

A Linguistic Search Tool for Semitic Languages

Alon Itai

Knowledge Center for Processing Hebrew
Computer Science Department
Technion, Haifa, Israel
E-mail: itai@cs.technion.ac.il

Abstract

The paper discusses searching a corpus for linguistic patterns. Semitic languages have complex morphology and ambiguous writing systems. We explore the properties of Semitic Languages that challenge linguistic search and describe how we used the Corpus Workbench (CWB) to enable linguistic searches in Hebrew corpora.

1. Introduction

As linguistics matures, so the methods it uses turn towards the empirical. It is no longer enough to introspect to gather linguistic insight. Data is required. While most search engines look for words, linguists are interested in grammatical patterns and usage of words within these patterns. For example, searching the adjective "record" followed by a noun should yield "record highs" but not "record songs"; searching the verb *to eat* (in any inflection) followed by a noun, should yield sentences such as "John ate dinner." but not "Mary ate an apple" (the verb is followed by an article, not a noun)

To answer these needs, several systems have been constructed. Such systems take a corpus and preprocess it to enable linguistic searches. We argue that general purpose search tools are not suitable for Semitic languages unless special measures are taken. We then show how to use one such tool to enable linguistic searches in Semitic Languages, with Modern Hebrew as a test case.

2. Semitic Languages

Semitic languages pose interesting challenges to linguistic search engines. The rich morphology entails that each word contains, in addition to the lemma, a large number of morphological and syntactical features: part of speech, number, gender, case. Nouns also inflect for status (absolute or construct) and possessive. Verbs inflect for person, tense, voice and accusative. Thus one may want to search for *a plural masculine noun, followed by a plural verb in past tense with accusative inflection first person singular*.

Additional problems arise by the writing system. Some prepositions and conjunctives are attached to the word as prefixes. For example, in Hebrew the word *bbit*¹ may only

be analyzed as the preposition *b* (in) + the noun *bit* (house), whereas the word *bit*, which also starts with a *b* can be analyzed only as the noun *bit*, since the remainder, *it*, is not a Hebrew word. Thus to find a preposition one needs to perform a morphological analysis of the word to decide whether the first letter is a preposition or part of the lemma. Hence, in order to extract useful information the text has to be first morphologically analyzed.

All this leads to a high degree of morphological ambiguity. Ambiguity is increased since the writing systems of Arabic and Hebrew omit most of the vowels. In a running Hebrew text a word has 2.2-2.8 different analyses on the average. (The number of analyses depends on the corpus and the morphological analyzer – if the analyzer distinguishes between more analyses and if it uses a larger lexicon it will find more analyses.)

Ideally one would wish to use manually tagged corpora, i.e., corpora where the correct analysis of each word was manually chosen. However, since it is expensive to manually tag a large corpus, the size of such corpora is limited and many interesting linguistic phenomena will not be represented. Thus, one may either use automatically tagged corpora or use an ambiguous corpus and retrieve a sentence if any one of its possible analyses satisfies the query. We preferred the latter approach because of the high error rate of programs that attempt to find the right analysis in context. (An error rate of 5% per word entails that a 20 word sentence has probability $(1-0.05)^{20} \approx e^{-1}$ of being analyzed incorrectly.) Moreover, since these systems use Machine learning (HMM or SVM) (Bar Haim et al. 2005, Diab et al. 2004, Habash et al 2005), they prefer the more common structure, thus rare linguistic structures will be more likely to be incorrectly tagged. However these are exactly the phenomena a corpus linguist would like to search. Consequently, to successfully perform linguistic searches one cannot rely on the automatic morphological disambiguation and it would be better to allow all possible analyses and retrieve a sentence even if only one of the analyses satisfies the query.

¹ We use the following transliteration
אבגדחזכצפעסמלכיתחוזהדגבא abgdhwzXTiklmnsypcqršt

3. CWB

CWB – the Corpus Workbench – is a tool created at the University of Stuttgart for searching corpora. The tool enables linguistically motivated searches, for example one may search a single word, say "interesting". The query language consists of Boolean combinations of regular expressions, which uses the POSIX EGREP syntax, e.g. the query

```
"interest(s|(ed|ing)(ly)?)?"
```

yields a search for either of the words *interest*, *interests*, *interested*, *interesting*, *interestedly*, *interestingly*.

One can also search for the lemma, say "lemma=go" should yield sentences containing the words *go*, *goes*, *going*, *went* and *gone*. The search can be focused on part of speech "POS=VERB". CWB deals with incomplete specifications by using regular expressions. For example, a verb can be subcategorized as VBG (present/past) and VGN (participle). The query [pos="VB.*"] matches both forms and may be used to match all parts of speech that start with the letters VB. ("." matches any single character and "*" after a pattern indicates 0 or more repetitions, thus ".*" matches any string of length 0 or more.) Finally, a query may consist of several words thus ["boy"] [POS=VERB] yields all sentences that contain the word *boy* followed by a verb.

To accommodate linguistic searches, the corpus needs to be tagged with the appropriate data (such as, lemma, POS). The system then loads the tagged corpus to create an index. To that end the corpus should be reformatted in a special format.

CWB has been used for a variety of languages. It also supports UTF-8, thus allowing easy processing of non Latin alphabets.

4. Creating an Index

In principle we adopted the CWB solution to partial and multiple analyses, i.e., use regular expressions for partial matches. We created composite POS consisting of the concatenation of all subfields of the analysis. For example, the complete morphological analysis of *hšxqnim* "the (male) players" is

"NOUN-masculine-plural-absolute-definite", we encode all this information as the POS of the word, [pos="šxqnim-NOUN-masculine-plural-absolute-definite"], the lemma is *šxqnim*, the main POS is noun, the gender masculine, the number plural, the status absolute and the prefix *h* indicates that the word is definite. We included the lemma, since each analysis might have a different lemma.

To accommodate for multiple analyses, we concatenate all the analyses (separated by ":"). For example,

mxiiibim

:*mxaiib*-ADJECTIVE-masculine-plural-abs-indef

:*xiib*-PARTICIPLE-Pi'el-*xwb*-unspecified-masculine-plural-abs-indef

:*xiib*-VERB-Pi'el-*xwb*-unspecified-masculine-plural-present:PREFIX-*m*-preposition

:*xiib*-NOUN-masculine-plural-abs-indefinite

:PREFIX-*m*-preposition- *xiib*-ADJECTIVE- masculine-plural -abs-indef.

The analyses are:

1. The adjective *mxiiib*, gender masculine, number plural, status absolute and the word is indefinite;
2. The verb *xiib* it is a participle of a verb whose binyan (inflection pattern of verb) is Pi'el, the root is *xwb*, the person is unspecified, gender masculine, number plural, the type of participle is noun, the status absolute and the word is indefinite.
3. A verb whose root is *xwb*, binyan Pi'el, person unspecified, number plural and tense present.
4. The noun *xiib*, prefixed by the preposition *m*.
5. The adjective *xiib*, prefixed by the preposition *m*.

Thus one can retrieve the word by any one of the queries by POS:

```
[POS="*-ADJECTIVE-*"], [POS="*-PARTICIPLE-*"],  
[POS="*-VERB-*"], [POS="*-NOUN-*"].
```

However, one may also specify additional properties by using a pattern that matches subfields:

```
[POS="*PREFIX-[^]*preposition[^]*-NOUN-*"]
```

indicating that we are searching for a noun that is prefixed by a preposition. The sequence [^]* denotes any sequence of 0 or more characters that does not contain ":" and is used to skip over unspecified sub-fields. Since the different analyses of a word are separated by ":" and ":" cannot appear within an analysis, the query cannot be satisfied by matching the part of the query by one analysis and the remainder of the query by a subsequent analysis.

To create an index from a corpus, we first run the morphological analyzer of MILA (Itai and Wintner 2008) that creates XML files containing all the morphological analyses for each word. We developed a program to transform the XML files to the above format, which conforms to CWB's index format. Thus we were able to create CWB files.

Our architecture enables some Boolean combinations. Suppose we wanted to search for a two-word expression noun-adjective that agree in number. We therefore could require that the first word be a singular noun and the second word a singular adjective or the first word is a plural noun and the second word a plural adjective. The query

```
( [pos="*NOUN-singular-*"]  
[pos="*ADJECTIVE-singular-*"] )  
| ( [pos="*NOUN-plural-*"]  
[pos="*ADJECTIVE-plural-*"] )
```

However, one must be careful to avoid queries of the type
`[pos=".*NOUN.*" & pos=".*-singular-.*"]`
 since then we might return a word that has one analysis as
 a plural noun and another analysis as a singular verb.

5. Performance

To test the performance of the system we uploaded a file of 814,147 words, with a total of 1,564,324 analyses, i.e., 2.36 analyses per word. Table 1 shows a sample of queries and their performance. The more general the queries the more time they required. However, the running time for these queries is reasonable. If the running time is linear in the size of the corpus, CWB should be able to support queries to 100 million word corpora. One problem we encountered is that of space. The index of the 814,147 word file required 25.2 MB. Thus each word requires about 31 bytes. Thus a 100 Million word corpus would require a 3.09 Gigabyte index file.

6. Writing Queries

Even though it is possible to write queries in the above format we feel that it is unwieldy. First the format is complicated and one may easily err. However more importantly, in order to write a query one must be familiar with all the features of each POS and in which order they appear in the index. This is extremely user-unfriendly and we don't believe many people will be able to use such a system.

To overcome this problem, we are in the process of creating a GUI which will show for each POS the appropriate subfields and once a subfield is chosen a menu will show all possible values of that subfield. Unspecified subfields will be filled by placeholders. The graphic query will then be translated to a CWB query and the results of this query will be presented to the user. We believe that the GUI will also be helpful for queries in languages that now use CWB format.

Regular Expression	Time (sec)	Output File (KB)
<code>[pos=".*-MODAL-.*"] [pos=".*היה-.*"] [pos=".*-VERB-[^:]*-infinitive:.*"];</code>	0.117	13
<code>[word="על"] [word="מנת"] [pos=".*-VERB-[^:]*-infinitive:.*"];</code>	0.038	28
<code>[word="על"] [word="מנת"] [pos=".*-PREFIX-ש-.*"];</code>	0.025	5
<code>[pos=".*:הלך-[^:]*-present:.*"] [pos=".*-VERB-[^:]*-infinitive:.*"];</code>	0.099	2
<code>[word="בית"] [pos=".*:ספר-.*"];</code>	0.017	7
<code>[word="בית"] [pos=".*:ספר[^:]*-SUFFIX-possessive-.*"];</code>	0.014	1
<code>"כותב";</code>	0.009	
<code>[pos=".*:[^:]*-VERB-[^:]*:.*"];</code>	0.569	
<code>".*";</code>	1.854	
<code>[pos=".*"];</code>	1.85	
<code>".*"; [pos=".*"];</code>	3.677	
All the previous regular expressions concatenated	7.961	
<code>("כותב" [pos=".*:[^:]*-VERB-[^:]*:.*" ".*" [pos=".*"]);</code>	2.061	
<code>(([pos=".*:[^:]*-VERB-[^:]*:.*" & word="כותב" & word=".*" & pos=".*"]);</code>	0.168	

Table 1: Example queries and their performance.

7. Conclusion

Until now Linguistic searches were oriented to Western languages. Semitic languages exhibit more complex patterns, which at first sight might require designing entirely new tools. We have showed how to reuse existing tool to efficiently conduct sophisticated searches.

The interface of current systems is UNIX based. This might be acceptable when the linguistic features are simple, however, for complex features, it is virtually impossible to memorize all the possibilities and render the queries properly. Thus a special GUI is necessary.

8. Acknowledgements

It is a pleasure to thank Ulrich Heid, Serge Heiden and Andrew Hardie who helped us use CWB. Last and foremost I wish to thank Gassan Tabajah whose technical assistance was invaluable.

9. References

Official Web page of CWB: <http://cwb.sourceforge.net/>

Bar Haim, R., Sima'an, K. and Winter, Y. (2005). Choosing an Optimal Architecture for Segmentation and POS-Tagging of Modern Hebrew. *ACL Workshop on Computational Approaches to Semitic Languages*.

Buckwalter, T. (2002). Buckwalter Arabic Morphological Analyzer Version 1.0. *Linguistic Data Consortium catalog number LDC2002L49, ISBN 1-58563-257-0*.

Buckwalter, T. (2004). Buckwalter Arabic Morphological Analyzer Version 2.0. *Linguistic Data Consortium catalog number LDC2004L02, ISBN 1-58563-324-0*.

Christ, O. (1994). A modular and flexible architecture for an integrated corpus query system. In *Papers in Computational Lexicography (COMPLEX '94)*, pp. 22--32, Budapest, Hungary.

Christ, O. and Schulze, B. M. (1996). Ein flexibles und modulares Anfragesystem für Textcorpora. In H. Feldweg and E. W. Hinrichs (eds.), *Lexikon und Text*, pp. 121--133. Max Niemeyer Verlag, Tübingen.

Diab, M., Hacıoglu, K. and Jurafsky, D. (2004). Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. In *HLT-NAACL: Short Papers*, pp. 149--152.

Habash, N. and Rambow, O. (2005). Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of the 43rd Annual Meeting of the*

Association for Computational Linguistics, pp. 573--580, Ann Arbor.

Hajič, J. (2000). Morphological tagging: data vs. dictionaries. In *Proceedings of NAACL-ANLP*, pp. 94--101, Seattle, Washington.

Hajič, J. and Barbora Hladká, B. (1998). Tagging Inflective Languages: Prediction of Morphological Categories for a Rich, Structured Tagset. In *Proceedings of COLING-ACL 1998*. pp. 483--490, Montreal, Canada

Itai, A. and Wintner, S. (2008). Language Resources for Hebrew. *Language Resources and Evaluation*, 42, pp. 75--98.

Lee, Y-S. et al. (2003). Language model based Arabic word segmentation. In *ACL 2003*, pp. 399--406.

Segal, E. (2001). Hebrew morphological analyzer for Hebrew undotted texts. M.Sc. thesis, Computer Science Department, Technion, Haifa, Israel.