almost every day and naming conventions change. The example (S2.5), the name of a margarine product, depicts an extreme and unforeseeable name.

(S2.5) Product name of a margarine: *I can't believe it's not butter!* [AI99]

The ambiguity of words presents another problem. Sometimes a name will fit into several semantic classes or matches with several named entities. The name *Washington*, for example, will typically be used to describe a location. There are cases when that is not the case as example (S2.6) shows, where *Washington* is a person name. The named entity recogniser needs to decide whether the ambiguous name belongs to another semantic class.

(S2.6) *Consuela Washington, a longtime staffer and an expert in securities laws, is a leading candidate to be chairwoman of the Securities and Exchange Commission in the Clinton administration.* [Poi00]

Quite often one named entity has several representations as, for instance, persons names. The names *'Mr Smith'*, *'John Smith'* and *'Smith'* are some possible references to the same person. Yet, the same text could possibly mention another person *'Will Smith'*, which would make *'Smith'* ambiguous. Beside synonyms, other name references also appear in text. Coreferences such as *'the company'*, *'he', 'they',* and *'we'* refer to named entities mentioned earlier in the discourse. Much information is spread across sentences through such coreferences. Leaving coreferences unresolved means missing relevant information. Coreferences are dealt with separately in section 2.5. References expressing real world knowledge are rather difficult to deal with. The named entity in example (S2.7), for instance, refers nowadays to *'Bill Gates'*, and is a typical case of applied real world knowledge [AHG98, AI99].

(S2.7) Referring nowadays to Bill Gates: *The richest man of the world.*

Throughout the last decade, research on named entity recognition concentrated on several specific problems. As a result some methods for certain general and domain specific semantic classes are widely accepted. Besides the person names mentioned above, location names and company names have received special attention. If the text corpus consists of a certain domain language, such as medical reports, domain specific vocabulary and semantic classes must be taken into consideration [AI99].

Four different approaches are widely used for named entity recognition in information extraction. Semantic lexica are used for named entities which can be listed in reasonable time, or when name lists exist already. Handcrafted recognisers consist of a set of rules to recognise named entities. Stochastic methods are used to train recognisers with tagged training text. Bootstrapping applies repeatedly extraction patterns to recognise named entities and to sort them into their semantical classes. These methods perform differently on dissimilar named entities and typically two or more of them are applied in combination [AI99].

A semantic lexicon contains a list of names, their corresponding named entity (a normed form of the name) and the corresponding semantic classes. Named entity recognition uses lexicon based analysis, if such a semantic lexicon is available. Lexical analysis appears to perform better with rather small domain specific lexica than with large general lexica [AI99].

Other named entities are better identified using regular expressions to match names following certain naming conventions. In many languages, for example, names start with capitalised initials[5], which can be used to identify them in texts. Naming rules must be implemented and tested by a human expert. Usually regular expressions are used in combination with domain specific semantic lexica. They are also used to build such lexica in the first place. The information extraction system WHISK for semistructured and structured texts employs these techniques in many contexts [Sod99].

Implementation of name recognition rules by hand is quite time consuming and requires some language expertise. Instead we can use statistical methods and machine learning algorithms to train named entity recognisers automatically. Machine learning is based on hidden Markov models. A hidden Markov model is an unknown finite state automaton. We assume that a finite state automaton exists which consumes the text and reaches a final state exactly when a proper name and its surrounding context have been consumed.

[5]One exception is the German language, where all substantives begin with a capital letter.

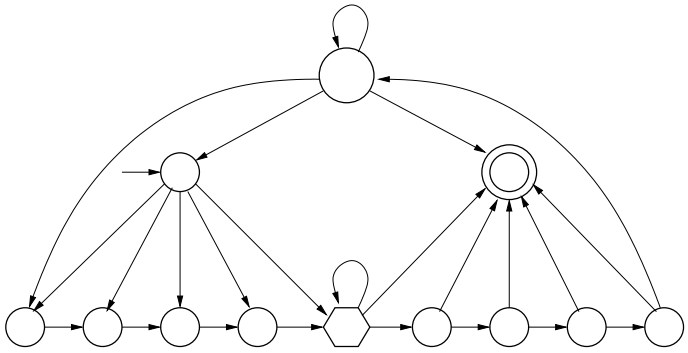

Figure 3: A finite state automaton topology for hidden Markov models.

Figure 3 above shows the layout of such a finite state automaton. The circles are states of the automaton each representing one word. The hexagonal shaped state matches the proper name. The arrows depict possible transitions from one state to another, that is the order in which the words may appear. The automaton matches every sentence that has a valid path through the automaton from the beginning state, marked by an arrow with no previous state, to the final state, denoted by two concentric circles. Each finite state automaton is equivalent to a regular expression. Constructing an automaton for a certain proper name is similar to constructing a regular expression as rule to match the name. We use hidden Markov models to estimate such automatons when they are not available or are too laborious to construct. The arrows of a hidden Markov model depict probable transitions from state to state. During a training phase, a probability is learnt for every arrow. We learn the probabilities by counting the occurrences of word sequences in a training text. Using the Viterbi algorithm, we will get the likelihood of the transitions from one state to the next for each state in the automaton. When we apply hidden Markov models to match proper names, several automatons may match the same sentence. We choose the one with the highest probability along the whole matching path. A hidden Markov model can have more than one (hexagonal) target state to reflect possible variations of the proper name. We need a training corpus with tagged named entities to construct a hidden Markov model [FM99, JM00].

The following observation points out quite clearly the limitations of hidden Markov models. Two separate groups of scientists noticed quite independently that roughly every doubling of training words increases the F-measure by 0.015. This rule has been verified for a maximum of 1.2 million training words, at the time the maximum training corpus available. In order to achieve a result with F-measure ≥ 0.9, the time needed for tagging enough words of training data has been estimated to be about 800 hours, or 20 man weeks. The assumption is that a rule set which obtains the same F-measure results can be handcrafted in far less time. This example makes it clear that hidden Markov models are only helpful if enough training data is already available [AI99].

2.4 Syntax Analysis

In contrast to POS tagging, *syntax analysis*, also called syntax parsing, looks beyond the scope of single words. During syntax analysis we attempt to identify syntactical parts of a sentence (verb group, noun group and prepositional phrases) and their functions (subject, direct and indirect object, modifiers and determiners). Simple sentences, consisting, for instance, of a main clause only, can be parsed using a finite state grammar. Simple finite state grammars are often not sufficient to parse more complex sentences, consisting of one or more subordinate clauses in addition to the main clause, or containing syntax structures, such as prepositional phrases, adverbial phrases, conjunction, personal and relative pronouns and genitives in noun phrases. Using finite state grammars in such cases may result in errors. Instead, those cases are handled by either heuristic hand coded rules or statistically founded methods which have to be trained with training text corpora. The important decision to be made for syntax analysis is basically the same as for named entity recognition and POS tagging. We have to decide what kind of parsing is to be employed: more robust shallow techniques, or deep complex syntax analysis [JM00].

The subsequent methods used in varying information extraction systems are different especially from the point on, when syntax parsing is applied. The type of syntax analyser employed depends on how the results are used later on during coreference resolution, pattern matching and discourse analysis. Shallow systems exclude certain syntactical constructs from their analysis based on how relevant they might be for information extraction tasks. As a result important information might be ignored. The SRI international FASTUS system favours a rather shallow and robust approach to syntax parsing. This decision is based on the claim that experience so far has proved this approach to be the superior one [AI99].

Nonetheless is might be necessary to analyse some of these structures in certain languages domains. A very promising syntax parsing tool has been created based on the formalism of functional dependency grammars (FDG). It is used in the ExDisco information extraction system. The FDG parser uses heuristic rules to create grammar dependency trees and performs a thorough, if not complete syntax analysis. The parser has specific heuristics dealing with ambiguity. The heuristic rules used for the FDG parser are not strictly language specific and the FDG parser has been adjusted to a variety of languages [TJ97, Yan+00, YGT00].

2.5 Coreferences and Discourse Analysis

Free text contains a rich variety of references and relationships which are beyond the scope of simple syntactical analysis and often represent some kind of semantics. Named entity recognisers, for instance, when unaware of references, may fail to find named entities. The early information extraction system Wrap-Up attempted to extract relationships from text by performing *discourse analysis*. The new information extraction subtask *coreference analysis* was introduced in MUC-6[6]. One information extraction system also carries out *inference analysis* [SL94, SL95, Gri97].

Coreference analysis deals with anaphoric references which often appear in the form of a pronoun or a definite noun. Anaphora typically appear as a pronoun or noun in a preceding or subsequent clause of the noun they refer to.

Examples (S2.8) and (S2.9) show a typical anaphoric coreference. The anaphora *her* in (S2.9) refers to the noun phrase *This woman* in (S2.8).

(S2.8) **This woman** *came here every day.*

(S2.9) *If I would have only known* **her** *name.*

Sometimes anaphoric references stretch out over several sentences. Often anaphora cannot be resolved because synonymous named entities have not been recognised as such during named entity recognition. This is especially the case if pragmatic or real world knowledge is required to decrypt the reference (see example (S2.7) on page 12). Coreferences can appear in many contexts and locating coreferences is quite important for information extraction. Facts are often spread out over several sentences. Ignoring coreference would mean omitting facts in many cases [Yul96].

[6]Coreference analysis was not a new task at that point but had been performed implicitly in earlier information extraction systems [AI99].

There are two common methods of coreference analysis: knowledge engineering and automatically trained systems. In the knowledge engineering approach we consider every noun phrase to be a potential coreferent and prune them so that only coreferences remain.

The following steps outline this pruning:

1. Find each candidate coreference and analyse its attributes.
2. Prune the scope for each candidate.
3. Rule out candidate coreferences by doing sortal and semantical checks.
4. Sort the remaining candidate coreferences by dynamic semantical preference.

The first step is to assign a set of attributes to each noun phrase. These attributes are sort (location, company, person), number (singular or plural), gender and syntactic features. The next step is to set the probable word or sentence scope for each candidate. We carry out a sortal check by comparing the attributes of the referent and antecedent of every candidate coreference. Then we sort all remaining candidates by a dynamic semantical preference to rule out competing ambiguous coreferences. Finally, we use the best candidate referent antecedent pairs to resolve the coreference [AI99].

Automatically trained systems are based on hand tagged training texts and use, for example, decision trees in deciding what is a coreference and to which noun phrase it is referring [HGA97].

It has been observed that coreference resolution fails to resolve coreferences in a few complex cases. Unfortunately, complex cases are those which would be especially useful for ambiguity resolution in other information extraction tasks [AI99].

Inference is information which can be derived from coreferences. Consider examples (S2.10) and (S2.11): In example (S2.10) and (S2.11) the inference is that *Harry* has become *president*. In order to understand this information, we need at first resolve the corefence that *He* is *Sam*. The fact that *Sam* was a *president* and that the verb *succeeded* implies that the office is transfered to *Harry*, is to be resolved. Most information extraction systems ignore inferences. The NYU Proteus system extracts information from inferences using hard-coded rules [Gri97].

(S2.10) *Sam was president.*

(S2.11) *He was succeeded by Harry.*

Discourse analysis aims to expose all kinds of information given throughout the discourse of a text. Such information can be, for instance, family relationships, ownership or even production chains. The Wrap-Up and Crystal information extraction systems use ID3 decision trees for discourse analysis on the MUC-5 task (see MUC-5 on page 3). Most information extraction systems do not perform discourse analysis [SL94, SL95].

2.6 Extraction Patterns

The resulting output of information extraction consists of single data items filled into the slots of data tuple templates. The data tuples populate the result database, one tuple for each relevant document of the input text corpus. The data items are pieces of information which have to be located in the text. *Extraction patterns* are used for this task.

An extraction pattern is a text pattern which matches a certain token and its surrounding context. While the surrounding context is constant, the token is variable. The context might consist of words or linguistic tags, such as POS or syntax tags. One single extraction pattern matches only a certain type of information so that it will always produce the same type of item in the result template. Consider the examples (S2.13) and (S2.14). The extraction pattern in example (S2.13) extracts bomb targets and the pattern in example (S2.14) extracts perpetrators.

The AutoSlog system introduced extraction patterns that extract noun phrases within the context of a sentence. Using a few hand picked noun phrases as *seed words* AutoSlog finds all extraction patterns in a text corpus that occur as context for the seed words. It produces, for example, with the seed words *'World Trade Center'* and *terrorists* from the sentence (S2.12) the extraction patterns (S2.13) and (S2.14). We use these extraction patterns in turn to extract bomb targets and perpetrators from text.

(S2.12) *World Trade Center was bombed by terrorists.*

(S2.13) `<x>` *was bombed*

(S2.14) *bombed by* `<y>`

AutoSlog uses 13 distinct linguistic extraction patterns in order to find such patterns. An example of AutoSlogs linguistic extraction patterns is the pattern (S2.16). These linguistic patterns are applied to every sentence where the seed words appear. In example (S2.15) the seed word *'World Trade Center'* occurs and the linguistic pattern (S2.16) matches. The match of the linguistic pattern is translated into the extraction pattern (S2.13) [Ril93].

(S2.15) `<`*World Trade Center* `subject> <`*was bombed* `passive verb>`

(S2.16) `<subject> passive verb`

Not all such patterns created will be of use because they might be too general, too specific or contain too many ambiguities. The AutoSlog-TS system extends AutoSlog with a sentence analyser doing the following: Every found extraction pattern is applied on a test text corpus. The text corpus consists of documents classified as relevant and irrelevant for the information extraction task. The sentence analyser gathers statistics about the number of matches for every pattern found in the relevant texts.

Then AutoSlog-TS ranks the extraction patterns according to their $RlogF$ metric shown in equation (2.5).

$$RlogF(pattern) = \frac{F}{N} \times log_2(F) \tag{2.5}$$

F is the number of unique lexicon entries matched by *pattern* and N is the total number of unique noun phrases matched by *pattern*.

It is assumed that the extraction patterns generated by AutoSlog and AutoSlog-TS will be reviewed by a human expert before they are finally used in building an information extraction system. The bootstrapping algorithm described in the following section represents a solution making such reviews unnecessary [Ril93, Ril96].

2.7 Bootstrapping

The extraction patterns created by AutoSlog and AutoSlog-TS (see previous section) are not necessarily of the required quality. Some patterns will be too general and extract wrong information, others are too specific and extract only the exact phrase they are created from. The review of the ranked patterns delivered by AutoSlog-TS requires work of a human expert. This is time-consuming. It is also unlikely that the seed words appear in all contexts where wanted information appears, and, hence, potentially good extraction patterns will be overlooked. All these problems are targeted by bootstrapping algorithms that automatically evaluate extraction patterns, extend the search of extraction patterns beyond the scope of the seed words, and bootstrap the result to a list of quality extractions.

The *RlogF* ranking performed by AutoSlog-TS has one weakness. It does not recognise false positive matches. With the help of a bootstrapping mechanism, we are able to automate the selection of the most appropriate patterns and improve the scoring of extraction patterns. Using seed words from certain semantic classes (person names, locations, etc.), the bootstrapping creates a *semantic lexicon* from noun phrases found in similar contexts as the seed words. The multi-level bootstrapping algorithm ranks patterns and their extracts separately to prevent bad noun phrases from being placed into the semantic lexicon. The algorithm consists of an inner loop, the mutual bootstrapping, and an outer loop, the meta bootstrapping.

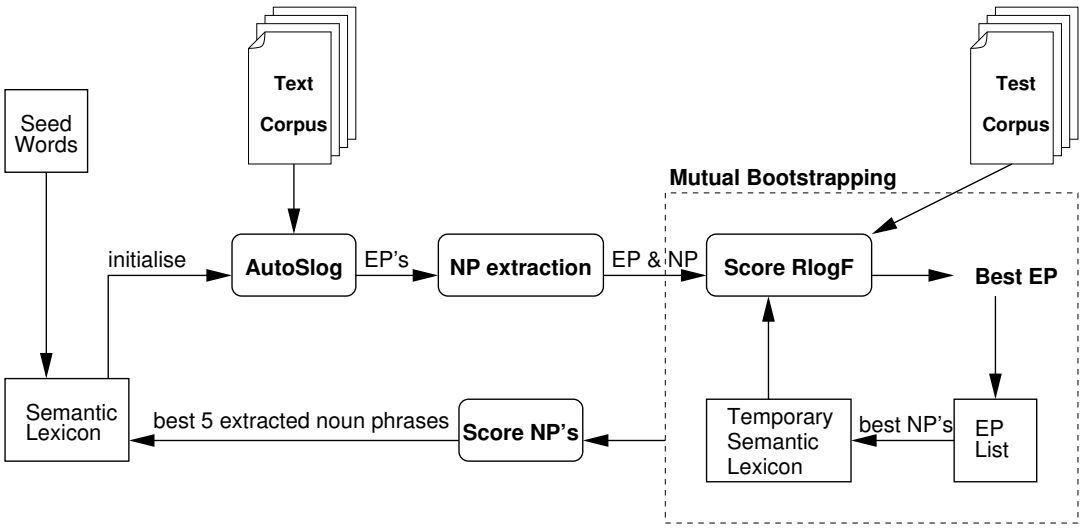

Figure 4: Multilevel bootstrapping.

Figure 4 depicts how AutoSlog and two levels of bootstrapping are combined for noun phrase extraction. We employ AutoSlog with a list of seed words to create extraction patterns from a text corpus. We apply all those candidate extraction patterns to

the text. We save the list of extraction patterns and their corresponding extracted noun phrases. The mutual bootstrapping loop iterates on this list. We score the candidate patterns during each loop using the *RlogF* metric (see equation (2.5), on page 19). We then automatically choose the best candidate extraction pattern, remove it from the candidate pattern list, and add it to a category pattern list. We add the corresponding extracted noun phrases to a temporary semantic lexicon. The bootstrapping loop ends when all candidate extraction patterns have been used, no new words are found, or the *RlogF* score falls below a user defined threshold. We score the noun phrases of the temporary semantic lexicon based on how many category extraction patterns have extracted them and how highly those patterns scored. The equation (2.6) is used as score metric.

$$score(NP) = \sum_{k \in N} 1 + (0.1 \times RlogF(pattern_k)) \tag{2.6}$$

N is the set of extraction patterns in the extraction pattern list which extract noun phrase NP.

During every iteration of the outer loop we add the best five noun phrases to the semantic lexicon. We then initialise AutoSlog with a seed word list that has been extended by the five new phrases. The multilevel bootstrapping algorithm ends when the meta bootstrapping has iterated a certain number of times, no new words are found, or the noun phrase score falls below a certain threshold. We apply the multilevel bootstrapping on seed words, one semantical class at a time. The algorithm produces a semantical lexicon containing noun phrases falling all into the same semantical class as the seed words. Semantical lexica are used, for instance, during named entity recognition [RJ99].

To perform multi-level bootstrapping, we still need a test corpus, and this bootstrapping algorithm extracts only categories of noun phrases. The ExDisco system uses a bootstrapping algorithm which needs only seed words but no test text corpus. A different style of syntax tagging makes it possible to categorise also verb groups.

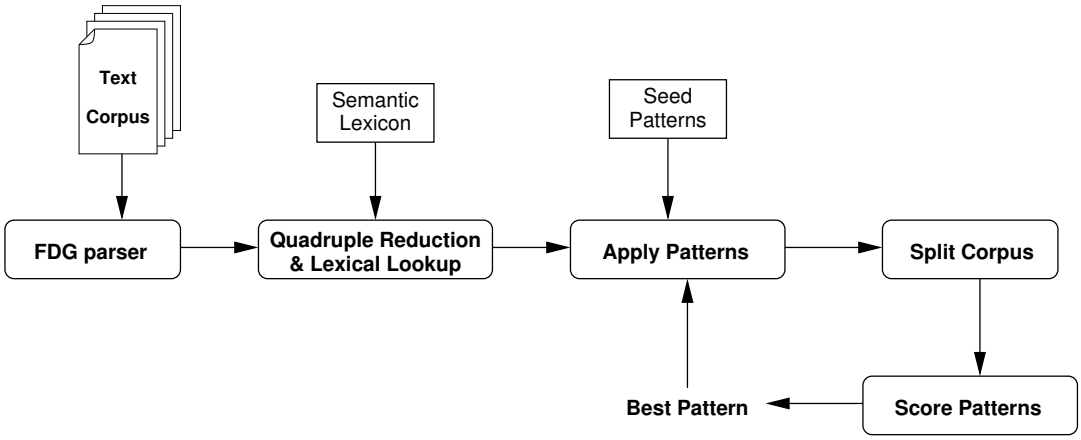

Figure 5: The ExDisco bootstrapping.

Figure 5 outlines the ExDisco system. The whole text corpus is firstly parsed with a FDG parser (see section 2.4) which reduces all clauses to a set of quadruples. The quadruples are further reduced to a set of pairs, for example subject-object or verb-object, so that one or more pairs are produced from each quadruple. We generalise these patterns even further by replacing each lexical item, that is a recognised member of a semantical class, by its class name.

During the bootstrapping the following steps are iterated:

1. Apply all seed patterns on the whole text corpus.
2. Split the text corpus into two categories, so that one category contains all relevant texts in which one or more seed patterns scored and the other category contains all the other texts.
3. Score all the patterns gained from the text corpus based on their density of distribution in relevant documents in comparison to their density of distribution in all texts.
4. Use the highest scoring patterns to generate concept classes by merging those pairs which appear in the correlated text.

The pattern (S2.17), for instance, contains a class of verbs which occur with the relevant subject-object pair `company-person`. These verbs have been merged into the same pattern. During each iteration we add one such pattern to the seed patterns. ExDisco stops when no new patterns are discovered [YGT00, Yan+00].

(S2.17) `company` {*hire/fire/expel*} `person`

3 Multilingual Information Extraction

Information Extraction has for many years meant extracting templates from text in the English language. Almost all research effort was spent in improving the performance of English information extraction systems. When it became evident that information extraction systems can perform considerably well, the focus of research began to shift toward information extraction in other languages.

By shifting the focus into other languages than English, the problem of information extraction, as described in the previous chapter, gets a new dimension. The first objective is to create information extraction systems that perform the same tasks but on a different language. On second glance, there are many more challenges in a multilingual context than transferring monolingual information extraction systems into a new language. Users might not know the language of the texts, or corpora may consist of texts written in more than one language. Knowledge about a language, linguistic tools and text corpora in a language may not be available.

Adaptation of information extraction systems to new languages means that we need to adapt the components of information extraction to new languages. Some of these components are easily adaptable using one of the methods used to build these components for English language text. Others are already implemented and available. Some are, however, not available and adapting them by implementing the same ideas used in English will fail due to greater complexity of linguistic features in many languages. In many cases the resources available in English that initiated the invention of the methods in the first place are not available in other languages, and the effort it would take to create these resources seems excessive.

The additional dimension of multilinguality, nevertheless, offers some opportunities that were not available when English information extraction systems were created. Machine translation is the key technology, which, despite its still poor quality, makes transfer of linguistic tools and knowledge across languages possible. Remarkable ideas are still being published enabling linguistic tasks in languages that seemed impossible not too long ago.

In multilingual environments, such as the Internet, where texts appear in numerous languages, monolingual information extraction is a restricted tool. A multilingual framework for linguistic tools that would deal with texts in various given languages is needed. Information extraction results should extend across language borders. Such multilingual framework will be useful for information extraction, as well as information extraction components and other linguistic applications.

Within this chapter we will look at the difference between monolinguality and multilinguality and we will study how information extraction can be adapted to new languages. We will additionally analyse multilingual concepts for information extraction and its components. We will get introduced to the concepts of text alignment, cross-language projection, language recognition and machine translation and we discuss their benefits to multilingual information extraction. We will analyse the adaptability of POS taggers, named entity recognisers, coreference and discourse analysers to new languages using classical and new approaches. Bootstrapping and extraction patterns are language independent concepts and we will not explicitly discuss them in this chapter. Nor will we discuss syntax analysers. For now we assume that syntax analysis is a language specific tasks and needs to be handcrafted separately for each and every language.

3.1 Monolingual, Bilingual and Multilingual Information Extraction

A monolingual information extraction system performs on texts written in one language. In the last chapter we discussed English language information extraction − an example of monolingual information extraction. We assumed that everything, the text on which information extraction is performed, the subtasks, and the generated output were in English. Thus, we define monolingual information extraction as follows:

Definition 1 *A monolingual information extraction system performs information extraction on text input written in one single language and produces results in that same language.*

We will see various concepts using tools to transfer linguistic knowledge from one language into another in the following sections. Unlike English monolingual information extraction, the new approaches process during training or application of information extraction more than one language. They perform, however, still monolingual information extraction in respect to Definition 1 as long as the input text and the results are in the same language. We could certainly argue that those architectures are designed to perform some tasks in more than one language and are, thus, not monolingual. However, when we compare them with the capabilities of multilingual information extraction systems, we will discover that they are monolingual although they utilise multilingual linguistic tools.

The difference between mono-, bi- and multilingual information extraction is the number of languages appearing in input and output of the system. Monolingual information extraction systems process text in one language, bilingual information extraction systems handle two languages and multilingual systems two or more languages.

Confusingly the terms 'bilingual', 'multilingual' and 'language independent' are being used with various meanings in recent publications. The question, for example, if two monolingual systems together constitute a bilingual information extraction system will be answered differently depending on who is being asked. Often the impression remains that people use the terms 'multilingual' and 'language independent' to make their objects of research sound brighter. Some information extraction systems use, for example, monolingual information extraction tools for a target language L_2 and bilingual tools for a source language L_1 and L_2. Yet the result is a system performing monolingual information extraction in L_2 despite bilinguality being involved throughout the task (see Figure 6 on page 30). The question remains when each term is to be used. We define bi- and multilingual information extraction as follows:

Definition 2 *A bilingual or multilingual information extraction system processes input text in two or two or more languages.*

Following on the same line of thought, we can define language independent information extraction as follows:

Definition 3 *A language independent information extraction system processes input text in any language.*

An information extraction system taking bi- or multilingual input does not necessarily process output in all, or even any, of the input languages. A multilingual information extraction system, for instance, may produce all information extraction results in English even though it processes texts in a number of languages. Some information extraction systems keep their results in a language independent representation, an interlingua (see section 3.3).

The question on what the terms mono-, bi- and multilinguality and language independence describe is not only relevant for information extraction as a whole but also for its components and linguistic tools in general. Let us consider the components of information extraction systems. All of these components deal with input text or some processed parts of it. Inevitably every component of information extraction has to be either specialised for text in one language, or a few specified languages, or it has to be able to perform its task no matter what language is involved. Each such

component is either dependent on a certain input language or not. As mentioned earlier, some components as, for instance, bootstrapping, are language independent because they can be applied on any tagged and tokenised texts whatever the language. The other language dependent components, nonetheless, need to be adapted to each and every new language. Let us take a closer look at these components of bi- and multilingual information extraction.

POS taggers, named entity recognisers, syntax parsers and coreference resolvers are examples of linguistic tools. We define monolingual linguistic tools as follows:

Definition 4 *A monolingual linguistic tool processes input text in one language and produces output text in the same language.*

Again, the definition above leaves the problem of whether the processing, or creation of the tools involves multilingual methods unresolved. The linguistic tool is treated as a black box, which from outside is monolingual. Analogously, we describe bi-, multilingual and language independent linguistic tools with the following definitions:

Definition 5 *A bilingual or multilingual linguistic tool processes input text in two or two or more languages and produces output text in those same languages.*

Definition 6 *A language independent linguistic tool processes input text in any language and produces output text in a language independent representation or in the language of input texts.*

Language independent tools, as they are defined above, are not restricted to tasks, such as bootstrapping, where the language is irrelevant. In theory, a tool performing a language dependent task but being able to handle all languages, is also a language independent linguistic tool.

A text corpus is monolingual if it contains texts written in one language only. A bilingual or multilingual text corpus contains texts written in one of two or several languages. In some cases text contains a mixture of sentences in different languages.

Language recognition is able to recognise language for distinct sentences (see section 3.2). Yet, such texts are hardly of use in information extraction. The detection of information spread out over several sentences written in different languages seems very unlikely unless a specific tool for this task is developed.

Multilinguality increases the use cases of information extraction systems and linguistic tools. Users and creators of the system may or may not know the languages involved. The one linguistic use case for monolingual information extraction is to extract information from texts in the same language. Typically, training the system involves only that one single language.

We find the following use cases for a bilingual system in languages L_1 and L_2.

i. The user of the system knows only L_1, only L_2, or both L_1 and L_2.

ii. The system processes texts in L_1 only, L_2 only, or both L_1 and L_2.

iii. In order to build the system, linguistic expertise in L_1 only, L_2 only, or both L_1 and L_2 is required.

iv. Linguistic resources for training the system, such as training or test corpora, are available in L_1 only, L_2 only, or both L_1 and L_2.

Such use cases can lead to situations where we must translate the extraction results into another language, so that the user is able to read them. Another observation is that language expertise, or resources for some languages, may not be available. Obviously, these problems grow exponentially when more than two languages are involved in a multilingual system.

An architecture for multilingual information extraction can be developed under diverging priorities. A system might be designed, for instance, to maximise the result quality, to minimise building efforts, to be open for extensions of new domains and languages, or fit other needs. If a system is, for example, meant to be used in a certain bilingual domain language, there is probably no point in trying to keep it domain or language extensible. Generally systems should be as open to extensions as possible. Nowadays, the most restricting factor in deciding what architecture to choose is the availability of linguistic resources. In the future, the number of available resources in many languages will change. At some point the above restriction may become insignificant in comparison to other factors.

At the moment research focuses on multilingual systems for small specific domains, trying to keep the architecture open for new languages. New language independent representations, such as template interlingua, have been introduced. The problem for multilingual systems dealing with many languages is especially the question in which language, or interlingua to store the extracted results. If users have different language backgrounds we may need to translate the results into any language required [Azz+97, Bol+97, Kam97, Din98, Gri+99].

3.2 Language Recognition

The first task for an information extraction system in a multilingual context will be to recognise the language of the documents in a text corpus. In some situations a text corpus consists of texts in a mixture of languages. That, for example, is the case with documents retrieved through a web search, or document collections poorly maintained in the context of the European Union. In such cases the document languages must be identified prior to information extraction.

We can use bi-gram frequency matrices in order to recognise a particular language. This method is based on the observation that every language has its own orthographic characteristics that appear more frequently. A bi-gram is a sequence of two characters. We consider punctuations, for example full stop, comma and exclamation mark, as word separators, and summarise them as one symbol representing space. If a text is written in several languages, for instance, one paragraph in one language, or one sentence in one language, we employ paragraph separators or sentence separators. A full bi-gram frequency table contains the frequency of every possible bi-gram for one language. Hence, a language recogniser needs a frequency table for every language it is to recognise. In order to generate such bi-gram frequency tables, we need to select representative texts from every included language. If, for instance, a text contains an untypically high number of loan words or proper names from another language, the frequencies will not represent that language properly.

Each bi-gram has a distinct frequency in every language[7]. In other words, the frequency for a bi-gram in one language shows the probability of it to be of that language. Let L be the set of languages implemented in the language recogniser.

Further, let ab be a bi-gram and table T_l the frequency table for bi-grams in language l. The probability $P_l(ab)$ for ab to be of language $l \in L$ is:

$$P_l(ab) = \frac{T_l[ab]}{\sum_{k \in L} T_k[ab]} \tag{3.1}$$

It has been observed that often particular bi-grams appear frequently in several languages. Dutch and German or Spanish and Portuguese, for example, have often rather similar bi-gram frequencies.

[7] With the exception of a few strongly related or otherwise orthographically very similar languages.

The probability of a word w is estimated to be of language l as follows:

$$P_l(w) = \frac{P_l(\_w[1]) + P_l(w[|w|]\_) + \sum_{i=1}^{|w|-1} P_l(w[i]w[i+1])}{|w|+1} \tag{3.2}$$

In equation (3.2) $|w|$ is the length of w in characters, $w[i]$ represents the i-th character of w and '_' represents the word separator.

Proper names and loan words from other languages impact the performance of language recognition. In order to decrease that negative impact, we analyse the context heuristics of words in a whole sentence or paragraph. The probability for a word of being of a certain language depends also on the language estimation of its neighbouring words. Close neighbours will have a stronger impact on the language estimation and we will have to consider only words within a specified window. We use a weight of $1/(1+x)$ for the influence of neighbour probability, where x is the distance in words. At first, we cumulate the probability of the i-th word w_i in the sentence or paragraph and the weighted probabilities of the s closest neighbour words of W_i, to be of language l, as follows:

$$Q_l^s(w_i) = P_l(w_i) + \sum_{j=1}^{\lfloor s/2 \rfloor} (P_l(w_{i-j}) + P_l(w_{i+j})) \times \frac{1}{1+j} \tag{3.3}$$

We still need to normalise this cumulated values in order to obtain the heuristic probability of the i-th word in the sentence or paragraph w_i for language l and the window size s, $P_l^s(w_i)$, as shown in the following equation:

$$P_l^s(w_i) = \frac{Q_l^s(w_i)}{\sum_{k \in L} Q_k^s(w_i)} \tag{3.4}$$

In order to finally decide on the language a paragraph or sentence is written in, we merge the heuristic probabilities of all words. For paragraph $p = w_1, w_2, \ldots, w_n$ the probability $P_l(p)$ is calculated as follows:

$$P_l(p) = \frac{\sum_{i=1,\ldots,n} P_l^s(w_i)}{n} \tag{3.5}$$

We pick the language with the highest probability $P_l(p)$ as our result.

The method for language recognition as described above has not been systematically tested. Thus, possible weaknesses exist, such as where texts contain an exceptionally high number of loan words and foreign expressions [Hag99].

3.3 Machine Translation

Machine translation is one of the traditional challenges of natural language processing. It has various linguistic problems in common with information extraction. Machine translation can be used in situations where other methods for information extraction fail due to lack of linguistic resources. If we want to use machine translation for information extraction, we need to understand its strengths and weaknesses. Automated translation of free text still produces many errors or leaves ambiguous phrases untranslated. Machine translation is nowadays only used as a tool for human translators, the final translation is always a result reviewed by a human. Typical pitfalls of machine translation are ambiguities, pragmatics, discourse and idiomatic phrases. Around ten years ago machine translation was predicted to remain a not fully solved problem for years to come. That has turned out to be the case so far. Generally, machine translation will not be able to translate randomly given free text in a satisfactory manner for use in information extraction [Arn+94].

Machine translation resources, as information extraction resources, are not evenly distributed between languages. Machine translation systems for the most widespread and important languages, English, Spanish, French and German, are further developed in comparison to less spoken and less important languages. In addition, linguistic problems within certain languages limit the availability of machine translation. Machine translation from English to Russian, for instance, is considered fairly simple in comparison to translation from Russian to English due to the complexity of Russian grammar.

Let us first take a look at the most obvious architecture in order to understand how machine translation is useful for information extraction. Let us assume that a text corpus in language L_1, an information extraction system in language L_2 and a machine translation system translating from L_1 to L_2 are available.

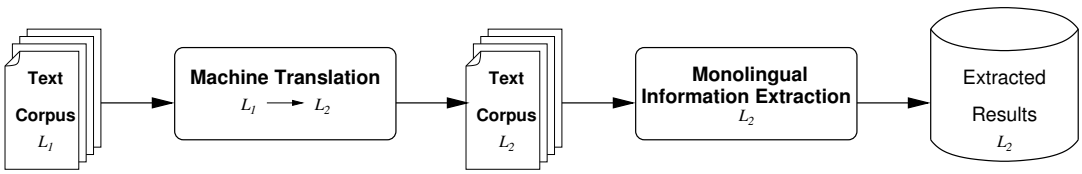

Figure 6: A simple IE architecture using machine translation.

As depicted in Figure 6, the text corpus T_{L_1} is first translated into language L_2. Then information extraction is performed on the translated text. The main weakness of machine translation will come to effect when we apply machine translation on T_{L_1} first. The translation errors will have a negative impact in the performance of

information extraction. Machine translation does not perform well on free text. It is worth observing how machine translation is affected by language domains especially considering that information extraction is applied on text of a certain language domain. A general machine translation system performs well on some domains, yet on others its performance can be worse than that on free text. Nevertheless it is possible to adapt machine translation to language domains [Arn+94].

Information extraction does not demand full text translation when we use another architecture. It is sufficient to translate the information extraction results only. The data items of information extraction results consist of rather simple language, such as noun phrases or verbs. Several researchers report satisfactory results using off-the-shelf products for machine translation of information extraction results [RSY02].

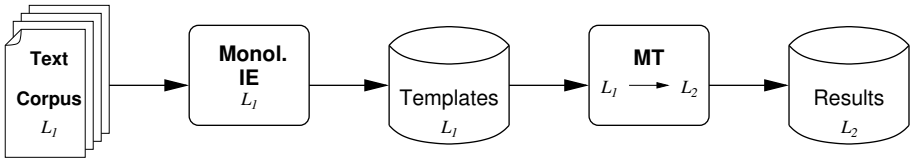

Figure 7: Machine translation of information extraction results.

Figure 7 depicts this combination of machine translation and information extraction. Firstly, full monolingual information extraction is performed on the text corpus T_{L_1}. The machine translation is thus applied on the information extraction results. If we only consider the performance of machine translation, this architecture is superior. That is because in comparison to the one we discussed previously, machine translation performs better on information extraction results than on full text. Information extraction results contain less complex language structures than full text. Unfortunately, that leaves any possible language resource shortage problem of language L_1 unresolved. A full monolingual information extraction system for L_1 is needed when using this architecture. It offers merely information extraction results in L_2, yet, makes no information extraction in L_2 available.

In contrast, the architecture in Figure 6 produces extraction results in L_2 while the text corpus is originally in L_1. Neither the architecture in Figure 6, nor the architecture in Figure 7 delivers monolingual information extraction for a new language without implementing it. There may be cases where these designs are of use, but they do not solve the problem of making information extraction available for a new language without actually implementing it.

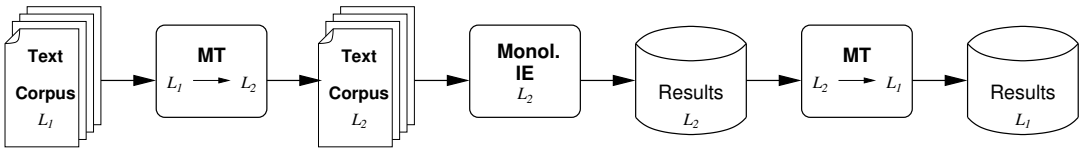

Figure 8: Monolingual information extraction using machine translation.

Figure 8 shows the addition needed to achieve a monolingual information extraction system for L_1 with the help of machine translation. A second machine translation system is needed[8] to translate the information extraction results back into L_1. That architecture has the same weakness as the first architecture we have seen in Figure 6. That is because machine translation is to be carried out on the full text corpus before information extraction.

The previous three architectures show us, that as long as machine translation performs unsatisfactorily on full text, it is only of small help in making monolingual information extraction available for new languages. So far only general off-the-shelf machine translation systems have been used in combination with information extraction. Due to the linguistically restricted and often domain specific contexts of information extraction, a machine translation system adapted to specific contexts could perform much better.

One limitation of machine translation is worth observing closer due to its relevance to multilingual information extraction. A machine translation system requires a transfer system which translates words, phrases or sentences from a source language L_1 into a target language L_2. If the previously performed linguistic analysis is rather shallow, this transfer system will necessarily contain a large amount of language specific rules for both L_1 and L_2. A machine translation system which analyses the linguistic structure of L_1 in detail, will probably need fewer language specific heuristics within the transfer system. Figure 9 on the following page depicts this coherence.

[8]The machine translation systems $MT_{L_1 \to L_2}$ and $MT_{L_2 \to L_1}$ are not equivalent and need to be constructed separately. In fact, there are cases where $MT_{L_1 \to L_2}$ is performing well, whereas, $MT_{L_2 \to L_1}$ performs poorly due to a more complex language structure of language L_2. Known cases with such problems are, for example, English-Russian and English-Japanese.

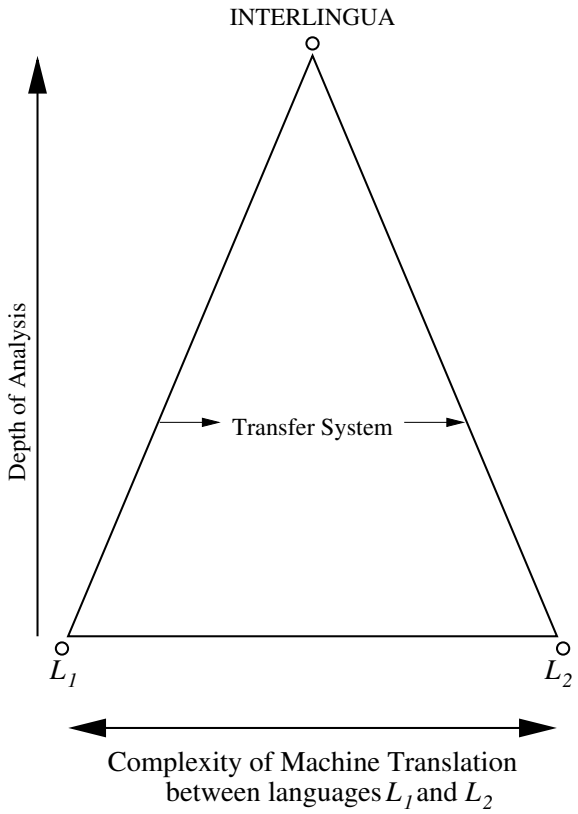

Figure 9: Coherence of interlingua and complexity of machine translation.

Detailed analysis is more expensive and one would think it is better to have an extensive transfer system. Yet, the drawback comes if we need to add another language into the system. To translate between three languages L_1, L_2, and L_3, for instance, six transfer systems ($L_1 \rightarrow L_2$, $L_2 \rightarrow L_1$, $L_1 \rightarrow L_3$, $L_3 \rightarrow L_1$, $L_2 \rightarrow L_3$, $L_3 \rightarrow L_2$) are needed. The number of transfer systems needed grows exponentially with linear growth of languages in the system. In an environment where translation between several languages is needed, we obviously will have to try to keep transfer systems as light-weighted as possible. We could try to find an abstract language description that contains all information in a language independent fashion. This, however, would require a full linguistic analysis, but would in turn decrease the number of translation systems dramatically. This so called *interlingua* could be used to construct a representation in any language without any translation by just inverting the analysis process. However, the construction of an all-purpose interlingua is probably impossible, and construction of a full linguistic analysis tool hypothetical [Arn+94].

Nevertheless, the idea of interlingua has been adapted to information extraction. While a full text interlingua representation is out of reach, it has been observed that due to simple linguistic structure of the data tuples of information extraction results, it is possible to find language independent representations for them. Numbers, location names, country names and titles, for example, may not need translation at all. Others can be represented in a way which makes the generation of language specific representation possible. Somewhat more difficult is the interlingua representation of verbs [XNS00].

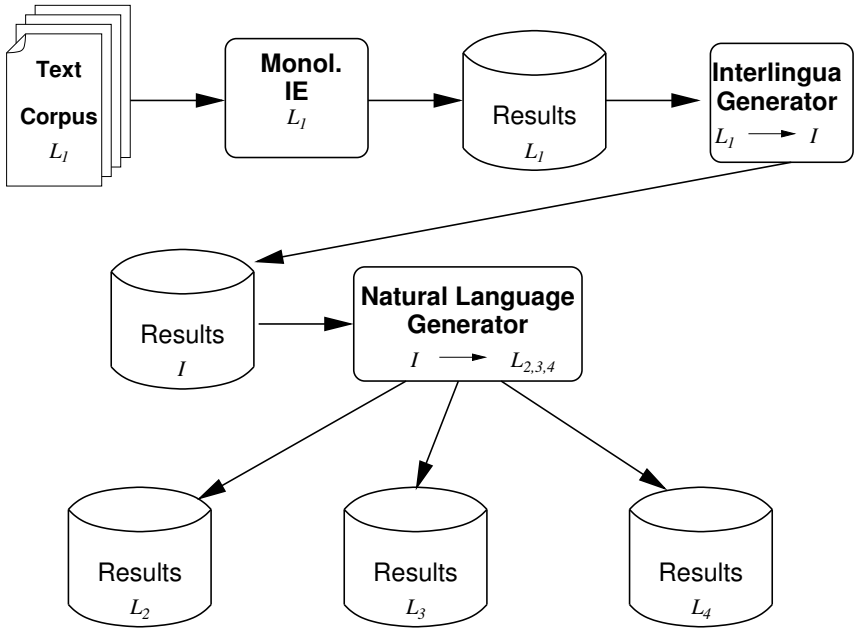

Figure 10: Interlingua in information extraction.

In Figure 10 an information extraction system that uses an interlingua result representation I, is outlined. After information extraction in language L_1 is completed, an interlingua generator translates all result tuples into language independent representation I. Later on the results stored in language independent representation I can be translated into languages L_2, L_3 and L_4 by using a natural language generator.

Theoretically, we could improve that architecture further by generating an interlingua representation of the text corpus and applying information extraction on the interlingua version of the text corpus. We would only need one information extraction system for the interlingua texts. An appropriate interlingua for full text, however, is yet to be invented.

3.4 Text Alignment and Cross-Language Projection

The previous section considered the problem of introducing new languages into information extraction in a rather coarse scope. Instead of treating information extraction as a black box task, we can adapt the components of information extraction systems into new languages. That enables us to choose suitable methods for individual languages and components in the case of language resource shortage. We will also benefit from efforts of related language technologies. A range of approaches are already developed for adapting common linguistic tools into new languages.

One common method to construct linguistic tools is to train these based on Hidden Markov models using tagged training corpora and the Viterbi algorithm (see sections 2.2 and 2.3). Cross-language projection attempts to make such training corpora available for new languages. With these training corpora, statistical linguistic tools can be made available. Examples for such tools are POS taggers, named entity recognition, noun phrase bracketers, and morphological analysers. All that is required is a tagged training corpus for an already developed language, and a parallel text corpus for the new language. We then project the tags across the parallel corpus using text alignment.

A parallel bilingual text corpus $T_{L_1,L_2} = T_{L_1} \cup T_{L_2}$ is a text collection (t_{i,L_1}, t_{i,L_2}), $t_{i,L_1} \in T_{L_1}$, $t_{i,L_2} \in T_{L_2}$ where for each text t_{i,L_1} written in language L_1, the text t_{i,L_2} is the translation into language L_2. T_{L_2} is the parallel text corpus of T_{L_1}.

An aligned text corpus T_{L_1,L_2} has an alignment between every corresponding text pair (t_{i,L_1}, t_{i,L_2}). In *sentence aligned* text pairs, every sentence $s_{j,L_1} \in t_{i,L_1}$ is aligned to a sentence $s_{k,L_2} \in t_{i,L_2}$. *Word aligned* text pairs contain alignments between single words in each sentence. Due to different language structures a 1-to-1 word-by-word alignment is not always possible as the following examples show.

Examples (S3.1) - (S3.6), on the following page, show possible scenarios of word alignment between POS tagged English and French named entities. The tags in these examples are from the 45-tag Penn Treebank tag-set [MSM93]. The tag '*DT*' marks a determiner, '*NN*' a noun in singular, '*NNS*' a noun in plural and '*VBG*' describes a verb in gerund.

(S3.1)

DT	*NNS*
The	*laws*
Les	*lois*
DT	*NNS*

(S3.2)

VBG	*NN*
living	*room*
∅	*salon*
	NN

The 1-to-1 alignment (S3.1) is an example of straightforward word-by-word alignment. However, the other examples show that 1-to-1 alignment is not always possible. (S3.2) is an example of n-to-1 alignment. There are different ways to align the words in the case of several words translating into one word. The goal here is to project POS tags across the alignments. If we would align n source language words (*living* and *room*) to one target language word (*salon*) the projection result would be a POS tag 'VBG NN' for the French noun *salon*, whereas, the correct POS tag is 'NN' only. One way to deal with this is to align some superfluous words to the empty word ∅ as is done in (S3.2) and (S3.4).

(S3.3)

NNS	
Laws	
Les	*lois*
NNS	*NNS*

(S3.4)

	NNS
∅	*Laws*
Les	*lois*
∅	*NNS*

The 1-to-n alignment encounters similar problems as we can see in the examples (S3.3) and (S3.4). In both examples the projection of POS tags does not produce the correct POS tag 'DT' for *Les*.

(S3.5)

		NNS	
		potatoes	
pommes	*de*	*terre*	
NNS	*NNS*	*NNS*	

(S3.6)

	NNS	
	veterans	
anciens		*combattants*
NNS		*NNS*

Also the word-to-phrase alignment examples (S3.5) and (S3.6), which are typical examples of phrasal 1-to-n alignments, show that projected tags are often erroneous across 1-to-n alignments. In certain cases one simple heuristic rule will be enough to post-process and remove those errors. There are, however, cases where mistakes are more complex to locate and correct [YNW00, ON00].

Let us now consider how cross-language projection can be used to introduce information extraction and its components into new languages. If a parallel bilingual corpus is available, for example, various bible translations and the Canadian Hansards[9], we can use the following configuration:

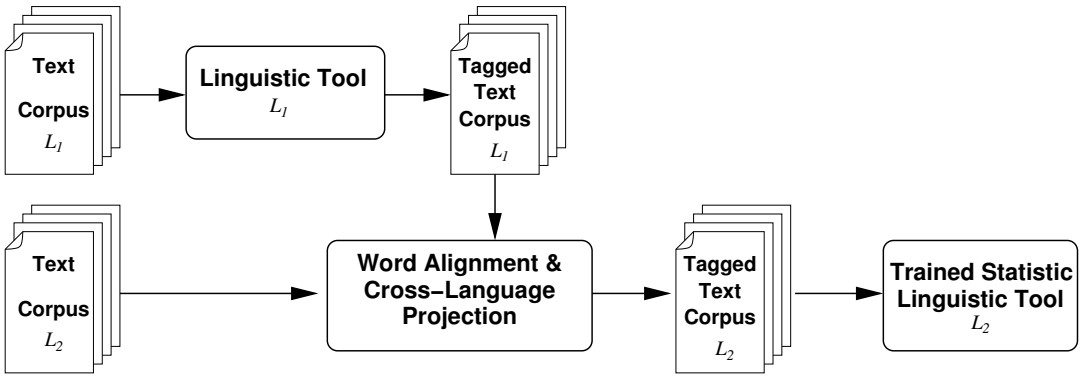

Figure 11: Cross-language projection on parallel corpora.

As outlined in Figure 11, we need a parallel bilingual text corpus T_{L_1,L_2}. We assume that a certain linguistic tool for language L_1 is available and apply it on T_{L_1}. Next, we execute word alignment on the parallel bilingual corpus so that we can project the tags from T_{L_1} to T_{L_2}. The result is a tagged text corpus T_{L_2}. We can use T_{L_2} for training a similar linguistic tool for language L_2 using the statistical methods described previously.

There are only a few given bilingual parallel text corpora available, restricting the number of occasions when this architecture can be of use. It may be as much effort to obtain a parallel corpus as to create monolingual tools for the new language from scratch, especially, when the texts belong to a specific domain language. Bearing in mind the restrictions pointed out in the previous section, we can use machine translation in creating bilingual parallel text corpora.

[9]The Canadian Hansards are the official records of the Canadian Parliament. A part of the Canadian Hansards is available as bilingual text corpus with sentence alignment in English and French language.

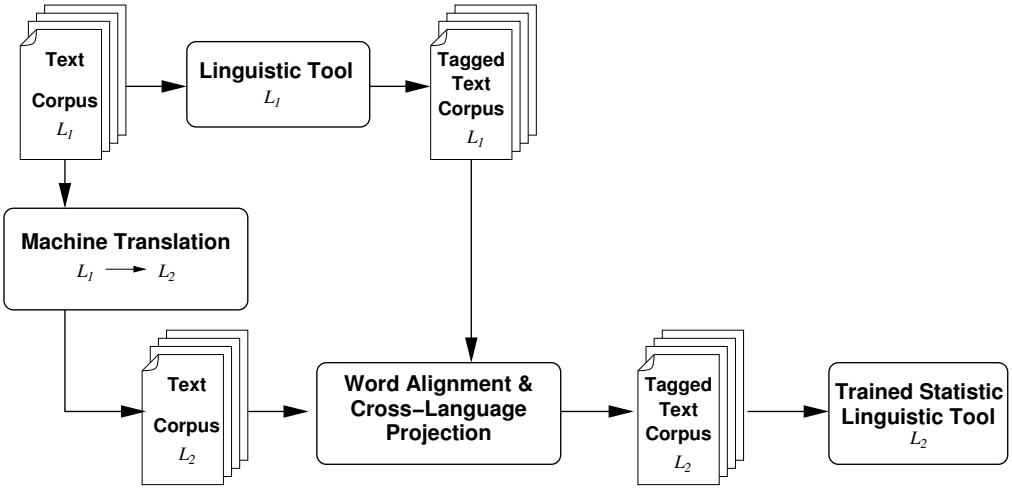

Figure 12: A cross-language projection architecture with machine translation.

Figure 12 shows where machine translation fits into the cross-language projection architecture. Machine translation from L_1 to L_2 is used to create a parallel text corpus T_{L_2} of the existing text corpus T_{L_1}. Yet, that means that machine translation is applied to full text and it is likely to lead to errors caused by bad translation. Alignment, however, benefits from machine translation because machine translated text is usually already sentence aligned.

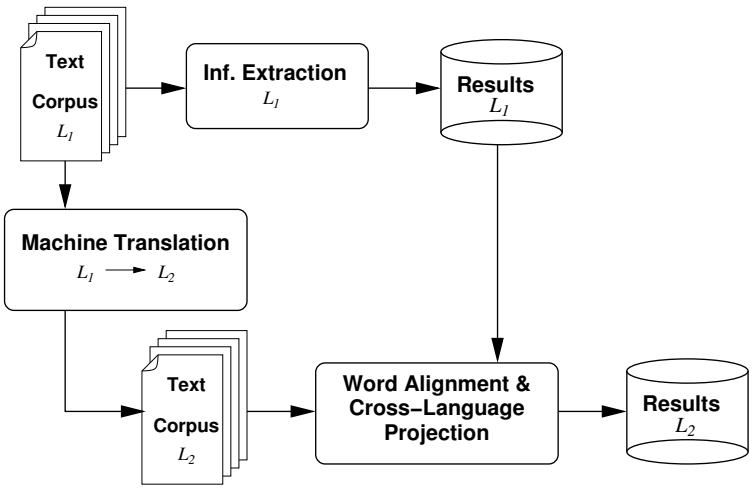

Figure 13: Cross-language projection of information extraction results.

Cross-language projection can also be used to project information extraction results across a parallel bilingual text corpus as outlined in Figure 13. We need to tag the information extraction results in the text corpus T_{L_1} in order to projected them across the word alignment. This is an alternate method for translating information extraction results [YNW00, RSY02].

3.5 Part-of-Speech Tagging

POS tagging is a language specific task. In a multilingual environment we must create an individual POS tagger for each and every language involved. It does not appear to be possible to re-use parts of part-of-speech taggers in a multilingual environment. There are three methods widely used for English POS tagging (see section 2.2). We can choose to implement any of those in the new language. Yet, in difference to English language, resources, such as tagged training corpora or grammar rule sets, are not available in most other languages. Neither can we expect those methods to perform as well as they do on English texts. In fact, there may be languages for which POS tagging is easier than English as well as those for which it is more difficult. The reason for this is differently distinct or complex language structure. Many languages have more complex grammar and morphologic structures than English. That may have effect on some methods and languages. However, without closer investigation it is difficult to predict which method works best. Usually, the resources available in a language will already imply the starting point.

The rule based approach to POS tagging requires linguistic expertise in the language and is a time consuming task. Each language has its own grammatical phenomena which may be very different to that of English. In most languages, rule based approaches will probably lead to slightly better POS taggers than stochastic taggers. Yet, they are expensive and difficult to construct.

Stochastic approaches depend on the availability of training data. The construction methods for stochastic English taggers can be easily adapted to new languages. The probabilities for whole sentence POS tags can be determined using the Viterbi algorithm, but only if a large tagged training corpus is available.

Brill taggers using transformation based learning can be trained for any language as long as a tagged training corpus is available. Brill taggers have already successfully been constructed for a number of languages [Bri95].

When we take a closer look on the options for creating a POS tagger for a new language, it seems that the shortage of resources is the decisive bottleneck. That insight has led to the invention of methods utilising other language resources available. A few methods have been suggested using cross-language projection.

As described in section 3.4, POS tags can be projected across parallel aligned text corpora. The resulting corpus with projected tags can be employed to train POS taggers using stochastic methods or transformation based learning [YNW00].

Alternatively, bilingual dictionaries can be utilised to tag texts in a new language. A dictionary entry for one word in the new language L_2 typically contains a set of possible translations into the language L_1 where a POS tagger is available. Assuming we know the probabilities of the possible POS tags for all these translations, we can calculate the average probability for a certain POS tag based on all these translations and use it as the probability for the tag of the word in L_2 [CY02].

3.6 Named Entity Recognition

Recognition of named entities is an important part of information extraction. Unrecognised named entities have noticeable negative effect on the overall performance of information extraction. Introduction of new languages to information extraction gives rise to the need for named entity recognisers. We need to construct named entity recognisers for new languages and new semantic classes, as named entity recognition is usually specialised to a certain language domain. Four main methods are used in English language to build named entity recognisers: Semantic lexica, that contain a list of proper names; rule based named entity recognisers, that use handcrafted heuristics (we require linguistic and domain expertise to implement these rule); stochastic named entity recognisers, that are trained on tagged training corpora (these training corpora need to be tagged or must be already available somehow); and bootstrapping, that automatically builds semantic lexica. Transformation based learning can be used to combine stochastic and rule based named entity recognition. All these methods can only be adapted for new languages when either training corpora or linguistic expertise are available in that language. In many languages these resources are not available. Thus, other approaches have been tried in order to craft named entity recognisers for new languages.

One alternative method to create training corpora for a new language L_2 is cross-language projection. If machine translation between L_1 and L_2, and a tagged training corpus in L_1 is available, a training corpus in L_2 can be obtained (see also section 3.4) [YNW00].

Another method for named entity recognition, which bypasses the problem of resource shortage, is based on a bootstrapping method. It requires an untagged training corpus for a certain language and domain, and about 100 seed words for different semantic classes. The idea is to use words morphological structures to identify named entities. The same type of words often share common prefixes or suffixes. Moreover, most languages contain such common word prefixes or suffixes.

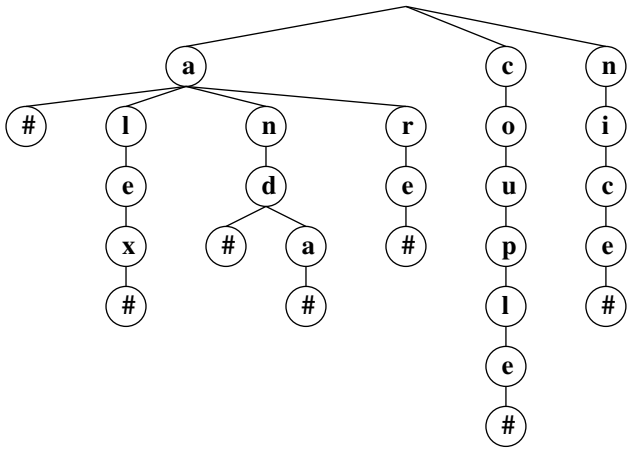

Figure 14: A morphological prefix trie.

We can use trie structures to represent all found prefixes and suffixes in a text. Figure 14 shows the prefix trie for the sentence (S3.7). The '#' character represents word boundaries. Alike, trie structures can be used to represent context structures, for example, each trie node contains a token/word. The path from root to leave nodes represents the forward or backward context the token appears in.

(S3.7) *Alex and Anda are a nice couple.*

We use the text corpus to build tries for named entity recognition as follows:

1. Generate prefix and suffix tries, and forward and backward context tries of the corpus as preparation for the bootstrapping. During this stage, we gather statistics and save them within each node. We also need to count the following probabilities in the text corpus: For prefix and forward context tries we count the probability for the node to be an ending node, for suffix and backward context tries we count the probability to be a starting node of a token.

2. During the bootstrapping process, introduce all seed words sequentially into the tries and recompute the node probabilities. In every trie node a probability for each node to represent the starting or ending point of a certain semantic class is generated whenever a seed word is matched morphologically. When the probability of prefix- or suffix trie node to belong to a certain semantical class boosts over 0.5, the context will also gain a higher probability to embed a named entity of that semantical class. When the probability of a semantical class in a context trie node gains a probability of over 0.5, all words appearing in that context are added to the seed words.
3. The bootstrapping ends when no seed words are left.

Trie based bootstrapping can be used to build statistical named entity recognisers for languages with suitable morphological structures [CY99].

Named entity recognition is not strictly a language dependent task. Many named entities have similar or identical representations in a number of languages. Some closely related languages have such a similarity between them that recognisers in one language produce usable results in related languages [Poi+03].

A multilingual named entity recogniser can be designed so that it is able to re-use its language independent parts. It is also desirable that we can integrate different recognition methods. This enables us to use the most economical or easily implementable named entity recognition method for each individual language. It is also desirable to modularise language domain specific features in the architecture.

Figure 15, on the following page, outlines a multilingual named entity recognition architecture which integrates rule based named entity recognition, stochastic named entity recognition, and named entity recognition using transformation based learning. At a first glance it may not seem obvious, but the architecture also implements multilinguality in a modularised fashion. The architecture is built on the transformation based learning architecture that iteratively applies rule based and stochastic named entity recognition methods on the text. In the case where only one method is available in any given language, we simply omit the other methods.

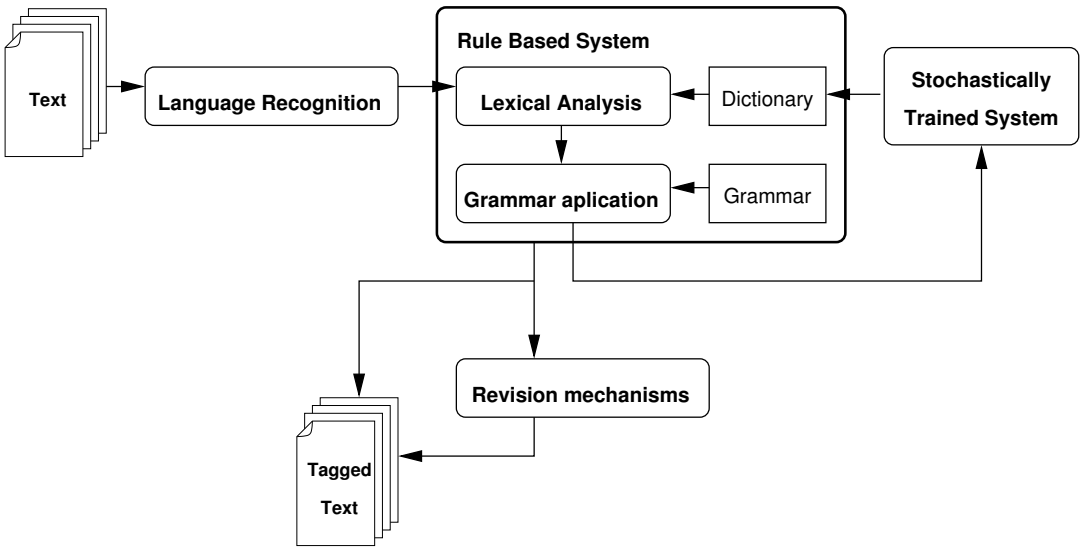

Figure 15: A multilingual named entity recognition architecture.

We archive language modularisation as follows: Initially, language recognition is applied on each text. Lexical analysis is typically language dependent and every text is handed over to its own language morphological analyser and POS tagger. Grammar rules may be language dependent or independent. Every text is first handed to all language independent rules and then to its language dependent rules. We implement this as a rule tree. The text is then handed down from rule to rule starting from the root and ending at a leave according to its language.

Stochastic systems are typically language dependent. Languages that are closely related may benefit from the application of stochastic named entity recognition of a related language if one is not available for the required language.

In a few cases human revision may be required, to resolve known cases of ambiguity. The *Washington* example (S2.6), on page 12, depicts one of these cases. We can integrate language domain modularisation by integrating specific rules into the rule tree structure [Poi+03, Gro+02].

3.7 Coreference and Discourse Analysis

Coreference analysis is important for information extraction as it connects parts of information which otherwise would be lost in the discourse of text. Adaption of English coreference analysis methods means either founding coreference analysis on knowledge bases or training corpora. The decision on the method is usually motivated by the availability of resources for the new language. Language specialities, however, may be a restrictive factor for that decision. The knowledge based approach means implementing rules by hand. This requires expert knowledge of the new language and its grammar. The training corpus approach requires a set of tagged texts in the new language. Some concepts for multilingual coreference resolution architectures have been published already. These coreference resolvers are designed to deal with texts in various languages and re-use all components of the analysis tool, some of which are not strictly language dependent.

The multilingual information extraction system M-LaSIE implements a multilingual coreference resolver founded on a knowledge base. In the M-LaSIE system the text is first POS tagged and fully parsed. It is then transformed into a *quasi logical form*(QLF), a representation which embodies lexical and syntactical information as first level logical predicates. These QLFs are then connected to a semantic net into which coreferences can easily be added.

The QLF representation of text itself is language independent, only the transition into QLF is language dependent. Hence, the task of coreference resolution in the semantic net is language independent. When adding new items to the semantic net, we apply the following four rules to identify possible anaphora.

1. New items which are proper names are compared to all existing items that are proper names.

2. All new items are compared with each other.

3. New items which are pronouns are first compared with all items in the paragraph, and then with each previous paragraph in turn until the antecedant is found.

4. Other new items are compared with all existing items in the semantic net.

The heuristics of this coreference resolver are fairly simple and benefit from the thorough language analysis undertaken as preparation. This coreference analyser resolves the simpler anaphora easily but overlooks the complex coreferences. [AHG98].

Another multilingual coreference resolution architecture can be constructed from a heuristic knowledge base. As prerequisite we need a POS tagged text corpus. The system gains multilinguality by sorting the heuristics into categories and within those categories into language independent and language dependent heuristics. This method is quite similar to translation based learning used for POS taggers (see section 2.2).

We depict the system as a hierarchical tree structure of heuristics where candidate coreferences are passed down individual paths along the tree. The path depends on the nature of the coreference as well as the language it consists of. Every heuristic rule scores the candidate link. Some heuristics are also able to rule out and remove candidates. Finally, we choose the highest ranking link of each item to be the coreference. The system can perform on texts in any language. In order to improve the quality, some language dependent heuristics need to be added to the knowledge base for each newly added language. The performance of this system depends mainly on the amount and quality of heuristic rules in the knowledge base [Aon94, MBS98].

An alternative approach is taken by the SWIZZLE system using multilingual coreference analysis. It has been claimed that SWIZZLE even improves the performance of monolingual coreference analysis. As prerequisite, an aligned parallel bilingual corpus is required which already contains coreference resolution tags for both languages. The SWIZZLE system matches the different, already resolved, anaphora between the aligned corpora in finding resolutions for coreferences in the other language. The system works on the assumption that some coreferences are easier to resolve in some languages than in others [HM00].

4 A Multilingual Framework for Information Extraction

In the previous chapters we have investigated the components of information extraction and how they can be adapted to new languages and multilingual environments. In this Chapter I wish to propose an architecture to integrate these components into, a multilingual information extraction framework. The framework will have the following characteristics:

i. The architecture is to be open for new languages and language domains.

ii. Language dependent components can be integrated without harming the multilingual nature of the system.

iii. Results can be retrieved in any language implemented.

These key characteristics ensure that the framework is extensible and flexible for general use as well as domain specific information extraction.

Figure 16 on the following page shows the architecture for a multilingual information extraction system. The large rectangular boxes represent the components of the information extraction system and the small rectangles represent resources (lexical dictionaries, name lexicons, rule sets and seed pattern lists). The dashed arrows depict optional paths in the work flow, indicating that subtasks of POS tagging and syntax analysis can also be omitted.

As seen in the previous chapter, some components of information extraction can be implemented as multilingual components, not consisting of a collection of monolingual components. The algorithms of these components process, without adjustment, a number of languages. Other components are strictly language dependent and individual implementations for each and every language are required. Language dependent components have their own language specific resources that need to be maintained. The vertical dashed line in Figure 16 divides multilingual and language specific components and resources.

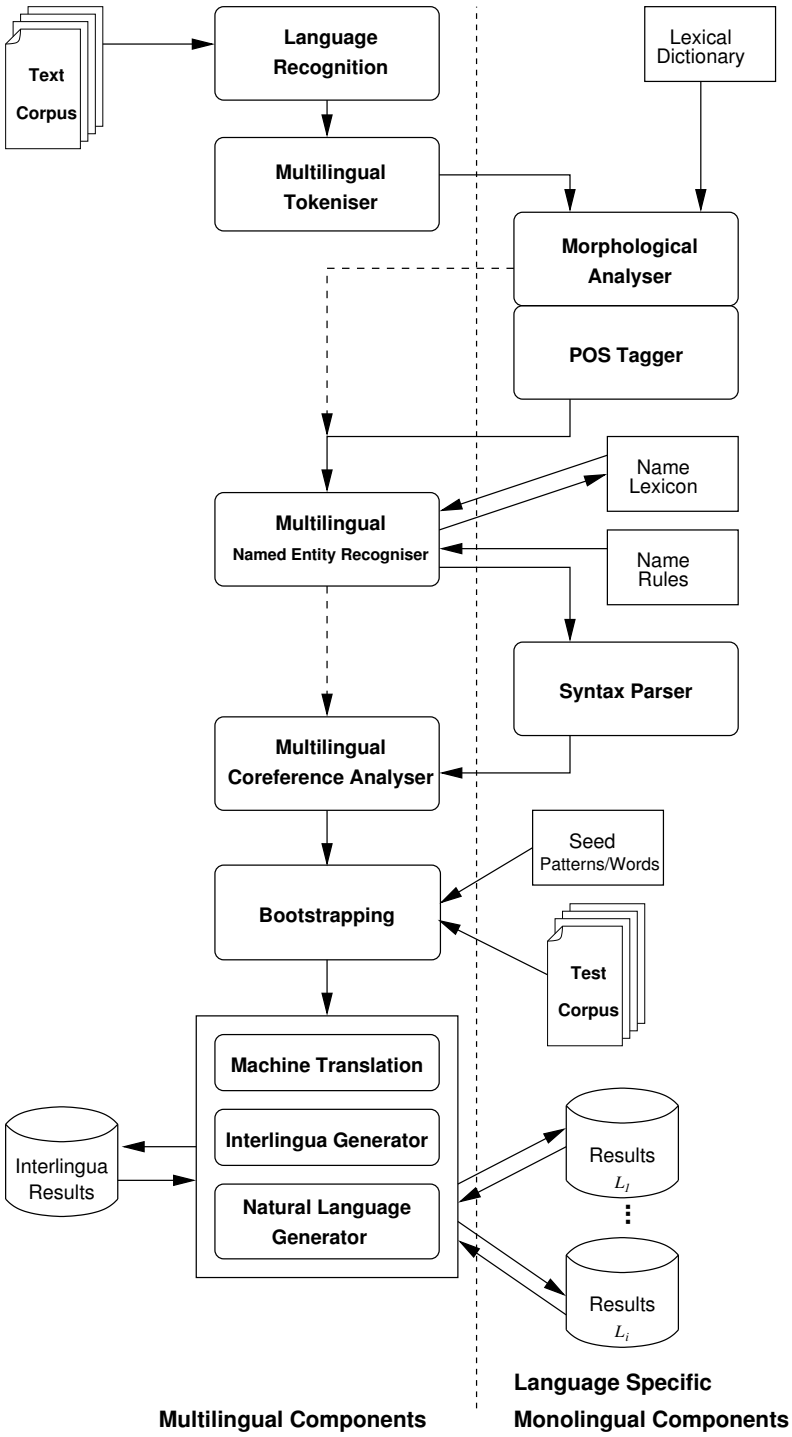

Figure 16: A multilingual information extraction framework.

The multilingual information extraction system outlined will do the following when applied on a text corpus: In the beginning the language (or languages) of the individual texts in the corpus is recognised and we tag this information to each text. We will see later how this language tag is used to decide which language specific components to apply. The successive components perform their tasks on the texts and after bootstrapping we receive information extraction results. If an essential component is unavailable for the required language, the information extraction task must be rejected. Note that at this stage all information extraction results are in a language specific representation.

Let us now consider the problem of managing information extraction results in numerous languages. We have seen in section 3.3 how machine translation is employed to translate information extraction results and how an interlingua representation of the results is useful in decreasing translation work. We combine these concepts in this architecture to store information extraction results and deliver them in any language. We store information extraction results received in a language specific representation in their original form. If for the type of results an interlingua representation exists, the interlingua generator creates the interlingua representation of the results and they are stored in a separate database. When a language specific information extraction result that is not yet available is requested, we have two options: If an interlingua representation of the results exists, we can use the natural language generator to create a version of the results in the required language; or we can use machine translation to translate the results from the original language into the required one. If, however, we have neither a machine translation module for translating from the original into the requested language, nor an interlingua representation is available, or interlingua generation or natural language generation is not available, we cannot deliver the requested results [BNX98, Din98, XNS00].

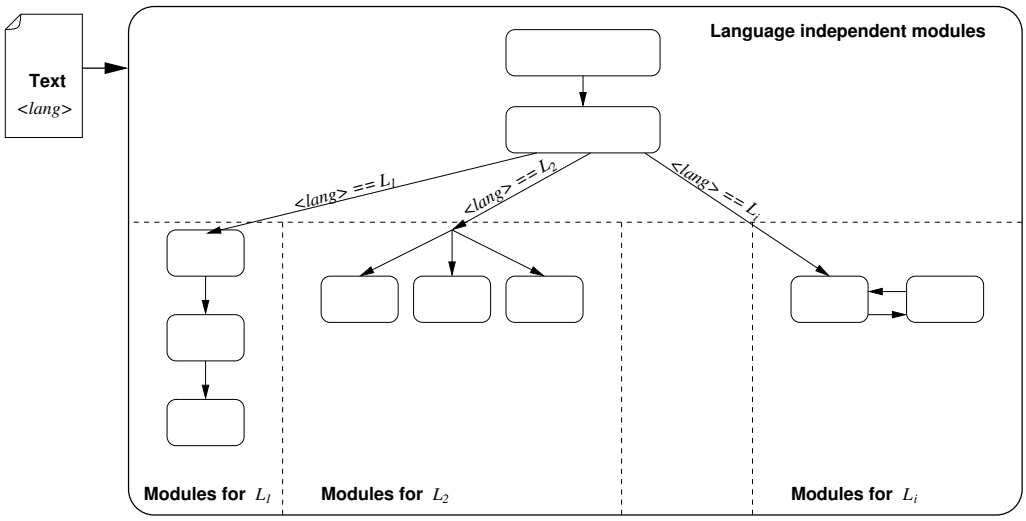

Figure 17: Multilingual component architecture using a tree layout.

It is important that we can embed components into the framework whenever they are available, as creating new information extraction components is laborious. Most components, however, are not created for use in a multilingual environment. To embed language dependent information extraction components, we will use a tree architecture that allows integrating of components in a modularised manner. We will see that this design equips every component of the system with a multilingual interface enabling easy addition of new languages. Figure 17 depicts the multilingual architecture for integrating language dependent components using a tree layout.

Let us assume that every text, on which the multilingual component is to be applied, has received a tag $<lang>$ during language recognition indicating the language of the text. This tag is used to decide which specific language dependent module to apply on the text.

We may have one or more language independent modules which we place at the root of the tree. This is depicted as two module boxes in the area marked as language independent in Figure 17. Language independent tasks are, for example, rule sets of rule based components that do not match on language specific features. While still rather uncommon, there might be more language independent tasks when components are modularised into language dependent and language independent parts.

We use the $<lang>$ tag as decision criteria on which path a text progresses through the module tree. The language specific modules can be arranged in various layouts. Three examples are suggested in Figure 17.

The first layout (down the path L_1, at the bottom left of Figure 17) is a simple linear sequence of one or more language specific modules applied on the text. This layout is appropriate where we have only one component per language available.

The second layout (down the path L_2, at the bottom centre of Figure 17) is designed to offer choice where several mutually exclusive components are to be integrated. That will be the case, for instance, for rule based and statistical tagging tools. The system will be configured to use one of those modules as default but the user can opt to use any of the others for the task at hand. Using one module rules out the use of the others. A somewhat similar layout can be used to simultaneously apply mutually exclusive modules. This could be useful for comparing or merging results produced by different modules.

The third layout (down the path L_i, at the bottom right of Figure 17) follows the method of iteratively applying two or more modules that we have investigated for named entity recognition (see also Figure 15 in section 3.6). Assuming we have two or more component modules complementing each other, we can choose to iteratively apply them on the text.

We can add new language capabilities to the multilingual component by inserting another subtree for any new language. If a language has not yet been implemented, we can configure the component to apply the modules of a related language on the text. We can also integrate user interaction to aid, for example, in ambiguity resolution.

In summary, the multilingual information extraction framework recognises text language, delegates the texts of various languages to the appropriate components and manages the multilingual information extraction results. The system is restricted by the language specific implementations. The overall quality depends on the quality of the implementation of components that are used throughout a specific information extraction task [Gro+02, Poi+03].

5 Conclusions

The Internet increasingly consists of texts in multiple languages. Information extraction is a powerful tool to access information in large text collections. There is a need for multilingual information extraction to access information spread across language borders. While various topics related to multilingual information extraction have been examined sufficiently, little has been attempted to consolidate these related topics in a general multilingual information extraction framework. This paper has reviewed the problems and possibilities surrounding multilingual information extraction and its underlying technology. Through a review of monolingual English information extraction, I have established the key information extraction components and how they relate to each other. I have united information extraction and numerous multilingual concepts in an attempt to propose a general multilingual information extraction framework.

I have studied the main methods available to build the key information extraction components. Assessing the restrictions and requirements of these is important when creating information extraction components in new languages. As multilingual information extraction can be assembled from monolingual components, constructing them for new languages is another step towards multilingual information extraction. The methods used for creation of English components can be imitated in other languages. Yet, it is important to be able to estimate what will be needed to construct similar information extraction components for new languages. Possessing information extraction components for various languages alone is not necessarily sufficient to achieve multilingual information extraction. In consequence, I have proposed definitions of monolingual and multilingual information extraction as well as monolingual and multilingual linguistic tools to highlight when multilinguality is achieved, and when it is not. I have defined multilinguality based on the number of input languages a system accepts.

I have analysed various concepts employing machine translation in combination with information extraction, some of which can be used to provide information extraction in new languages. Most of those layouts, however, suffer from one or another drawback and are useful in a few circumstances only. Machine translation turns into a more powerful tool when combined with cross-language projection across aligned bilingual corpora, enabling the creation of information extraction components and other stochastic linguistic tools for new languages.

I have discussed the options available for the adaption of part-of-speech tagging, named entity recognition, coreference analysis and discourse analysis to new languages. Stochastic methods and handcrafted rules can be employed as well as methods using cross-language resources or language similarities to make new language resources available. The construction of any tool requires linguistic resources and expertise. Hence, the availability of useful resources is the restrictive factor in bringing information extraction to new languages. I have reviewed a language recognition method that is able to identify the language of individual texts. Language recognition of each text is a necessary preparation for multilingual information extraction.

Multilingual information extraction produces results in multiple languages. To make these results useful, the language border can be crossed using interlingua representation for the results. Even though in natural language processing interlingua is more a theoretical concept than a practical solution to various problems, creating and utilising interlingua for the restricted language of information extraction results is possible. The multilingual information extraction framework presented in this paper is to be as general as possible and open to extension. Other layouts are probably more efficient when extendability is not required.

For the time being the availability of resources for languages concerned dictates the methods which can be used for multilingual information extraction. It is to be expected that the number of linguistic resources will grow for the most common languages in the coming years. Greater availability of resources offers more possibilities for text processing tasks. The multilingual concepts especially, and those transferring knowledge from one language to another are like to evolve in the near future. Globally accessible information is and will be an ongoing motivation for research in this field.

References

[AHG98] Azzam, S., Humphreys, K., Gaizauskas, R.,
Coreference Resolution in a Multilingual Information Extraction.
Department of Computer Science, University of Sheffield, Sheffield, United Kingdom, 1998.
Also available at `http://citeseer.nj.nec.com/518290.html`

[AI99] Appelt, D. E., Israel, D. J.,
Introduction to Information Extraction Technology.
A Tutorial prepared for the 16th International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, Sweden, August 1999.

[Aon94] Aone, C.,
Customizing and Evaluating a Multilingual Discourse Module.
Proceedings of the 15th International Conference on Computational Linguistics (COLING),Vol. 2, p. 1109-1113, Kyoto, Japan, 1994.

[Arn+94] Arnold, D.J. et al,
Machine Translation: an Introductory Guide.
Blackwells-NCC, London, United Kingdom, 1994.

[Azz+97] Azzam, S., Humphreys, K., Gaizauskas, R., Wilks, Y.,
Using a Language Independent Domain Model for Multilingual Information Extraction.
Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI), Workshop on Multilinguality in the Software Industry: the AI Contribution (MULSAIC), Nagoya, Japan, August 1997.

[BNX98] Buitelaar, P., Netter, K., Xu, F.,
Integrating Different Strategies for Cross-Language Information Retrieval in the MIETTA Project.
Proceedings of the 14th Twente Workshop on Language Technology (TWLT), Enschede, Netherlands, December 1998.
Also available at `http://citeseer.nj.nec.com/435447.html`

[Bol+97] Bolioli, A. et al,
MILK: a Hybrid system for Multilingual Indexing and Information Extraction.
Proceedings of the Second Conference on Recent Advances in Natural Language Processing (CRANLP), Tzigov Chark, Bulgaria, September 1997.

[Bri95] Brill, E.,
Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging.
Computational Linguistics, 4 (Vol. 21), p. 543-565, MIT Press, Massachusetts, December 1995.

[Che03] Chen, K. H.,
Indexing and Abstracting.
Lecture notes, Department of Library and Information Science, National Taiwan University, Taiwan, 2003.
Also available at `http://www.lis.ntu.edu.tw/~khchen/course/ia/slide/ia200312.pdf`

[CY99] Cucerzan, S., Yarowsky, D.,
Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence.
Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing (EMPL) and Very Large Corpora (VLC), Maryland, June 1999.

[CY02] Cucerzan, S., Yarowsky, D.,
Bootstrapping a Multilingual Part-of-speech Tagger in One Person-day.
Proceedings of the Sixth Workshop on Computational Natural Language Learning (CoNLL), Taipei, Taiwan, August-September 2002.

[Din98] Dini, L.,
Parallel Information Extraction Systems for Multilingual Information Access.
Proceedings of the European Intelligent Systems and Control Conference (Euriscon), Athens, Greece, June 1998.

[FM99] Freitag, D., McCallum, A. K.,
Information Extraction with HMMs and Shrinkage.
Proceedings of the 16th National Conference on Artificial Intelligence (AAAI), Orlando, Florida, 1999.

[Gri97] Grishman, R.,
Information Extraction: Techniques and Challenges.
Technical Report 00-95, Computer Science Department, New York University, New York, 1997.

[Gri+99] Grishman, R. et al,
Multilingual Information Management: Current Levels and Future Abilities, Chapter 3: Cross-lingual Information Extraction and Automated Text Summarization.
New York, April 1999.
Also available at `http://www-2.cs.cmu.edu/~ref/mlim/`

[Gro+02] Grover, Claire, et al.,
Multilingual XML-Based Named Entity Recognition for E-Retail Domains.
Proceedings of the Third International Conference on Language Resources and Evaluation (LREC), Las Palmas, Spain, May 2002.

[Hag99] Hagman, J.,
Construction and Performance of a Language Recognizer.
Joint Research Centre of the European Commission, Institute for Systems, Informatics and Safety (CSIS), Ispra, Italy, September 1999.
Also available at `http://citeseer.nj.nec.com/533776.html`

[HGA97] Humphreys, K., Gaizauskas, R., Azzam, S.,
Event Coreference for Information Extraction.
Proceedings of the Eighth Conference of the European Chapter of the Association for Computational Linguistics (EACL), Workshop on Operational Factors In Practical, Robust, Anaphora Resolution for Unrestricted Texts, p. 75-81, Madrid, Spain, July 1997.

[HM00] Harabagiu, S., Maiorano. S.,
Multilingual Coreference Resolution.
Proceedings of the First Conference of the North American Chapter of the Association for Computational Linguistics (NAACL) and Sixth Conference on Applied Natural Language Processing (ANLP), Seattle, Washington, May 2000.

[JM00] Jurafsky, D., Martin, J. H.,
Speech and Language Processing.
Prentice Hall, New Jersey, 2000.

[Kam97] Kameyama, M.,
Information Extraction across Linguistic Barriers.
AAAI Spring Symposium Series on Cross-Language Text and Speech Retrieval, Stanford, 1997.

[MUC91] *Proceedings of the Third Message Understanding Conference (MUC-3).* Morgan Kaufmann Publishers, San Francisco, California, 1991.

[MUC92] *Proceedings of the Fourth Message Understanding Conference (MUC-4).* Morgan Kaufmann Publishers, San Francisco, California, 1992.

[MUC93] *Proceedings of the Fifth Message Understanding Conference (MUC-5).* Morgan Kaufmann Publishers, San Francisco, California, 1993.

[MUC95] *Proceedings of the Sixth Message Understanding Conference (MUC-6).* Morgan Kaufmann Publishers, San Francisco, California, 1995.

[MUC98] *Proceedings of the Seventh Message Understanding Conference (MUC-7).* Morgan Kaufmann Publishers, San Francisco, California, 1998.

[MSM93] Marcus, M. P., Santorini, B., Marcinkiewicz, M. A., Building a large annotated corpus for English: The Penn treebank. *Computational Linguistics, 19(Vol. 2), 313-330*, MIT Press, Massachusetts, 1993.

[MBS98] Mitkov, R., Belguith, L., Stys, M., Multilingual robust anaphora resolution. *Proceedings of the Third Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Granada, Spain, June 1998.

[ON00] Och, F. J., Ney, H., A Comparison of Alignment Models for Statistical Machine Translation. *Proceedings of the 18th International Conference on Computational Linguistics (COLING)*, Saarbrücken, Germany, 2000.

[Poi00] Poibeau, T., A corpus-based approach to Information Extraction. Laboratoire d'Informatique de Paris-Nord, Paris, France, 2000. Also available at `http://thierry.poibeau.free.fr/articles/jass.pdf`

[Poi+03] Poibeau, T., et al, The Multilingual Named Entity Recognition Framework. *Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Budapest, Hungary, April 2003.

[Ril93] Riloff, E.,
Automatically Constructing a Dictionary for Information Extraction Tasks.
Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI), Massachusetts, 1993.

[Ril96] Riloff, E.,
Automatically Generating Extraction Patterns from Untagged Text.
Proceedings of the 13th National Conference on Artificial Intelligence (AAAI), Portland, Oregon, 1996.

[RJ99] Riloff, E., Jones, R.,
Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping.
Proceedings of the 16th National Conference on Artificial Intelligence (AAAI), Orlando, Florida, 1999.

[RSY02] Riloff, E., Schafer, C., Yarowsky, D.,
Inducing Information Extraction Systems for New Languages via Cross-Language Projection.
Proceedings 19th International Conference on Computational Linguistics (COLING), Taipei, Taiwan, August-September 2002.

[SL94] Soderland, S., Lehnert, W.,
Wrap-Up: a Trainable Discourse Module for Information Extraction.
Journal of Artificial Intelligence Research, Vol. 2, p. 131-158, Massachusetts, 1994.

[SL95] Soderland, S., Lehnert, W.,
Learning Domain-Specific Discourse Rules for Information Extraction.
AAAI 1995 Spring Symposium on Empirical Methods in Discourse Interpretation and Generation, Massachusetts, 1995.

[Sod99] Soderland, S.,
Learning Information Extraction Rules for Semi-structured and Free Text.
Machine Learning, 1-3 (Vol. 34), p. 233-272, MIT Press, Massachusetts, 1999.

[TJ97] Tapanainen, P., Järvinen, T.,
A non-projective dependency parser.
Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP), Washington, D.C., 1997.

[Rij79] van Rijsbergen, C. J.,
Information Retrieval.
2nd edition, Chapter 7, Butterworths, London, United Kingdom, 1979.

[Vou97] Voutilainen, A.,
The EngCG-2 tagger in outline.
Department of Linguistics, University of Helsinki, Helsinki, Finland, 1997.
Also available at `http://www.ling.helsinki.fi/~avoutila/cg/doc/engcg2-outline/engcg2-outline.ps`

[XNS00] Xu, F., Netter, K., Stenzhorn, H.,
MIETTA − A Framework for Uniform and Multilingual Access to Structured Database and Web Information.
Proceedings of the Fifth International Workshop on Information Retrieval with Asian Languages (IRAL), Hong Kong, China, September - October 2000.

[Yan+00] Yangarber, R., Grishman, R., Tapanainen, P., Huttinen, S.,
Unsupervised Discovery of Scenario-Level Patterns for Information Extraction.
Proceedings of the Sixth Conference on Applied Natural Language Processing, (ANLP), Seattle, Washington, May 2000.

[YGT00] Yangarber, R., Grishman, R., Tapanainen, P., Huttunen, S.,
Automatic Acquisition of Domain Knowledge for Information Extraction.
Proceedings of the 18th International Conference on Computational Linguistics (COLING), Saarbrücken, Germany, 2000.

[YNW00] Yarowsky, D., Ngai, G., Wicentowski, R.,
Inducing Multilingual Text Analysis Tools via Robust Projections across Aligned Corpora.
Department of Computer Science, John Hopkins University, Baltimore, 2000.
Also available at `http://citeseer.nj.nec.com/yarowsky00inducing.html`

[Yul96] Yule, G.,
Pragmatics.
Oxford Introduction to Language Study, Oxford University Press, 1996.