

An Integrated Digital Tool for Accessing Language Resources

Anil Kumar Singh, Bharat Ram Ambati

Language Technologies Research Centre, International Institute of Information Technology
Hyderabad, India

{anil, ambati}@research.iiit.ac.in

Abstract

Language resources can be classified under several categories. To be able to query and operate on all (or most of) these categories using a single digital tool would be very helpful for a large number of researchers working on languages. We describe such a tool in this paper. It is different from other such tools in that it allows querying and transformation on different kinds of resources (such as corpora, lexicon and language models) with the same framework. Search options can be given based on the kind of resource being queried. It is possible to select a matched resource and open it for editing in the specialized interfaces with which that resource is associated. The tool also allows the extracted or modified data to be saved separately, apart from having the usual facilities like displaying the results in KeyWord-In-Context (KWIC) format. We also present the notation used for querying and transformation, which is comparable to but different from the Corpus Query Language (CQL).

1. Introduction

Language resources form the backbone of Natural Language Processing (NLP) and many other disciplines. As the number of types, size and complexity of these resources increases, it becomes harder to keep track of all these resources and to access them. Researchers might also be interested in finding similar documents or linguistic units in all the resources available. This scenario brings out the urgent requirement for an integrated tool for accessing language resources (Singh, 2006). Such a tool should not only allow search and browsing, it should also allow them to be transformed in various ways, either in place or via a copy. If this tool could be connected to other tools and interfaces, then its usefulness will be further enhanced. We describe such a tool in this paper.

There have been previous attempts at building a general tool for accessing language resources, but the focus usually was on certain kinds of resources, say either corpora (Cunningham et al., 2000; Cunningham et al., 2003; Novák, 2007; Mirovsk, 2006) or lexicon (Havasi et al., 2005; Lenci et al., 2000). The variety of resources handled by the tool described in this paper is more than most of the other tools.

To give one example, language models, though very useful language resources, have not been included in the list of resources handled by the other resource access tools. There are tools for compiling language models (Stolcke, 2002), but not for browsing or querying them.

One of the tools that is perhaps the most similar to the tool being described here is Sketch Engine¹. It provides a wide variety of functionalities to access corpora. Some of these are:

- Searching word, lemma, root, POS tag of current word
- Left and right context upto a window size of 15
- A GUI to specify these options
- One can also directly give a CQL query.

This is similar to Jaxe², where one can provide provide XPath directly or use a GUI for providing attributes and values which get converted to an XPath based query internally.

Apart from CQL based querying, another usual practice is to have a query tool for syntactically annotated corpora such that the data is converted internally to relational database and the query is written using SQL (Kallmeyer, 2000). Yet another tool TigerSearch³ is also meant for linguistically annotated corpora.

The tool that we describe here is meant to be an integrated digital tool for accessing various kinds of language resources such as corpora, lexicon and language models. It allows resources to be searched in batch mode with many useful options and conditions. The resources currently handled include raw text, syntactically annotated corpora, corpora in XML format, bilingual dictionaries and n -gram models of various kinds. The matched documents can be saved at some other location. It is also possible to extract the matching portions and save them separately. A transformation option allows the retrieved documents to be modified. Also, certain kinds of documents (like syntactically annotated corpora) can be opened in their own annotation interfaces and can be edited directly. In the case of corpora, the matched sentences can be alternatively shown in the KWIC format. The tool also provides more general and user friendly facilities for searching syntactically annotated corpora and the corpora in XML format. Some basic statistics about the documents can also be displayed. Matched documents can be converted to some other supported format.

Another facility is to select one document and find all the similar documents based on contextual-distributional similarity. Many more extensions are being implemented and the tool has the potential to become a single interface through which a large variety of resources can be easily searched, accessed, analyzed and edited. Even in the

¹<http://www.sketchengine.co.uk/>

²<http://jaxe.sourceforge.net/>

³<http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERSearch>

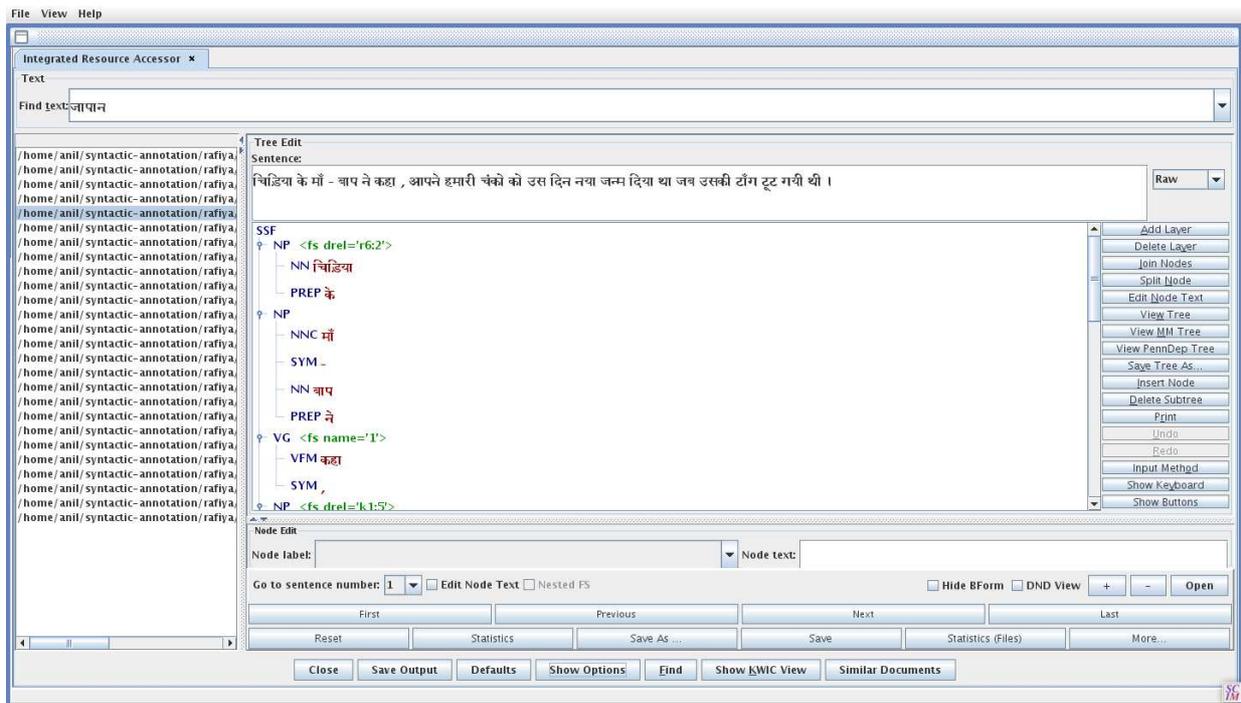


Figure 2: Search and edit syntactically annotated corpus

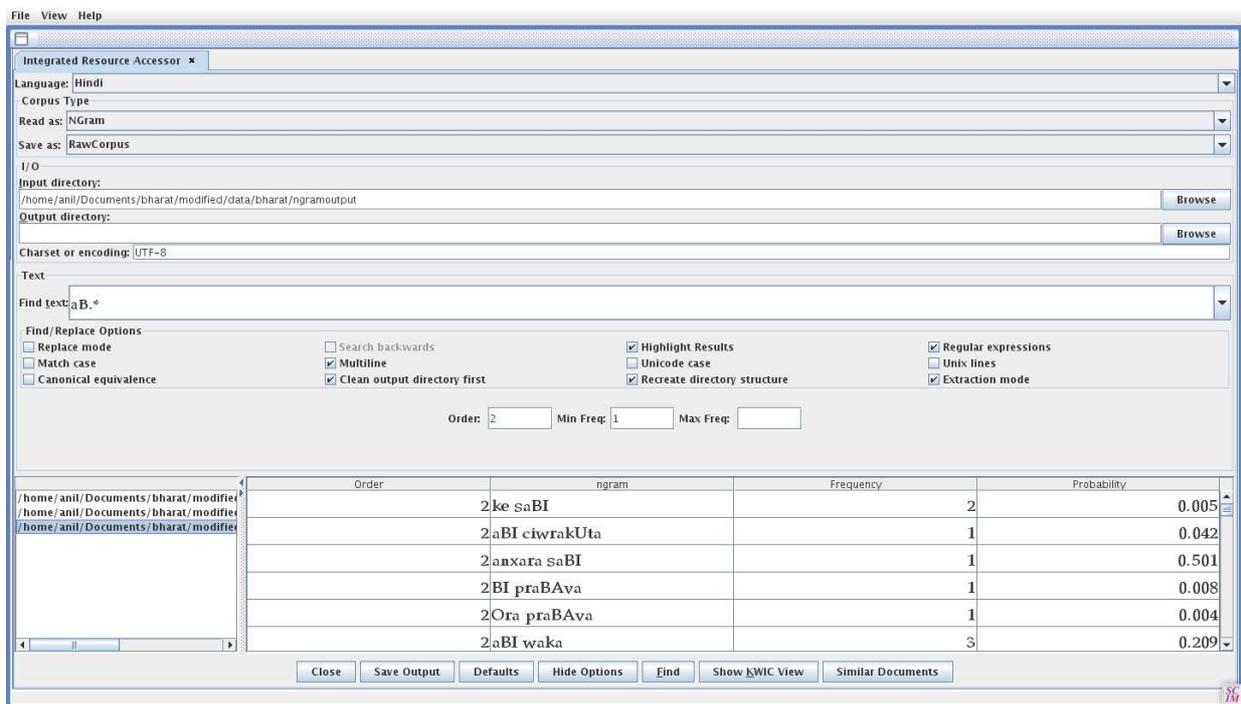


Figure 3: Search an n-gram language model

- Someone wants to browse, search and edit XML documents
- After searching for some pattern, someone wants to view the results in KWIC format
- An annotator wants to browse, search and edit syntactically annotated documents
- After manual annotation, an annotation adjudicator

wants to perform a 'sanity check' on the data

- After finding some document as a result of a search, someone wants to find contextually-distributionally similar documents

The demos for these scenarios can be seen at <http://ltrc.iiit.ac.in/anil/sanchay/integrated-resource-access-tool/>. Figures 2-4 illustrate three of these.

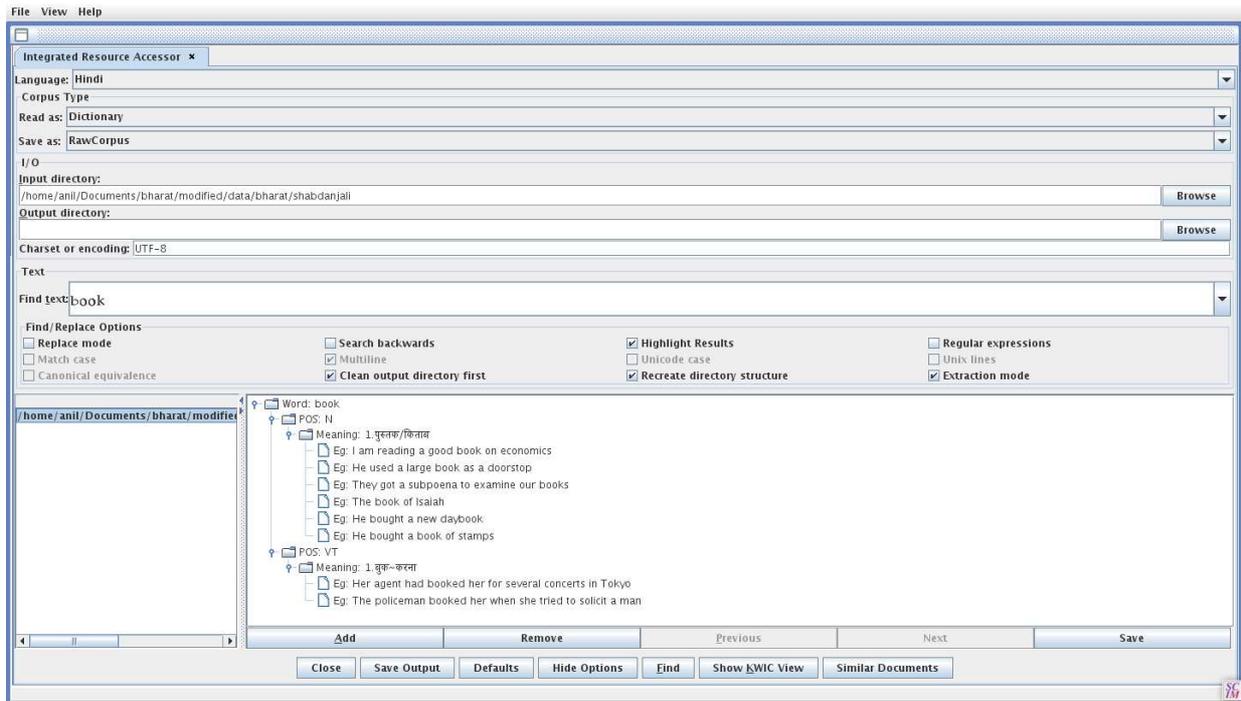


Figure 4: Search and edit a bilingual dictionary

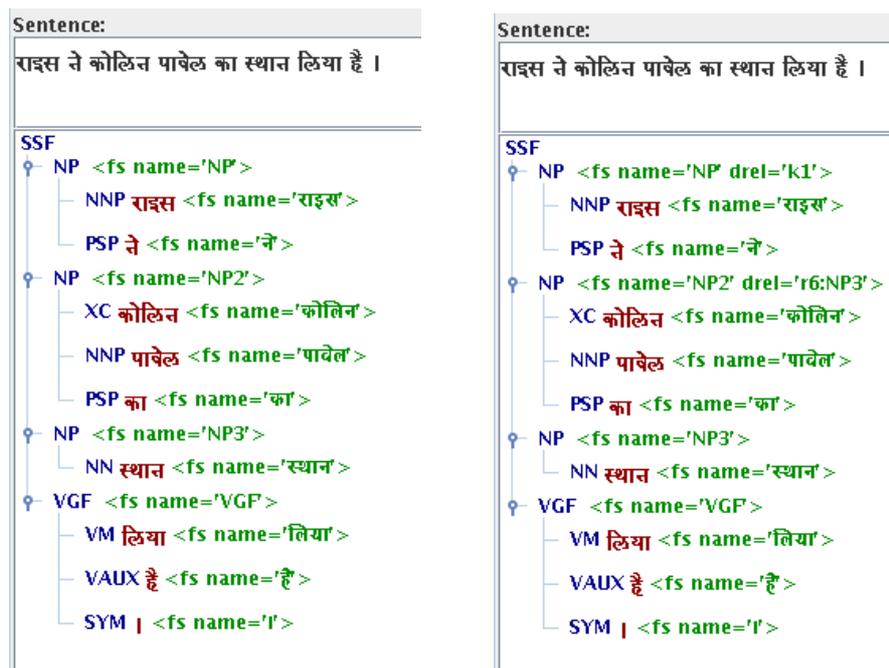


Figure 5: The result of a transforming query: The matched sentence is shown on the left, while the modified sentence is shown on the right.

4. Generalized Query and Transformation

The resource access method can be specified in terms of the following parameters:

- The type of resource being accessed
- The input and output formats
- The input and output locations
- The language and encoding: Important for languages which are not well supported on computers. Uses the Sanchay language and encoding support mechanism (Singh, 2008)
- The query (for searching as well as transformation), which depends on the type of the resource
- Additional options such as specifying whether the resource is to be displayed in an editor or not, whether

the results have to be displayed in KWIC format or not, whether the matching data is to be saved separately or not

To use the tool, the user has to go through the following steps, even though the exact sequence of the step would, of course, vary depending on the requirements and the kind of resources being accessed:

- Enter the text or regular expression or query to be searched and transformed
- Enter the replacement text or (if applicable)
- Click on the Find button to get a list of matching documents
- Double click on one of them to open it in the interface associated with that kind of resource
- Perform operations within that document
- So on for other documents
- Save the results after replacement or extraction, if applicable

The query for a resource like an n -gram language model would specify the text or regular expression to be searched, the size of the n -gram (unigram, bigram etc.) and frequency or probability range. For lexicon, the query would include the lexical item (possibly a regular expression) and attributes like the part-of-speech category.

For syntactically annotated corpora, the query can be relatively more complex. The annotated data is stored in a tree like structure where every node can have a feature structure consisting of attribute-value pairs. This simple structure allows many different kinds of syntactically annotated resources to be stored in the same format. The tree structure can be used to represent one particular kind of information such as chunks, phrase structure or dependency structure. Other kinds of information can be represented through attributes of the nodes. For example, the base tree can represent chunks whereas phrase structure and dependency structure can be represented implicitly via attributes. The APIs in Sanchay allow, for example, the dependency tree to be generated using the information stored in the attributes, i.e., a chunk tree can be easily converted to a dependency tree and vice-versa (see Figure 1), without losing any information.

Irrespective of the kind of information, the resources stored using this representation can be queried in terms of the tree nodes and their attributes. For queries we use a 'dot notation' for tree nodes and attributes that is somewhat similar to CQL. The notation is extended to express transformation rules and return values (on the right hand side) as shown in the examples below:

1. C.l = 'ने' AND C.t = 'PSP'
2. C.l = 'ने' AND C.t = 'PSP' ->
C.A[1].a['drel'] = 'k1'
3. C.l = 'का | के | की' AND C.t = 'PSP' ->
C.A[1].a['drel'] = 'r6:C.A[1].N[1].a['name']

4. C.l = 'ने' AND C.t = 'PSP' ->
C.A[1].a['drel'] and C.l

In the above examples, C is the current node (i.e., the node being searched), $A[1]$ is the ancestor at the distance of 1 (i.e., the parent node), $N[1]$ is the following sibling at a distance of one (i.e., the next node), l is the lexical data (i.e., the word or text, depending on the depth of the node), t is the part-of-speech (POS) tag and $a['drel']$ is the value of the attribute 'drel' (dependency relation).

Thus, the first query above searches for the word ने (ne) with the POS tag PSP (post-position). The second query searches for the same word, but it also specifies a transformation such that the 'drel' attribute of the parent node is assigned the value 'k1' ($kartA$, or roughly, agent). The third query searches for the words का, के and की (kA , ke and kI) having the tag PSP. It also specifies a transformation such that the 'drel' attribute of the parent node is assigned the value 'r6' (genitive) and this relation is pointed to the node next to the parent node by using ':' as the separator between the relation label and the relation reference or pointer. Figure 5 show a sentence before and after the application of these two transformations.

In the last query above, the variables on the right hand side represent return values.

This notation (for data as well as for queries) allows a wide variety of annotated corpora (including those with some semantic annotation) to be searched and transformed easily. The notation is much simpler than XPath queries, making it easier to use for those not familiar with XML.

5. Some Other Details

The tool, like other parts of Sanchay, is implemented completely in Java and is platform independent. A clear separation has been maintained between the underlying API and the front end. As mentioned earlier, resources searched can be directly opened in the annotation interfaces or in a text editor and edited. There is also a facility to search for similar documents and to compare two similar documents. For similarity, a two step algorithm is used. First, the context models of words in the documents are created. Then, one of the several distributional similarity measures (e.g., relative entropy) is used to calculate the similarity of two documents.

The language-encoding facility provided by Sanchay (Singh, 2008) is used for languages which may not be supported on some operating systems.

6. Conclusion and Future Work

We described a single tool to allow access to different kinds of language resources. The tool works for annotated corpora, language models and dictionaries. It accepts data in several formats and can be used to save data in these formats. Resources can be queried using a query language that allows transformations using a simple intuitive syntax. It is possible to open the matched documents in the associated editing or annotation interfaces and work on them. Matched data can be extracted and saved in a different location.

Many extensions are planned to make it more general. The most important ongoing extension is the online version of this tool, so that the resources can be accessed through a browser. Support for a lot of other formats and kinds of resources (especially ontologies like WordNet and ConceptNet) will also be added. Other facilities in Sanchay, like automatic annotation, n -gram compilation, word list compilation, dictionary FST (Finite State Transducer), approximate string search, different kinds of similarities etc. will also be connected with this tool. Another extension which is partially implemented is to allow access to resources from a BASH like shell in Sanchay. This would be useful for more experienced users.

7. References

- Hamish Cunningham, Kalina Bontcheva, Valentin Tablan, and Yorick Wilks. 2000. Software infrastructure for language resources: a taxonomy of previous work and a requirements analysis. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC-2)*, page <http://gate.ac.uk/>.
- H. Cunningham, V. Tablan, K. Bontcheva, M. Dimitrov, and Ontotext Lab. 2003. Language engineering tools for collaborative corpus annotation. In *Proceedings of Corpus Linguistics 2003*, pages 80–87. Wiley.
- Catherine Havasi, James Pustejovsky, and Marc Verhagen. 2005. Bulb: A unified lexical browser. In *Proceedings Language Resources and Evaluation Conference*.
- Laura Kallmeyer. 2000. A query tool for syntactically annotated corpora. In *Proceedings of the 2000 Joint SIG-DAT conference on Empirical methods in natural language processing and very large corpora*, pages 190–198.
- A Lenci, N Bel, F Busa, N Calzolari, E Gola, M Monachini, A Ogonowski, I Peters, W Peters, N Ruimy, M Villegas, and A Zampolli. 2000. Simple: A general framework for the development of multilingual lexicons. *International Journal of Lexicography*.
- J Mirovsk. 2006. Netgraph: A tool for searching in prague dependency treebank 2.0. In *Proceedings of The Fifth International Treebanks and Linguistic Theories Conference*.
- Jens Nilsson and Joakim Nivre. 2008. Malteval: an evaluation and visualization tool for dependency parsing. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.
- Václav Novák. 2007. Cedit – semantic networks manual annotation tool. In *Proceedings of NAACL-HLT*, pages 11–12, New York, USA.
- Anil Kumar Singh. 2006. Anil kumar singh. building an integrated digital tool for language resources. issue statement for the digital tools summit. east lansing, michigan. 2006.
- Anil Kumar Singh. 2008. A mechanism to provide language-encoding support and an nlp friendly editor. In *Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP)*, Hyderabad, India.
- A. Stolcke. 2002. Srilm – an extensible language modeling toolkit. In *Proc. of Intl. Conf. on Spoken Language Processing*, Denver, Colorado.