Text Chunker for Punjabi

Ubeeka Jain†* and Jasbir Kaur†

†R.I.E.I.T., Railmajra, Punjab, India


Abstract

Parsing is the process of assigning a parse tree to the sentence. There are many problems related to the process of full parsing. Shallow parsing or chunking is the alternative for full parsing. In chunking the phrases of the sentences are chunked together. chunking is more efficient and robust as it takes less time and always gives a solution. It is often deterministic as it gives only one solution to a problem. Chunkers are used in a large no. of NLP applications. Such as information extraction, named entity recognition, spell checkers, search etc. Chunkers are relatively difficult to build for Indian languages as there arise many problems during the system development. Chunkers identify the noun or verb etc chunks. Chunks are the non-overlapping regions. In this work, first standardized text chunker for Punjabi language is built and the greedy based algorithm is used for the machine learning and training of data set.

Keywords: Natural language Processing (NLP), Part of Speech Tagger (POS), Punjabi chunker

1. Introduction

In NLP Computers are used to understand and manipulate text and speech to do some useful work NLP is the branch of Computer science mainly dealing with developing of systems by which computers can interact with human using natural language . NLP includes various computational and analyzing processes which enable machine to understand the language. Punjabi is an Indo-Aryan language. It is the 10th most spoken language in the world and native language of about 131 million people. Most of the Punjabi speaking people live in Punjab region of Pakistan and India. It is also spoken in Himachal Pradesh, Haryana and Delhi and many countries in abroad. Punjabi is written in two different scripts called Gurmukhi and Shahmukhi.

Some of the applications for NLP are Part of Speech tagging (POS), Question Answering system, Name Entity Recognition (NER), and Multiple Word Expression (MWE) etc. which are used in machine translation.

Chunking: chunking is the process of dividing the sentence into chunks. Chunks are the non-overlapping regions in a sentence. Chunks are correlated group of words (Abney et al., 1991).

The phrase chunker divides the sentence into noun phrases or verb phrases. These phrases are grouped together i.e., all the verbs occurring in a sentence are chunked in a single chunk and all the noun phrases are grouped in another single chunk. There also exist adjective phrases and noun adverb phrases. (Anil K Singh et al., 2008)

There are many levels of language analysis. These are shown in the following figure. The parsing phase lies in the syntax level of language analysis. Parsing is the process of generation of parse tree for a sentence.

Chinking is the alternative to parsing. There exists no complete grammar for any language. Ambiguity exists for many sentences. Ambiguity is the generation of more than one parse tree for one sentence. Full parsing takes a reasonable time for large amount of data. Chunking is more efficient and robust as it takes less time and always gives a solution. It is often deterministic as it gives only one solution to a problem. Context is Small and local. it can be applied to very large text resources i.e. web. (Kudo et al 2001)

The output of the chunker consists of series of non-overlapping regions that are also non recursive and do not contain each other. Thus the output of chunker is different from the parsing and it is easier as compared to parsing.

Rest of the paper is organized as follows the section 2 describes the applications of chunker. Section 3 contains the tagset for POS tagging and chunking. Section 4 briefs about corpus development. Section 5 consists of overview of framework. Section 6 briefs about system design and implementation. Section 7 contains testing and results. Section 8 concludes the conclusion.
2. Potential Applications

Chunkers are used as a resource component for many NLP applications.

A. Information extraction: the chunker divides the sentence into chunks of interrelated data. Noun phrase and verb phrase are chunked and can be used in information extraction systems. IE focuses on discovering names of people and events they participate in, from a document.

B. Question Answering system: the complete chunk can be used as the answer of the question asked. Question-answering provides the user with either just the text of the answer itself or answer-providing passages.

C. Spell Checkers: checks the wrongly typed words within the sentence.

D. Named entity identification: in this system the main aim is to identify the particular words in the document. Such as people, places and other nouns in the sentence.

E. Search: searching of a particular noun or verb can be done. As the sentence is chunked in pieces, search becomes an easy task and the whole chunk can be represented as the search result.

F. Machine translation: machine translation is the process of translating one language into another language. Chunking is useful in this task as the chunks are converted into another language.

3. Tagset for Parts of Speech Tagging

POS tag set used in development of this chunker is the standard tagset given by TDIL for Punjabi language. There are 35 standard tags for Punjabi (TDIL).

Table 1 Tagset for Parts of Speech Tagging

<table>
<thead>
<tr>
<th>No.</th>
<th>Tag</th>
<th>Tag Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>N_NN</td>
<td>Common Noun</td>
</tr>
<tr>
<td>2</td>
<td>N_NNP</td>
<td>Proper Noun</td>
</tr>
<tr>
<td>3</td>
<td>N_NST</td>
<td>Noun loc</td>
</tr>
<tr>
<td>4</td>
<td>PR_PRP</td>
<td>Personal Pronoun</td>
</tr>
<tr>
<td>5</td>
<td>PR_PRF</td>
<td>Reflexive Pronoun</td>
</tr>
<tr>
<td>6</td>
<td>PR_PRL</td>
<td>Relative Pronoun</td>
</tr>
<tr>
<td>7</td>
<td>PR_PRC</td>
<td>Reciprocal Pronoun</td>
</tr>
<tr>
<td>8</td>
<td>PR_PRQ</td>
<td>Wh-word Pronoun</td>
</tr>
<tr>
<td>9</td>
<td>PR_PRI</td>
<td>Indefinite</td>
</tr>
<tr>
<td>10</td>
<td>DM_DMD</td>
<td>Deictic Demonstrative</td>
</tr>
<tr>
<td>11</td>
<td>DM_DMR</td>
<td>Relative Demonstrative</td>
</tr>
<tr>
<td>12</td>
<td>DM_DMQ</td>
<td>Wh-word Demonstrative</td>
</tr>
<tr>
<td>13</td>
<td>DM_DMI</td>
<td>Indefinite Demonstrative</td>
</tr>
<tr>
<td>14</td>
<td>V_VM</td>
<td>Main Verb</td>
</tr>
</tbody>
</table>

For Chunking, mainly seven tags are used. This is based on the grammatical or the syntactical category. The chunks are represented in square brackets and the right hand side contains the head naming the chunk.

Table 2 Tagset for Chunking

<table>
<thead>
<tr>
<th>No.</th>
<th>Chunk</th>
<th>Chunk Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>_NP</td>
<td>Noun chunk</td>
</tr>
<tr>
<td>2</td>
<td>_CCP</td>
<td>Conjunction chunk</td>
</tr>
<tr>
<td>3</td>
<td>_VGF</td>
<td>Verb chunk</td>
</tr>
<tr>
<td>4</td>
<td>_RBP</td>
<td>Adverb chunk</td>
</tr>
<tr>
<td>5</td>
<td>_JJP</td>
<td>Adjective chunk</td>
</tr>
<tr>
<td>6</td>
<td>_VGINF</td>
<td>Verb infinite</td>
</tr>
<tr>
<td>7</td>
<td>_BLK</td>
<td>Bulk phrase</td>
</tr>
</tbody>
</table>

The guidelines mentioned in tagset given by the TDIL are followed for chunking. Seven chunks are used. First is the noun phrase chunk. It is given the tag _NP and the head is noun. Examples of noun chunk are:

- [[ਨਾਂਕਰਿਆਂ

  _NP]]]
- [[ਨਾਂਕਰਿਆਂ

  _NP]]]

The conjunction chunk is tagged as _CCP. Conjunctions are the words used to join phrases, words, clauses. The example is:

- [[ਨਾਂਕਰਿਆਂ

  _CCP]]]
- [[ਨਾਂਕਰਿਆਂ

  _CCP]]]
Verb chunks are classified as verb chunk denoted by \_VGF and infinite verb chunk denoted by \_VGINF. The examples are:

\[
[\text{[[\text{V_VM} \text{ V_VM} \text{ VNF}]]} \_VGF] \\
[\text{[[\text{N_NN} \text{ VM} \text{ VM} \text{ VF}]]} \_VGF] \\
[\text{[[\text{V_VM} \text{ VNF}]]} \_VGINF] \\
[\text{[[\text{V_VM} \text{ VNF}]]} \_VGINF]
\]

Adverb chunks are denoted by \_RBP. These are tagged in accordance with the tagset of POS. the example is:

\[
[\text{[[\text{ਆਲ}} \text{ CC} \text{ CCS} \text{ RB}]]} \_RBP] \\
[\text{[[\text{V_VM} \text{ VNF} \text{ RB}]]} \_RBP]
\]

Adjective chunks are given the tag \_JJP. This includes all the adjective chunks. The example is:

\[
[\text{[[\text{ਪ੍ਰ}} \text{ PSP} \text{ JJ}]]} \_JJP] \\
[\text{[[\text{JJ} \text{ PSP} \text{ JJ}]]} \_JJP]
\]

In Bulk phrase all the miscellaneous data is given the tag \_BLK. The example is:

\[
[\text{[[\text{V_VM} \text{ VF} \text{ RD_PUNC}]]} \_BLK] \\
[\text{[[\text{V_VM} \text{ VNF} \text{ PSP} \text{ N_NN} \text{ PSP}]]} \_BLK]
\]

4. Corpus Development

Corpus is developed for training and testing of the system. The training data contains one thousand sentences of Punjabi which are tagged using the already developed HMM based POS tagger for Punjabi and then manually chunking of the corpus. This chunked corpus is given for training of the system using machine learning tools. The data is collected from various sources like online news, stories, newspaper articles etc. The sample of training data is as follows:

\[
[\text{[[\text{ਪ੍ਰ}} \text{ N_NN}]]} \_NP ] [\text{[[\text{V_VM} \text{ VF}]]} \_VGF] \\
[\text{[[\text{ਪ੍ਰ}} \text{ PSP}]]} \_BLK ] [\text{[[\text{ਪ੍ਰ}} \text{ P_NP} \text{ V_VM} \text{ VF} \text{ N_NN} \text{ PSP}]]} \_BLK] \\
[\text{[[\text{V_VM} \text{ VNF} \text{ RD_PUNC}]]} \_BLK]
\]

The design of the chunker is as described in the flowchart. A sketchy idea is described below that how the input text is processed and the output is given in the form of chunked data.

For the chunking of the raw text, the input text is given to the chunker. Normalization of the text is done. In normalization unwanted chars from the input are removed and some formatting is added for further processing by the algorithm. If the input text is not tagged then POS tagging of the text is done using the already built HMM based POS tagger. The POS tagger tags the whole text into 35 standard tags. Then the tokenization of the sentences is done. The words from the tagged data are removed and the POS tag pattern is created. We concern only about the pattern of the tags for further processing. Then the combination with all the chunk tags is created. It is analyzed that which tag pattern correspond to which chunk. We have used seven tags in the system. Using the training data the most frequent chunk tag pattern is found and the input is given that chunk name.

6. System Design and Implementation

The chunking system is divided into two portions. First is training and the second is testing.

Training Process: first of all we have collected the training data. The training data is raw text collected from various sources which is first of all POS tagged. The chunks are identified and tagged in POS data. This training data is saved in a separate file. For the training process of the system machine learning approach is used. the words are removed and only the tag pattern is analyzed. The system checks the pattern and the chunk associated with it and makes a hash table for every pattern. Every tag pattern and the related chunk in the training data is saved in the directory along with the frequency of the occurrence of the pattern. The training file is saved in the memory as binary file.
Text Chunker for Punjabi

**Testing Process:** during the testing process greedy based algorithm is used. When the POS tagged data is input to the system then the already trained system takes the POS tag pattern and checks the frequency of the pattern in the directory. After frequency analyses of the pattern in the directory the most frequent chunk is found and the output as the chunked data is given. The system is implemented in Microsoft visual c# for POS tagging of the data we have used the HMM based POS tagger already developed by Punjabi university.

**Sample input and output:** This section provides some sample Punjabi sentences given as the input to the system and output as chunked data is given by the system.

**Input 1:**

```
ਊਡ ਲਾਟਤੀ ਜੋ ਹਿਸਾਬ ਵੇਲੀ ਕਰ ਪ੍ਰਤੀਕਾਂ ਪੁਰਸ਼ੁਆਲਾਂ ਦਿਕਦਾ ਲਗਾਈ। ਜਦੋਂ ਜੋ ਹਿਸਾਬ ਤੇਵੇ ਵੇਲੀ ਕਰ ਪ੍ਰਤੀਕਾਂ ਪੁਰਸ਼ੁਆਲਾਂ ਦਿਕਦਾ ਲਗਾਈ।
```

**Input 2/output 1 (POS tagging):**

```
ਊਡ ਲਾਟਤੀ ਜੋ ਹਿਸਾਬ ਵੇਲੀ ਕਰ ਪ੍ਰਤੀਕਾਂ ਪੁਰਸ਼ੁਆਲਾਂ ਦਿਕਦਾ ਲਗਾਈ। ਜਦੋਂ ਜੋ ਹਿਸਾਬ ਤੇਵੇ ਵੇਲੀ ਕਰ ਪ੍ਰਤੀਕਾਂ ਪੁਰਸ਼ੁਆਲਾਂ ਦਿਕਦਾ ਲਗਾਈ।
```

**Final output:**

```
[[ਊਡ ਲਾਟਤੀ ਜੋ ਹਿਸਾਬ ਵੇਲੀ ਕਰ ਪ੍ਰਤੀਕਾਂ ਪੁਰਸ਼ੁਆਲਾਂ ਦਿਕਦਾ ਲਗਾਈ। ਜਦੋਂ ਜੋ ਹਿਸਾਬ ਤੇਵੇ ਵੇਲੀ ਕਰ ਪ੍ਰਤੀਕਾਂ ਪੁਰਸ਼ੁਆਲਾਂ ਦਿਕਦਾ ਲਗਾਈ।]]
```

The input given to the chunker is either raw data on which we done POS tagging using HMM based POS tagger or already POS tagged data is input to the system.

### 7. Testing and Result

After training the system with chunked data we perform the testing of the system with raw data. The various formulas used in result are as follows:

- **Precision:** \( P = \frac{\text{No. of correct answers}}{\text{No. of answers given}} \)
- **Recall:** \( R = \frac{\text{No. of correct answers given by the system}}{\text{Total No. of answers}} \)
- **F-measure:** F-measure is defined as balances of Recall and Precision by using a parameter \( \beta \)

\[
F = \frac{(1+\beta^2)FP}{(\beta^2P + F)}
\]

\( \beta \) is weighted as \( \beta=1 \)

When \( \beta=1 \), F-measure is called F1-measure

\[
F1 = \frac{2RP}{P+R}
\]

Following results were obtained while testing the raw corpus within the system. The raw corpus used for testing was in Unicode. For training the system, ie for in the training phase, the chunker was trained with using about 1000 sentences. Increasing the accuracy of the system can increase this further to any extent there. 1000 is total no. of sentences for testing and 750 is correct answers given by system:

\[
P = \frac{750}{800} = .93 \approx 93%
R = \frac{750}{1000} = .75 \approx 75%
F-measure = \frac{2(75+93)}{75+93} = 83%
\]

Keeping into mind the fact that this is the first standard chunker, these results are considered as good.

### Comparison with existing systems

With best of our knowledge there exist no chunker available for Punjabi which has used standardized POS tagset given by TDIL. There exist chunkers for other Indian languages. We compare our system with the existing systems. In 1995, Ramshaw and Marcus obtained a precision of 91.8% and a recall of 92.3% for base np chunks when trained on 200000 words(A. Ramshaw, P. Marcus et al, 1995). Zhou in 2000 used the HMM method and achieved the recall and precision of 92.25 and 91.99 respectively(Zhou et al, 2000). Jisha P Jayan and Rajeev R Got the results for malayalam chunker- Equal : 184/200 (92.00%) Different : 16/200 (8.00%) the system gives about 92% of accuracy for Malayalam.

95.82% of the accuracy is obtained by Dhanalakshmi for tamil chunker(Dhanalakshmi, 2009). 92.63% for chunk boundary identification task and 91.70% for
the composite task of chunk labeling with a recall of 100% is obtained by Akshey singh, sushma for hindi chunker( Akshay Singh et al,2005). The precision and recall rates of 96.12% and 98.03% are obtained by Dipanjan Das, Monojit Choudhury for Bengali language chunker(Dipanjan et al).

Conclusion

This paper presents the implementation and results for the chunking system of Punjabi. This system performs the chunking of Punjabi text into seven chunks. To the best of our knowledge it is the first chunker for Punjabi language based on standardized POS tagset given by TDIL. The development of Punjabi language is in first phase. This effort will reduce the gap of development of resources. Chunker is used as an essential tool for the further development of resources. This work will definitely motivate the future researchers for development in the area of Punjabi.

References


Anil Kumar Singh, (2008) Language Technologies Research Centre, IIIT, Hyderabad India, NLP for Less Privileged Languages: Where do we come from? Where are we going? In IJCNLP Workshop on NLP


Unified Parts of Speech (POS) Standard in Indian Languages - Draft Standard –Version 1.0 Department of Information Technology Ministry Communications & Information Technology Govt. of India


Jisha P Jayan, Rajeev R R Parts Of Speech Tagger and Chunker for Malayalam – Statistical Approach Computer Engineering and Intelligent Systems ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)


Dipanjan Das, Monojit Choudhury, An Affinity Based Greedy Approach towards Chunking for Indian Languages Department of Computer Science and Engineering Indian Institute of Technology, Kharagpur