6 Comparison of Existing Approaches

In this section we compare the approaches according to the types of tasks and texts they can handle as well as the types of features they consider. We also compare tagging requirements and learning characteristics. Table 2.2 on page 17 can be consulted to locate the detailed descriptions of approaches and systems.

6.1 Types of Tasks Handled

The main task handled by current IE systems is to fill a *template* that contains several *attributes,* which is typically done in two steps:

1. *Fragment extraction* (or *slot filling*) to find text fragments that yield suitable values for the defined attributes.
2. *Relationship recognition* (or *template unification*) to combine the found attribute values into templates, resolving coreferences as required.

The first step corresponds to the *extraction of explicit information* task and the second one to the *relationship recognition* task identified in Section 3.1. The other tasks described in that section as potential steps of a *comprehensive* IE approach are generally not yet handled by current IE systems.

Most of the described approaches handle the first step only. Hence they are limited to corpora where each document contains a single template; otherwise additional pre- or postprocessing is necessary to split the input at template boundaries or to arrange the found attribute values into adequate templates.

Some systems—*Crystal*[1], *Whisk* and *TIMES*—handle template unification at the sentence level. Thus no special processing is necessary if each template is expressed within a single sentence in a input text. This might be sufficient for some domains but it is not a general solution to the template unification task.

Other approaches go further by unifying templates at a logical level, beyond sentence borders: the *Amilcare* extension of *(LP)*^{2}, *IE*^{2}, *SIFT*, and the extended version of *SNoW-IE* (which in turn does not completely handle the fragment extraction task). However, *IE*^{2} requires hand-written rules for this purpose and *Amilcare* required rules specifying which attributes introduce new templates, so neither is a completely trainable solution. *SIFT* is a very early statistical system that in 1998 was able to reach near-state-of-the-art results compared to the hand-written system participating in the *MUC-7* conference but is unlikely to be still competitive today.

[1] Crystal does not identify exact attribute values but only sentence constituents containing attribute values, thus it always requires postprocessing.

6.2 Types of Texts Handled

Three types of texts are often distinguished (cf. [Sod99, Sec. 1], [Eik99, Sec. 2.5]):

- *Free texts* are grammatical natural-language texts, e.g. newspaper articles or scientific papers.
- *Semi-structured texts* are not fully grammatical and sometimes telegraphic in style, e.g. newsgroups or e-mail messages or classified ads.
- *Structured texts* contain textual information strictly following a predefined (but not necessarily known) format where items are arranged in a fixed order and separated by delimiter characters or strings. Examples are comma-separated values or web pages generated from a database.

Even though some systems are designed for certain types of texts, it cannot be assumed that some class of IE approaches is particularly suitable for a particular kind of text. Furthermore, all classes have in common that the performance on structured texts is better than on free texts.

Some approaches—the original version of *Crystal*[2], *IE*[2], *TIMES* and *SIFT*—rely heavily on linguistic information and are thus suitable for free texts only. Most other approaches are suitable for both free and semi-structured texts—they make use of linguistic information as far as it is available, but do not necessarily require it.

Most other systems make little or no use of linguistic knowledge, thus they are suited for semi-structured and structured texts. *Whisk*, *SRV* and *BWI* claim to be targeted at any text type, from free text to structured text. Approaches that allow variable input will play a major role in the future research, since in real world domains an IE system will be confronted with the large diversity of texts.

6.3 Considered Features

There is a wide variety in the types of features that are considered for learning by different approaches. All systems utilize the words (tokens) in a text as the main lexical features. Not only the presence or absence of a word but also the word order play an important role. Morphological information is used not quite as universally, but very frequently. Especially POS (part-of-speech) tags are used by a wide variety of systems. Some systems also utilize a stemmer or lemmatizer to determine the base forms of words.[3]

For linguistic information beyond the word level, several approaches[4] rely on simple chunkers that identify various types of clauses (noun, verb, prepositional clauses etc.) in a sentence. More refined chunk parsers that also assign grammatical roles for chunks (subject, direct or indirect object) are employed by *Crystal* and *Whisk* (for free texts). Only a single system, *SRV*, makes use of a deep parser (based on the link grammar theory). Rule and knowledge-based systems tend to embed more syntactic information since syntax is often used for rule construction. Statistical systems

[2] [Sod97b] describes an extension to semi-structured text.
[3] *(LP)*[2] and *BIEN*, optional for *Active HMMs*.
[4] Such as *TIMES*, *(C)HHMMs*, *BIEN*, and the extended version of *BWI*.

consider predominantly linguistic information related to single tokens due to their token-based processing of the text.

Semantic information is used less frequently than syntactic. Typically, it comprises simple gazetteers or word lists assigning semantic classes to words.[5] Some approaches[6] use a complete thesaurus, WordNet [Fel98]. Knowledge-based systems use their own built-in knowledge-bases.

Some approaches[7] consider features derived from the shape of words/tokens, e.g. token type (lower-case, capitalized, all-caps, digits, etc.) or prefixes and suffixes. Most approaches work on plain text input without formatting, but a few can utilize structural information from HTML or XML documents: *Stalker* and *BWI* can handle HTML tags (treating them as normal tokens), *Active HMMs* optionally consider the HTML context of text tokens.

While usually the handled types of features are fixed in advance, the *Amilcare* system chooses an adaptive way to consider linguistic information ("LazyNLP"): the amount of linguistic information available for learning rules is gradually increased until the effectiveness of the generated rules stops improving.

The three main classes of IE approaches differ significantly in the amount of used features. Knowledge-based approaches utilize comparably few features restricting them on semantic and syntactic information. Some statistical systems try to exploit all available information about text elements generating relatively big amount of features. Rule-based systems tend to rely heavily on linguistic features for rule generation.

6.4 Tagging Requirements and Learning Characteristics

Most approaches require training texts to be fully tagged, i.e. all items to extract must be marked (either embedded within the texts or in external documents). Full tagging of a large number of documents is a serious burden. Some systems alleviate this requirement by using *active learning* on partially tagged texts (the extended version of *Rapier*, *Whisk*, *Stalker* in Co-Testing setting, *Active HMMs*). None of these systems allows *incremental learning*, i.e. it is not possible to update the extraction model on-the-fly without requiring a full retraining. The knowledge-based approaches described in Sec. 5.5 utilize human review and interaction instead of postulating pretagged texts.

The general trend should go towards relaxing the input requirements on the training texts by incorporating better learning models. Statistical systems partially succeed in processing not fully consistent text corpora, while rule-based and knowledge-based systems rely on traditional elaborately prepared text resources.

[5] Used by *Crystal*, *(LP)*^{2}, *TIMES*, and *BIEN* for various word classes.
[6] *Rapier*, *SRV*, *TIMES*.
[7] *SRV*, *BWI*, *MEMM*.