CHAPTER 3

THE SemTRex FRAMEWORK

This chapter discusses the components of the SemTRex framework and its functionalities. The UNL indexer is also discussed in detail.

3.1 OVERVIEW OF THE SemTRex FRAMEWORK

Figure 3.1 shows the components of the SemTRex framework. The objective of the SemTRex framework is to represent the semantic content of a text at various levels.

Starting with UNL graphs, the first component of the SemTRex framework, CoReX, a semantic indexer builds indices as a concise representation of a sentence. This concise representation can be viewed as subgraphs of the sentence UNL graphs, and can be used as semantic indices. CoReX has been designed mainly to fulfil the need of IR systems, wherein the semantic index is a conceptual representative of a sentence, and is sufficient to retrieve relevant documents.
Figure 3.1 SemTRex Framework
In the case of NLP applications, such as, QA and summary generation systems, the answer or the summary needs to be more specific to the query, and consequently the semantic index needs to be representative of a paragraph or set of sentences. To build such a semantic index, the SemTRex framework initially attempts to build a discourse structure, using RST with the UNL. The UNL-RST discourse parser which forms the second component of the SemTRex framework, constructs a discourse structure for a clause, sentence, paragraph and document, given a set of UNL sentence, graphs and features.

Similar to the UNL-RST discourse parser, the UNL-saṅgati discourse parser, which is the third component of the SemTRex framework attempts to construct a discourse structure for a clause and sentence, using UNL graphs and UNL features. The UNL-RST- saṅgati discourse parser which forms the fourth component of the SemTRex framework, attempts to build a discourse structure for a clause, sentence, paragraph and document, given a set of UNL graphs of sentences along with UNL features and RST features.

The fifth component namely, the sūtra builder, builds a concise representation for the UNL-RST- saṅgati discourse structure, and finally the last component, the sūtra indexer generates inverted indices for sūtra representation.

Though UNL can be used to handle the semantics of a text at the intra sentence level by using nested UNL graphs, this research focusses on using the RST and saṅgati to handle semantics beyond the word level (Balaji et al 2012a).

The UNL-RST discourse structure, and UNL-RST-saṅgati discourse structure, UNL-saṅgati discourse structure, and the sūtra
representation can be used by any NLP application such as machine translation, summary generation, and abstract generation. The semantic indices, namely, the UNL indices and the sutra based indices constructed by the SemTRex framework are well suited for the query focussed NLP applications, such as IR, QA and summary generation systems.

The next section discusses the UNL indexer, CoReX, in detail.

3.2 CoReX- THE UNL INDEXER

Given a set of UNL graphs, CoReX identifies language independent UNL-based indices. Since the UNL documents are stored as graphs, the UNL-based indices form a sub graph. A sub graph forming an UNL index is comprised of a UW, a co-occurring UW, and a UNL relation linking these concepts. CoReX identifies such isomorphic sub graphs, and creates an index from the sub graphs, thereby capturing the semantic essence of the NL text.

3.2.1 Indexing Mechanism of CoReX

A UNL document is represented as a directed graph $G = (V, E)$ where, $V$ is a set of vertices (nodes) and $E$ is a set of edges that denote UWs and UNL relations respectively. Here, $V = \{n_1, n_2, n_3, ..., n_s\}$ and $E = \{r_1, r_2, r_3, ..., r_t\}$, where, the suffixes $s$ and $t$ denote the number of UWs and UNL relations present in the UNL graph, $G$. In the rest of this section, the terms, graph, vertex, edge and sub graph will be used as synonyms for the UNL document, the UW, the UNL relation and the UNL index respectively.

A node, “n” of a UNL graph $G$ is considered for indexing, if it meets any of the following criteria in the specified order.
(a) If the node, \( n \) is connected to one or more vertices in the graph then the node \( n \), the nodes that are connected to it, and the UNL relations that link them, are considered for indexing. (connectivity criteria).

(b) If the node, \( n \) occurs frequently (more than a threshold) in the UNL graph, \( G \), then the node is considered for indexing (frequency criteria).

A text index needs to be a significant representative of the text. The UW that is connected to many UWs implies that these UWs are semantically coupled in the document and need to be given importance. Furthermore, a UW that occurs frequently in a UNL graph indicates that it is predominantly used in the document, and could carry significant information of the text, which also needs to be given importance. Hence, CoReX makes use of these two criteria for identifying the UNL indices.

The indices identified are categorized into three types, namely, Concept-Relation-Concept (CRC), Concept-Relation (CR), and Concept only (C), indices. The Concept, \( C \), denotes a UW (node) and the Relation, \( R \), denotes the UNL relation.

These three sets of indices are identified as follows. If the node, \( \text{“} n_i \text{”} \) is connected to a set of nodes, \( n_j, n_{j+1}, \ldots, n_k \), through a set of UNL relations, \( r_j, r_{j+1}, \ldots, r_k \), then the node, \( n_i \), satisfies the connectivity criteria, and the following sets of indices are identified.

\[
\text{CRC-Index set}_i = \{ n_i - r_j, n_j - r_{j+1}, \ldots, n_i - r_k, n_k \};
\]

\[
\text{CR-Index set}_i = \{ n_i - r_k, n_i - r_{j+1}, \ldots, n_i - r_k \};
\]

\[
\text{C-Index set}_i = \{ n_i, n_{j+1}, \ldots, n_k \}.
\]
If a node, “\(n_i\)” satisfies the frequency criteria, the node \(n_i\) gets indexed and added to the C-Index set.

The above process is repeated for all the nodes present in the UNL graph \(G\). This results in identifying a set of sub graphs for each node present in the UNL graph \(G\).

### 3.2.2 Using CoReX for IR system

The idea behind forming three types of indices, is to capture the various semantic associations available in the UNL graph. The CRC type of indices form a sub graph that contains two nodes connected by a UNL relation, whereas, the CR type of indices focuses on the various UNL relations that are associated with the node \(n\), and the C type of index ignores the UNL relation between the nodes or UWs, and hence, the node \(n\), and the nodes that are connected to \(n\) are identified as the C type indices. When such indices are used in the NLP applications like the IR system, the indices will aid in retrieving the relevant documents depending on the query requirement. Since the index of a text document needs to bridge the gap between the query and the text documents, an efficient text index should fulfill the user’s need even when the query does not provide sufficient details. The usage of the three types of indices can be explained from the following two perspectives.

(a) From the perspective of the user query

(b) From the document relevance perspective

From the query perspective, the CRC type of indices is used when the query can be converted to a UNL graph, and it can be broken down into CRC type of sub graphs. From the document perspective, the documents that are retrieved by the CRC type of indices are more relevant to the query, when compared to those retrieved by the other types of indices. They are given the
highest priority while ranking the retrieved documents. The CR type of indices are helpful mainly, when only a partial enconversion of the query is possible. The C type of indices are used when it is not possible to find any UNL relations between the concepts present in the query. Since the C type indices ignore the semantic relations between the concepts, the documents retrieved by the C type indices are less relevant when compared to the others, and are given the least priority while ranking the retrieved documents. However, they help to retrieve conceptually similar documents.

When a node is indexed, the information such as, pos tag, UNL attributes, their frequency in the corpus, etc, which are inherent in it are retained in the index too. Each index identified is stored along with the sentence and document identifiers. A weight factor is assigned to each index based on its sentence position, and the frequency in the document UNL graph, which are later used in the ranking process.

The Concept of CoReX is illustrated using an example. Figure 3.2 shows the UNL graph representation for the query given in Example 3.1.

**Example 3.1**

Query: “Intiyāviṇ aruṇārūṅaśāciyānkaṇaḥ”

English Translation: Museums of Government of India

English Transliteration: Inthiyavin Arusu Arungaatchiyagankal

The CoReX based CoRee enconverts the NL query into a UNL graph, as shown in the figure.
The CoReX based CoRee retrieves the document, using the following indices:

- India(iof\textgreater\text{country})- pos-Government Museum (icl\textgreater\text{institution})-(C-R-C-type index)
- Government Museum (icl\textgreater\text{institution})-pos (C-R-type index)
- India(iof\textgreater\text{country}) (C-type index)
- Government Museum (icl\textgreater\text{institution}) (C-type index)

The CRC type index, “India (iof\textgreater\text{country})- pos-Government Museum (icl\textgreater\text{institution})” retrieves the document that contains both the UWs, India(iof\textgreater\text{country}) and Government Museum (icl\textgreater\text{institution}) in the same sentence, along with the UNL relation between them, thereby contributing to the accuracy of the retrieved document.

On the other hand, the documents retrieved by the C type indices may not always be relevant to the query, as the query concepts, namely, the India (iof\textgreater\text{country}) and Museum (icl\textgreater\text{institution}), may occur in different sentences of the documents. Hence, the semantic link between the two concepts is not captured by the C type indices. Hence, using C-R-C type of indices leads to a better relevance. This difference has been used in the ranking mechanism of CoRee to rank the retrieved documents.
3.2.3 The Influence of the CRC type indices

The semantics in the index can be used for the other tasks in the search engine, “CoRee” (Balaji et al 2012). CoReX has been used in the query expansion and ranking mechanisms of CoRee. The query expansion uses the CRC type of indices to expand the words present in the query. The usage is illustrated through an example query given to the CoRee search engine.

**Query:** தஞ்சை கோவில்

**English Transliteration:** Tañcai kōvil

**English Translation:** Tanjore Temple

Documents pertaining to the following terms will be retrieved.

- Tañcai kōvil (Tanjore Temple) and Rājrāja cōḷan (Raja Raja Chozhan)
- Rājrāja cōḷan ((Raja Raja Chozhan)
- Tañcai kōvil (Tanjore Temple)

Though the word, ராஜராஜ கோவில் ((Raja Raja Chozhan) is not present in the query, the documents pertaining to ராஜராஜ கோவில் ((Raja Raja Chozhan) are retrieved as the query, “தஞ்சை கோவில்” (Tanjore Temple) is expanded as தஞ்சை கோவில் (Tanjore Temple (iof>temple)- and- Raja Raja Chozhan(iof>person). This is done by the usage of the CRC index. The query, தஞ்சை கோவில் matches with the C₁ of the C₁RC₂ index, and the C₂ present in the index is added as an expanded query term by the query expansion module prior to searching. Now, both the
UWs, தஞ்சை கோயில் (Tanjore Temple) and Raja Raja Chozhan are used together for search and retrieval. This increases the relevance of the retrieved results.

The ranking module makes use of the weight factor for the refinement of the results. The ranking mechanism of CoRee ranks the documents by giving the highest priority to the documents retrieved through the CRC indices. The documents retrieved through the CR type indices get the next priority, and the least priority is given to documents retrieved through the C type indices.

3.3 EVALUATION

CoReX has been implemented and tested, by using 33000 tourism domain specific Tamil language documents enconverted to UNL, on the concept based search engine, CoRee (Balaji et al 2012).

The CoReX is evaluated for its scalability by increasing the number of documents considered. It is also tested for its ability to retrieve semantically relevant documents to the user given query, by using “Precision” as the metric.

The precision has been calculated at various levels, namely, Precision@5(P@5), Precision@10(P@10) and Precision@20 (P@20). P@5, denotes the precision calculated for the first five documents that are retrieved by an IR system. Similarly, P@10 and P@20 indicates first ten and twenty documents retrieved by the IR system. The Precision has been calculated for all these levels of precision using 139 queries.
A performance comparison has also been done in two different ways, namely, by comparing a Tamil Search Engine named, “Search Ko” (Sobha et al 2010) with CoRee which uses only C type indices. The Search Ko search engine uses terms for indexing, and a morphological analyser to expand the queries. CoReX based CoRee has been compared with Search KO in order to bring out the difference between a conceptual search engine and a term based search engine. Comparing with CoRee that uses C type indices has been done, in order to bring out the impact of using the CR and CRC indices in an IR system.

Table 3.1 shows the precision of CoRee using CoReX, CoRee using C type indices, and Search KO.

**Table 3.1 Performance Evaluation of CoReX**

<table>
<thead>
<tr>
<th>Precision</th>
<th>Search Ko</th>
<th>C- Type index based CoRee</th>
<th>CoReX based CoRee</th>
</tr>
</thead>
<tbody>
<tr>
<td>P@5</td>
<td>0.23</td>
<td>0.33</td>
<td>0.76</td>
</tr>
<tr>
<td>P@10</td>
<td>0.4</td>
<td>0.47</td>
<td>0.74</td>
</tr>
<tr>
<td>P@20</td>
<td>0.3</td>
<td>0.45</td>
<td>0.72</td>
</tr>
</tbody>
</table>

It can be observed from the table that, the precision calculated at P@5 achieved by the term based search engine, Search KO is 0.23; while the C-type index based CoRee and CoReX based CoRee have achieved a precision of 0.33 and 0.76 respectively. This indicates that when terms are used for query-index matching, the search engine is unable to find the relevant documents pertaining to the query. While looking at the C-type index based CoRee, it uses only concepts (UWs) for query-index matching, and achieves a better precision, when compared to that of the term based search engine. This is because a single concept may map to many terms thereby aiding in the retrieval of more relevant documents. Finally, CoReX based
CoRee is able to retrieve a higher number of relevant documents when compared to the other two search engines, as it considers also the semantic relation along with the concept for query-index match. This is mainly because of the CRC and CR type indices, which aid in retrieving documents semantically closer to the query. It can also be observed that there is a decrease in the precision values for P@10 and P@20 when compared to P@5. This can be improved by enhancing the weight factor given to each index. Also, the search mechanism and the ranking mechanism of CoRee can be improved.

Figure 3.3 shows the growth of the size of the indices of three types against the number of documents. The initial version of CoReX was implemented, using a Modified Binary Search Tree (BST) data structure, which did not scale beyond 33000 documents. However, CoReX has now been implemented using MySQL Data Base (DB), and has been tested for up to 7 lakh documents.

![Figure 3.3](image)

**Figure 3.3** The graph showing size of indices(Y-axis) versus Number of documents indexed (X-axis)

The list of 82 queries which were used for the evaluation is given in the Appendix 2. The reason behind the superior performance of CoReX based
CoRee when compared to the Search Ko search engine is obviously its conceptual quality. For instance, for the query, “தோங்ஙா நாகராம் (துங்கா நாகரம்) which means “The City that does not sleep” which indicates, “Madurai” a city in the Southern Part of India, Search Ko retrieves the pages that contains the query term. In CoReX based CoRee, the CRC indices capture the relation between the concepts, “Thoonga Nagaram (icl>city)” and “Madurai (iof>city)” which results in more relevant documents. The CoReX based outperforms the CoRee which uses only C type indices, which is also a conceptual search engine. The UNL relations captured by the CRC and CR type of indices are the main reason for this. For instance, for the query, “திருநெல்வேலியின் சிற்பம் (Tirunelveliyin cirappu)” which means “The speciality of Tirunelveli”, the C type indices retrieves documents that contain the concepts, “Tirunelveli(iof>district) and “Speciality (icl>attribute)” either together or any one of the concepts, whereas, the CoReX based CoRee retrieves the documents containing the CRC, “Speciality (icl>attribute)-pos-Tirunelveli(iof>district)” and also the CRs “Tirunelveli(iof>district)-pos”, which results in the retrieval of more relevant pages about “Tirunelveli”, thereby increasing the precision.

3.4 SUMMARY

The SemTReX framework aims at building semantic indices, using the semantic representation formed by three techniques, namely, UNL, RST and saïgati. The usage of these techniques has aided the SemTReX framework in handling the semantics at various levels of the text, and consequently constructing semantic indices at these levels. The SemTReX framework has been constructed in such a way that it can be integrated with any NLP application, provided the NL texts are enconverted to UNL graphs.
CoReX is a simple and efficient concept based semantic indexing technique, which uses UNL graphs as the input. CoReX aims at retrieving semantically relevant documents, rather than key word based information retrieval systems. CoReX captures the semantic association between the UWs through its different type of indices, and aids the search mechanism to retrieve documents semantically relevant to the query. CoReX finds its usage also in ranking, and the query expansion mechanism of the concept based search engine CoRee.