IndiLem@FIRE-MET-2014: An Unsupervised Lemmatizer for Indian Languages

Abhisek Chakrabarty

Indian Statistical Institute, Kolkata

December 6, 2014
Task of a lemmatizer and its need.
Contents

- Task of a lemmatizer and its need.
- The proposed lemmatization Approach.
Contents

- Task of a lemmatizer and its need.
- The proposed lemmatization Approach.
- Results and error analysis.
What is lemmatization and why is it needed?

Lemmatization is a process that returns the base or dictionary form of a word in context, which is known as the lemma of that word in that context. A lemma of a word in a context is required to retrieve the meaning of that word in that context.

For example, ‘I retrieved that document.’ Here the lemma of retrieved is retrieve. If we cannot map retrieved to retrieve, the meaning of retrieved is not accessible.
What is lemmatization and why is it needed?

Lemmatization is a process that returns the base or dictionary form of a word in context, which is known as the lemma of that word in that context.

For example, 'I retrieved that document.' Here the lemma of retrieved is retrieve. If we cannot map retrieved to retrieve, the meaning of retrieved is not accessible.
What is lemmatization and why is it needed?

Lemmatization is a process that returns the base or dictionary form of a word in context, which is known as the lemma of that word in that context.

A lemma of a word in a context is required to retrieve the meaning of that word in that context.
What is lemmatization and why is it needed?

Lemmatization is a process that returns the base or dictionary form of a word in context, which is known as the lemma of that word in that context.

A lemma of a word in a context is required to retrieve the meaning of that word in that context.

For example, ‘I retrieved that document.’

Here the lemma of retrieved is retrieve.

If we cannot map retrieved to retrieve, the meaning of retrieved is not accessible.
What is lemmatization and why is it needed?

Lemmatization is a process that returns the base or dictionary form of a word in context, which is known as the lemma of that word in that context.

A lemma of a word in a context is required to retrieve the meaning of that word in that context.

For example, ‘I retrieved that document.’

Here the lemma of retrieved is retrieve.

If we cannot map retrieved to retrieve, the meaning of retrieved is not accessible.
Need of lemmatization for Indian languages

Major Indian languages (Hindi, Bengali, Gujarati etc.) are morphologically very rich and suffixing in nature.
Need of lemmatization for Indian languages

Major Indian languages (Hindi, Bengali, Gujrati etc.) are morphologically very rich and suffixing in nature.

Knowledge resources (dictionary, WordNet) for those languages usually store root words with their morphological and semantic descriptions.
Need of lemmatization for Indian languages

Major Indian languages (Hindi, Bengali, Gujrati etc.) are morphologically very rich and suffixing in nature.

Knowledge resources (dictionary, WordNet) for those languages usually store root words with their morphological and semantic descriptions.

Very often we face several inflected word forms in raw texts like stories, newspapers, poems etc.
Need of lemmatization for Indian languages

Major Indian languages (Hindi, Bengali, Gujrati etc.) are morphologically very rich and suffixing in nature.

Knowledge resources (dictionary, WordNet) for those languages usually store root words with their morphological and semantic descriptions.

Very often we face several inflected word forms in raw texts like stories, newspapers, poems etc.

To obtain the meaning and morphological properties of them, we have to determine the appropriate root words which is the task of a lemmatizer.
Need of lemmatization for Indian languages

Major Indian languages (Hindi, Bengali, Gujrati etc.) are morphologically very rich and suffixing in nature.

Knowledge resources (dictionary, WordNet) for those languages usually store root words with their morphological and semantic descriptions.

Very often we face several inflected word forms in raw texts like stories, newspapers, poems etc.

To obtain the meaning and morphological properties of them, we have to determine the appropriate root words which is the task of a lemmatizer.

So lemmatization is necessary for building up many NLP tools (WSD systems, translation systems etc.) for Indian languages.
Different from stemmer

A stemmer operates on a single word without knowledge of the context. Usually a stemmer returns the common portion of the variant word forms and the stem may be an invalid word. But on varying contexts, the lemma of a particular word may be different and the lemma must be a valid word of the language. For all of the words retrieved, retrieving, a stemmer may return retrieve as the stem as it is the common portion of all the inflected forms. But a lemmatizer should return retrieve.
Different from stemmer

A stemmer operates on a single word without knowledge of the context.

Usually a stemmer returns the common portion of the variant word forms and the stem may be an invalid word.

But on varying contexts, the lemma of a particular word may be different and the lemma must be a valid word of the language.
Different from stemmer

A stemmer operates on a single word without knowledge of the context.

Usually a stemmer returns the common portion of the variant word forms and the stem may be an invalid word.

But on varying contexts, the lemma of a particular word may be different and the lemma must be a valid word of the language.

For all of the words retrieved, retrieval, retrieving, a stemmer may return retriev as the stem as it is the common portion of all the inflected forms.
Different from stemmer

A stemmer operates on a single word without knowledge of the context.

Usually a stemmer returns the common portion of the variant word forms and the stem may be an invalid word.

But on varying contexts, the lemma of a particular word may be different and the lemma must be a valid word of the language.

For all of the words retrieved, retrieval, retrieving, a stemmer may return retriev as the stem as it is the common portion of all the inflected forms.

But a lemmatizer should return retrieve.
Proposed Lemmatization Algorithm

Our lemmatization algorithm requires a dictionary or WordNet for collecting the root words of a language. At first, the root words are stored in a trie structure. Each node in the trie corresponds to an unicode character of the language. The nodes that end with the final character of a root word are marked as "final" nodes.
Proposed Lemmatization Algorithm

Our lemmatization algorithm requires a dictionary or WordNet for collecting the root words of a language.
Proposed Lemmatization Algorithm

Our lemmatization algorithm requires a dictionary or WordNet for collecting the root words of a language.

At first, the root words are stored in a trie structure.
Proposed Lemmatization Algorithm

Our lemmatization algorithm requires a dictionary or WordNet for collecting the root words of a language.

At first, the root words are stored in a trie structure.

Each node in the trie corresponds to an unicode character of the language.
Proposed Lemmatization Algorithm

Our lemmatization algorithm requires a dictionary or WordNet for collecting the root words of a language.

At first, the root words are stored in a trie structure.

Each node in the trie corresponds to an unicode character of the language.

The nodes that end with the final character of a root word are marked as "final" nodes.
Proposed Lemmatization Algorithm

Our lemmatization algorithm requires a dictionary or WordNet for collecting the root words of a language.

At first, the root words are stored in a trie structure.

Each node in the trie corresponds to an unicode character of the language.

The nodes that end with the final character of a root word are marked as "final" nodes.
Examples

- ‘অংশ’/‘angsh’ = ‘অ’/‘a’
  + ‘ং’/‘ng’ + ‘শ’/‘sh’
Examples

- ‘অংশ’/‘angsh’ = ‘অ’/‘a’
  + ‘ং’/‘ng’ + ‘শ’/‘sh’

- ‘অংশ’/‘angshu’ = ‘অংশ’/‘angsh’ + ‘ু’/‘u’
Examples

- ‘অংশ’/‘angsh’ = ‘অ’/‘a’
  + ‘ং’/‘ng’ + ‘শ’/‘sh’

- ‘অংশু’/‘angshu’ =
  ‘অংশ’/‘angsh’ + ‘ু’/‘u’

- ‘অংশুক’/‘angshuk’ =
  ‘অংশু’/‘angshu’ + ‘ক’/‘k’
Examples

- ‘অংশ’/‘angsh’ = ‘অ’/‘a’ + ‘ং’/‘ng’ + ‘শ’/‘sh’
- ‘অংশ’/‘angshu’ =
  ‘অংশ’/‘angsh’ + ‘ু’/‘u’
- ‘অংশুক’/‘angshuk’ =
  ‘অংশ’/‘angshu’ + ‘ক’/‘k’
- ‘অংশধর’/‘angshudhar’ =
  ‘অংশ’/‘angshu’ +
  ‘ধ’/‘dha’ + ‘র’/‘r’
Examples

- ‘অংশ’/‘angsh’ = ‘অ’/‘a’
  + ‘ং’/‘ng’ + ‘শ’/‘sh’

- ‘অংশু’/‘angshu’ =
  ‘অংশ’/‘angsh’ + ‘ু’/‘u’

- ‘অংশুক’/‘angshuk’ =
  ‘অংশ’/‘angsh’ + ‘ক’/‘k’

- ‘অংশুধর’/‘angshudhar’ =
  ‘অংশ’/‘angsh’ +
  ‘ধ’/‘dha’ + ‘র’/‘r’

- ‘অংশগত’/‘angshgata’ =
  ‘অংশ’/‘angsh’ + ‘গ’/‘ga’
  + ‘ত’/‘ta’
Searching strategy in the trie
Searching strategy in the trie

To find the lemma of a surface word, the trie is navigated starting from the initial node.
Searching strategy in the trie

To find the lemma of a surface word, the trie is navigated starting from the initial node.

Navigation ends when either the word is completely found in the trie or after some portion of the word there is no path present in the trie to navigate.
Searching strategy in the trie

To find the lemma of a surface word, the trie is navigated starting from the initial node.

Navigation ends when either the word is completely found in the trie or after some portion of the word there is no path present in the trie to navigate.

While navigating, some situations may occur, depending on which we are taking decision to determine the lemma.
Searching strategy in the trie (continued)

If the surface word is itself a root word, then we will reach to a final node. If the surface word is not a root word, then the trie is navigated upto that node where the surface word completely ends or there is no path to navigate. We call this node as the end node.
If the surface word is itself a root word, then we will reach to a final node.
Searching strategy in the trie (continued)

If the surface word is itself a root word, then we will reach to a final node.

If the surface word is not a root word, then the trie is navigated upto that node where the surface word completely ends or there is no path to navigate.
Searching strategy in the trie (continued)

If the surface word is itself a root word, then we will reach to a **final** node.

If the surface word is not a root word, then the trie is navigated upto that node where the surface word completely ends or there is no path to navigate.

We call this node as the **end** node.
Searching strategy in the trie (continued)

Now two different cases may occur here.

1. In the path from initial node to the end node, if one or more than one final nodes are found, then pick that final node which is closest to the end node. The word represented by the path from initial node to the picked final node is considered as the lemma.
Now two different cases may occur here.
Now two different cases may occur here.

1. In the path from initial node to the end node, if one or more than one final nodes are found, then pick that final node which is closest to the end node.
Now two different cases may occur here.

1. In the path from initial node to the end node, if one or more than one final nodes are found, then pick that final node which is closest to the end node.

The word represented by the path from initial node to the picked final node is considered as the lemma.
Examples

Consider two inflected words 'অংেশর'/'angsher' and 'অংশীদােরর'/'angshidaarer'. 'অংেশর'/'angsher' comes from 'অংশ'/'angsh'. 'অংশীদােরর'/'angshidaarer' comes from 'অংশীদার'/'angshidaar'.

Examples

Consider two inflected words ‘অংশের’/‘angsher’ and ‘অংশীদারের’/‘angshidaarer’.
Examples

Consider two inflected words ‘অংশের’/‘angsher’ and ‘অংশীদারের’/‘angshidaarer’.

‘অংশের’/‘angsher’ comes from ‘অংশ’/‘angsh’.
Examples

Consider two inflected words ‘অংশের’/‘angsher’ and ‘অংশীদারের’/‘angshidaarer’.

‘অংশের’/‘angsher’ comes from ‘অংশ’/‘angsh’.

‘অংশীদারের’/‘angshidaarer’ comes from ‘অংশীদার’/‘angshidaar’.
Examples

Consider two inflected words ‘অংশের’/‘angsher’ and ‘অংশীদারের’/‘angshidaarer’.

‘অংশের’/‘angsher’ comes from ‘অংশ’/‘angsh’.

‘অংশীদারের’/‘angshidaarer’ comes from ‘অংশীদার’/‘angshidaar’.
Searching strategy in the *trie* (continued)

If no root word is found in the path from the initial node to the end node, then find the final node in the trie which is closest to the end node. The word represented by the path from initial node to the picked final node is considered as the lemma. If more than one final nodes are found at the closest distance then pick all of them. Now, generate the root word(s) which is/are represented by the path from initial node to those picked final node(s).
Searching strategy in the trie (continued)

2. If no root word is found in the path from the initial node to the end node, then find the final node in the trie which is closest to the end node.
Searching strategy in the trie (continued)

2. If no root word is found in the path from the initial node to the end node, then find the final node in the trie which is closest to the end node.

The word represented by the path from initial node to the picked final node is considered as the lemma.
Searching strategy in the trie (continued)

2. If no root word is found in the path from the initial node to the **end** node, then find the **final** node in the trie which is closest to the **end** node.

The word represented by the path from initial node to the picked **final** node is considered as the lemma.

If more than one **final** nodes are found at the closest distance then pick all of them.
2. If no root word is found in the path from the initial node to the end node, then find the final node in the trie which is closest to the end node.

The word represented by the path from initial node to the picked final node is considered as the lemma.

If more than one final nodes are found at the closest distance then pick all of them.

Now, generate the root word(s) which is/are represented by the path from initial node to those picked final node(s).
Searching strategy in the trie (continued)

Finally among the generated root word(s), pick the root word(s) which has/have maximum overlapping prefix length with the surface word. By the phrase 'overlapping prefix length' between two words, we mean the length of the longest common prefix between them. Even at this stage if more than one roots are selected, then select any one of them arbitrarily as the lemma. As it is very rare to have more than one root words in this stage and if more than one root exist, then all are viable candidates.
Searching strategy in the *trie* (continued)

Finally among the generated root word(s), pick the root word(s) which has/have maximum overlapping prefix length with the surface word.
Searching strategy in the trie (continued)

Finally among the generated root word(s), pick the root word(s) which has/have maximum overlapping prefix length with the surface word.

By the phrase ‘overlapping prefix length’ between two words, we mean the length of the longest common prefix between them.
Finally among the generated root word(s), pick the root word(s) which has/have maximum overlapping prefix length with the surface word.

By the phrase ‘overlapping prefix length’ between two words, we mean the length of the longest common prefix between them.

Even at this stage if more than one roots are selected, then select any one of them arbitrarily as the lemma.
Finally among the generated root word(s), pick the root word(s) which has/have maximum overlapping prefix length with the surface word.

By the phrase ‘overlapping prefix length’ between two words, we mean the length of the longest common prefix between them.

Even at this stage if more than one roots are selected, then select any one of them arbitrarily as the lemma.

As it is very rare to have more than one root words in this stage and if more than one root exist, then all are viable candidates.
Examples

Consider the dictionary root words 'শুনা'/'shuna', 'শুনানি'/'shunani' and 'শুনানা'/'shunano'. Now, take an inflected word 'শুন'/'shune' which actually comes from 'শুনা'/'shuna'.
Examples

consider the dictionary root words
‘শুনা’/‘shuna’,
‘শুনানি’/‘shunani’ and
‘শুনানো’/‘shunano’.
Examples

consider the dictionary root words ‘শুনা’/‘shuna’, ‘শুনানি’/‘shunani’ and ‘শুনানো’/‘shunano’.

Now took an inflected word ‘শুনে’/‘shune’ which actually comes from ‘শুনা’/‘shuna’.
Examples

consider the dictionary root words ‘শুনা’/‘shuna’, ‘শুনানি’/‘shunani’ and ‘শুনানো’/‘shunano’.

Now took an inflected word ‘শুনে’/‘shune’ which actually comes from ‘শুনা’/‘shuna’.
Results

Based on the evaluation of the Morpheme Extraction Task – FIRE 2014, the results obtained on Bengali data using our lemmatization system are given in the following Table.

<table>
<thead>
<tr>
<th>TOTAL. Precision</th>
<th>TOTAL. Recall</th>
<th>F1-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>56.19%</td>
<td>65.08%</td>
<td>60.31%</td>
</tr>
</tbody>
</table>
Based on the evaluation of the Morpheme Extraction Task - FIRE 2014, the results obtained on Bengali data using our lemmatization system are given in the following Table.
Based on the evaluation of the Morpheme Extraction Task - FIRE 2014, the results obtained on Bengali data using our lemmatization system are given in the following Table.

<table>
<thead>
<tr>
<th>TOTAL. Precision</th>
<th>TOTAL. Recall</th>
<th>TOTAL. F1-measure:</th>
</tr>
</thead>
<tbody>
<tr>
<td>56.19%</td>
<td>65.08%</td>
<td>60.31%</td>
</tr>
</tbody>
</table>
Source of Lemmatization Errors

Compound words and out-of-vocabulary words are not considered in our algorithm. Root words are taken from dictionary but if the coverage of the dictionary used is not good, then that will cause errors. However, as there is no such good language independent lemmatizer for Indian languages, we hope our effort is a positive contribution.
Source of Lemmatization Errors

- Compound words and out-of-vocabulary words are not considered in our algorithm.

- Root words are taken from dictionary but if the coverage of the dictionary used is not good, then that will cause errors.
Source of Lemmatization Errors

- Compound words and out-of-vocabulary words are not considered in our algorithm.

- Root words are taken from dictionary but if the coverage of the dictionary used is not good, then that will cause errors.

However, as there is no such good language independent lemmatizer for Indian languages, we hope our effort is a positive contribution.
IndiLem@FIRE-MET-2014: An Unsupervised Lemmatizer for Indian Languages

Abhisek Chakrabarty

Basics of Lemmatization

Lemmatization Algorithm

Experimental Setup and Results
Questions ??
References


Thank You