Akshita Jha – 201225081
Prathyusha Danda - 201225156

STATISTICAL MORPHANALYZER FOR HINDI

Introduction

Morphological analysis for Indian Languages (ILs) is defined as the analysis of a word in terms of its lemma (L), gender (G), number (N), person (P), case (C), vibhakti3 , tense, aspect and modality. A tool which predicts Morph Analysis of a word is called a Morph Analyzer (MA). Statistical Morph Analyzer (SMA) is an MA which uses machine learning to predict the morph information Using the training data and the feature-set, statistical models are formed. These models help to predict the morph-analysis of the test data. This works for all words, including out of vocabulary (OOV) words. SMA is language independent. We chose Indian Languages for our study and built an SMA which is targeted for different ILs. Indian languages are lexically and grammatically similar. Lexical borrowing occurs between languages.

Approach

Feature Set

The feature-set was chosen specifically to suit the Indian Languages. The following are the features used:

(i) Suffixes : Indian languages show inflectional morphology. The inflectional morphemes carry the G,N,P and C of a word. These morphemes generally occur in the form of suffixes. Hence, to capture the inflectional behaviour of ILs we considered the suffixes as a feature for the ML task. We considered suffixes whose length was maximum 7 characters.

(ii) Previous morph tags and next morph tags : Agreement is an important characteristic of ILs. Through agreement, GNPC of a token may percolate to the other tokens. An example to this is, if the subject (noun) is masculine, then the verb form should also be masculine. To capture agreement, we considered features which carried the GNPC of the neighbouring words. Previous morph tags feature captures predicted morph tag of previous 3 tokens. Next morph tags feature captures the set of morph tags of the next token, if found in the training corpus.

(iii) Word Forms: ILs are morphologically rich languages. Words carry rich information regarding GNPC. To capture this characteristic we considered three features relating to word forms. word present captures the word form of the present token. Word previous captures the word form of the previous token. Word next captures the word form of the next token.

(iv) Part of Speech (POS) : POS is one of the of the fundamental ML feature of any NLP task. Based on the POS of the word, the set of possible inflections can be found. For example, verbs have a set of inflections and nouns have another set. To capture such information we included POS in the feature-set.

(v) Other features : Features such as length of the token and character types in the token (eg. Numbers, alphabets and so on) have also been considered.

Choosing Class Labels
For the ML task, the class-labels for G, N, P, C were chosen from the training data itself. For lemma, the class-labels were formed based on the edit-distance8 operations required to convert the given token to its lemma. This idea was inspired by Chrupała (2006), who introduced the concept of edit-operations for lemmatization.

The Algorithm is explained using an example. Consider the token 'crying'. The lemma for 'crying' is 'cry'.

Step 1: The token and its lemma are reversed. crying becomes gniyrc and cry becomes yrc.

Step 2: Note the edit operations required to convert reversed token to the reversed lemma. To convert 'gniyrc' to 'yrc' we need to delete the characters at the 1st, 2nd and 3rd indices. Hence the edit operations would be [d 1, d 2, d 3], where ’d’ represents delete operation.

Step 3: The set of edit operations would form the class-label. [d 1, d 2, d 3] would be the class-label and would be added to the set of class-labels.

Note: [i x 1] -> insert 'x' at index 1
[s x y 1] -> substitute 'x' with 'y' at index 1

Language	**Training Corpus (tokens)**	**Class Labels Count**	**Percentage**
Hindi	368,060	410	0.12 %

Dictionary Population

Dictionary Population for suffixes:
Features have to be extracted, all possible distinct suffixes are extracted from the training corpus. Using these distinct suffixes, dict is populated with the suffix as the key and the count as the value, with count getting incremented for every suffix entry.

Dictionary Population for Previous Morph Tags:
The total number morph tags (class labels) are made a note off. Let the number of morph tags be m. Suppose the number of previous tokens whose predicted morph tags are to be considered as features is p. There will be p sets of m morph tags, as key entries in dict, and the count represents the value for each key entry, with count getting incremented for every single entry. Each of this n4 th set is used for representing the morph tag of nth previous word. Hence the number of entries in dict would be m*p.

Dictionary Population for Next Morph Tag
The dict is populated with m entries, with the key as the morph tag and the value as count, with count getting incremented for every entry. .

Dictionary Population for Word Forms
Suppose the total number of distinct words in the training corpus is w. Three sets of the w are entered as keys and and the count represents the value for each key entry, with count getting incremented for every single entry. These three sets of entries represent the word previous, word

present and word next features respectively.

Dictionary Population for POS

All the POS tags are added to dict, with the key as the POS tag and the value as count, with count getting incremented for every single entry. Refer Figure 3.1. In this figure only 4 POS tags are added, since it is just for explanation. The number of POS tags depends on the POS tagset which is used.

Dictionary Population for Other Features

For the length of the token a single entry with ”length” as the key and the count as the value, is made. count is incremented once. For the character types feature two entries are made in the dict. The first is ”number” as the key and the second is ”alphabet” as the key, and count represents the values, with count getting incremented for each entry. The first entry depicts the presence of numbers and the second entry depicts the presence of alphabets.

The keys in the dictionary represent the features and the values in the dictionary are numbers to denote each of the corresponding feature.