**CHAPTER 3**

**SYSTEM DESIGN AND METHODOLOGY**

The purpose of this chapter is to provide the details of the system design and methodologies to develop the Shallow Parser. Section 3.1 contains the system design of the Shallow Parser. Section 3.2 and 3.3 contains the learning tasks and the training data. First, the learning tasks to be performed by the modules that make up the memory-based parser are presented. Next, the CGN and the conversion of its contents to a format suited for a memory-based learner are dealt with.

**3.1 System Design**

The concept of shallow parser proposed by Abney (1991) has no clearly defined meaning and is used sometimes in a very limited sense, referring only to tagging and chunking and sometimes in a broader sense, referring also to semantic tasks such as named-entity recognition. It can best be interpreted as a family of related tasks attempting to recover some syntactic-semantic information in a robust and deterministic way at the expense of ignoring detailed configurational syntactic information. In our approach to shallow parsing, we use memory-based learning as machine learning method. As mentioned in the introduction to this thesis, memory-based shallow parsing has already been successful when trained on English language corpora. Speciﬁcally, the research described in this section has generally been performed in combination with the Wall Street Journal corpus, which is a treebank of formally written English. In principle, however, memory-based shallow parsing is language-independent and therefore it should be possible to apply it to any language. For this reason, the research objective aims at adopting the memory-based shallow parsing method for constructing the Hindi memory-based parser.

The groundwork for the memory-based shallow parsing approach is laid by Daelemans (1996) who claims that all linguistic tasks can be reformulated as classiﬁcation problems and that, as a result of this, it is possible to train memory-based learners for these tasks. To support his claim, he ﬁrst shows that tasks in natural-language processing are context-sensitive mappings between representations and then argues that every linguistic problem can be described by one of two kinds of mappings: disambiguation and segmentation. Disambiguation mappings assign one of a predeﬁned set of categories to a context. This type of mappings includes part of speech tagging, where the correct word class, for example noun, verb, or adjective, for a word is determined given its form and its place in the sentence. Segmentation mappings decide whether, given a target and its context, a boundary is associated with this target and, if so, which type of boundary. A typical segmentation task is the detection of chunk boundaries. The cascaded Memory-Based Shallow Parser (Buchholz et al., 1999) implements the ideas described above. The parsing task is split up into a number of subtasks, each of which can be reformulated as either a disambiguation or a segmentation task. Starting with only the words forming a sentence, the available information is enriched by the results of each subtask. This way, the inputs to subtasks can be partly composed of information that is not directly accessible in the original data, but generated by other subtasks. For this reason, the memory-based classiﬁcation modules trained to perform the subtasks are applied in sequence, so that each module precedes those dependent on its results. The Memory-Based Shallow Parser has modules for part of speech tagging, chunking, PNP1 ﬁnding and grammatical-relation ﬁnding, in that order. Each of these tasks is individually introduced in the remainder of this section.

**3.1.1 Part of speech tagging**

Part of speech tagging is the process of associating each word in a sentence with its correct word class. This word class, also called the part of speech of the word, expresses how this word is used in the sentence. One word can be used in diﬀerent ways and therefore the part of speech for a word is not always the same. A part of speech tagger should be able to assign the correct part of speech to a word, basing its decision on the word itself and the words surrounding it in the sentence. Daelemans et al. (1996) introduce a memory-based approach to part of speech tagging. They show that a memory-based learner can be trained to perform part of speech tagging with good results. In addition, a tagger-generator is presented that automatically generates memory-based part of speech taggers, given tagged corpora as example data. In this thesis, part of speech tagging is not performed by a separate module. Rather, part of speech tagging and chunking are both performed by a single module, which is based on the techniques described by Daelemans et al. (1996).

**3.1.2 Chunking**

Chunking divides a sentence into non-recursive, non-overlapping constituents, referred to as chunks. Abney (1991) introduced the concept of a chunk, suggesting that dividing a sentence into chunks is a useful intermediate step towards full parsing. Nevertheless, chunking is a valuable process in its own right when the entire grammatical structure produced by a full parse is not required. Studies by Grishman (1995) and Appelt et al. (1993), for instance, indicate that the information obtained by a shallow parse is suﬃcient for information extraction to be performed. In terms of the two kinds of mappings proposed by Daelemans chunking is a segmentation task.

The chunking module of the Memory-Based Shallow Parser (Veenstra, 1999) uses a classiﬁcation scheme by Ramshaw and Marcus (1995): each word is assigned a chunk tag denoting whether the word is inside a chunk (I), inside a chunk, but not in the same chunk as the word directly preceding it (B), or outside any chunk (O). As there is more than one chunk type to distinguish, an additional chunk type symbol is added to the chunk tag. A chunk tag I-NP, for example, means the word is inside a noun phrase (NP). Buchholz (2002) additionally extends the chunk tag by attaching the part of speech tag, eﬀectively eliminating the need for a separate part of speech tagging step.

**3.1.3 PNP ﬁnding**

Due to the fact that only non-recursive constituents are identiﬁed in the chunking step, prepositional phrases (PP) will not be recognised as such. Rather, PPs are divided into a PP chunk containing only a single preposition, and one or more NP chunks. In order to recover the original prepositional phrase, the PNP ﬁnding module of the parser joins PP chunks and the NP chunks related to it in PNP chunks. PNP ﬁnding is implemented in a manner very similar to grammatical-relation ﬁnding.

**3.1.4 Grammatical-relation ﬁnding**

Finding grammatical relations is the ﬁnal step in the parsing process. During this step grammatical relations between verbal chunks and chunks of other types are identiﬁed. Relations to verbs include the most important relations to capture the meaning of a sentence, such as the subject and object relations. Buchholz (2002) describes the design of a memory-based grammatical-relation ﬁnder. The technique described assumes that grammatical relations hold between the head words of chunks. The head word of a chunk is the most prominent word in that chunk.

Grammatical-relation ﬁnding is formulated as a disambiguation task: given the head word of a verbal chunk and the head word of another chunk it is predicted whether a grammatical relation holds between the two words and, if so, which type it has. Before relation ﬁnding can take place, the parts of speech and the chunks for the words in the sentence should have been determined. This information is used in the description of the instances fed to the relation ﬁnder. Apart from part of speech and chunk type, other features that were found to be good predictors for grammatical relations include spatial features, such as the number of intervening verb chunks and commas.

In the previous sections, it has been recognised that there is a need for techniques that can accelerate the development of natural-language processing applications for Hindi. Memory-based shallow parsing has been put forward as a technique satisfying this need. Although, for practical reasons, the emphasis of memory-based shallow parsing research has traditionally been on English, the approach is essentially language-independent and consequently does not exclude Hindi.

**3.2 Learning Task**

As explained in section 3.1, the parsing process is split up into 1) a combined part of speech tagging/chunking step, 2) PNP ﬁnding, and 3) grammatical-relation ﬁnding. Analogously to this subdivision, the actual memory-based parser has been designed as a modular system, consisting of separate memory-based classiﬁers for each distinct subtask. These classiﬁer modules can be approached from two diﬀerent angles. On the one hand, they can be dealt with as independent units that can be described, trained and optimised individually. Indeed, all the parsing steps described in this section have individually been the central topic of one or more publications. These publications proposed a translation of the tasks to the memory-based learning domain and gave an overview of its performance when tested in isolation.

On the other hand, the modules are interdependent components of a global parsing framework, as is illustrated in ﬁgure 3.1. The grammatical-relation ﬁnder, for instance, uses results of both the chunker and the PNP ﬁnder modules. Without these two modules, the grammatical-relation ﬁnder can only be applied in a test environment where the necessary information is extracted from a corpus. This interdependency, then, has two important consequences. First, actual parsing of a sentence requires that all modules have been implemented and joined together in the parsing framework. Secondly, the accuracy of the output of a module not only depends on its own classiﬁcation performance, but also on the accuracy of its input data, which are partly generated by other modules.

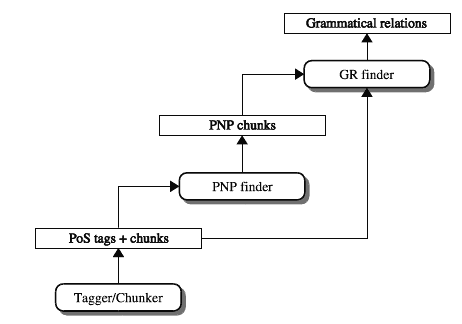


Figure 3.1: Architecture of the memory-based shallow parsing framework. The rounded rectangles represent the three parser modules; the normal rectangles and the arrows depict the information ﬂow between the modules.

**3.3 Data**

The previous section described the instance formats for the three learning tasks that make up for the memory-based shallow parsing framework. In this section, the extraction of the information required for generating these instances is presented. First, the format of the syntactic annotations of the EMILLE Corpus is reviewed. Then, the conversion of EMILLE annotations to the chunks and grammatical relations required for the learning tasks is explained.

**3.3.1 EMILLE Corpus**

The EMILLE Corpus has been constructed as part of a collaborative venture between [the EMILLE project](http://www.emille.lancs.ac.uk/) (Enabling Minority Language Engineering), Lancaster University, UK, and the Central Institute of Indian Languages (CIIL), Mysore, India. EMILLE is distributed by the European Language Resources Association.

The corpus consists of three components: monolingual, parallel and annotated corpora. There are fourteen monolingual corpora, including both written and (for some languages) spoken data for fourteen South Asian languages: Assamese, Bengali, Gujarati, Hindi, Kannada, Kashmiri, Malayalam, Marathi, Oriya, Punjabi, Sinhala, Tamil, Telegu and Urdu. The EMILLE monolingual corpora contain approximately 92,799,000 words (including 2,627,000 words of transcribed spoken data for Bengali, Gujarati, Hindi, Punjabi and Urdu). The parallel corpus consists of 200,000 words of text in English and its accompanying translations in Hindi, Bengali, Punjabi, Gujarati and Urdu. The annotated component includes the Urdu monolingual and parallel corpora annotated for parts-of-speech, together with twenty written Hindi corpus files annotated to show the nature of demonstrative use. The corpus is marked up using CES-compliant SGML, and encoded using Unicode.

The annotation scheme for Hindi was designed on the basis of the blueprint provided by Botley’s (2000) scheme devised for English demonstrative anaphors. A short description of Botley’s scheme is provided here in order to facilitate an understanding of the annotation scheme for Hindi demonstrative anaphors. The analysis of Botley’s three data sets was carried out using a set of linguistic labels which are typically termed ‘tags’ in corpus linguistics literature.

A two-step process was followed in Botley’s study:

1. Each demonstrative case in the three corpora was identified and classified, and distribution statistics were obtained.
2. The demonstrative types identified according to (1) were represented in terms of distinctive features found to be present or absent in them.

The demonstrative pronouns are understood in terms of an unordered paradigmatic set of five distinctive features, in Botley’s (2000) study:

R = Recoverability of antecedent

D = Direction of reference

P = Phoric Type

S= Syntactic function

A= Antecedent type

**3.3.2 Data Conversion**

The syntactically annotated EMILLE Corpus data described in the previous section form the basis for the experiments that are reported on in this thesis. The parsed sentences are used as training and test data for all learning tasks in the parser. However, while these learning tasks are all formulated in terms of chunks and relations between chunks, the EMILLE data are fully parsed. Thus, in order to create machine-learning instances from the corpus data, the chunks and the relations between them have to be extracted from the annotation structures.

**Heads and chunks**

The ﬁrst step in converting the CGN data to the intermediate format is the extraction of chunks from the dependency structures. The chunks in a sentence can be found by identifying their head words. From these head words, the other words in the chunks can be inferred by applying the following deﬁnition from Abney (1991): the root node R of a chunk with head h is the highest node in the parse tree T that has h as its head. Given this root node, the syntactic structure of a chunk is the largest continuous subgraph of T rooted in R and not containing the root of any other chunk. The words of a chunk, then, correspond to the leaf nodes of its chunk structure and the chunk label is deﬁned as the phrasal type of its root node. With Abney’s deﬁnition, ﬁnding chunks come down to identifying the head words in a sentence and determining the chunk structures they give rise to.

**Grammatical relations**

Having found the heads and chunks, grammatical relations can be extracted from the corpus data. In contrast with the Penn Treebank format, the corpus annotation explicitly represents these relations. A dependency link between a daughter node and its mother domain corresponds to a grammatical relation between the head word of the daughter node and that of the mother domain. For the purpose of converting the EMILLE data to intermediate format, these correspondences have to be recorded explicitly.