Building a Lexicon Database for Arabic Dialects

David Graff
Lead Programmer/Analyst, LDC

LDC Institute Presentation
Dec. 8, 2005
Overview

• Goals:
  – Desired properties of a lexicon and transcripts

• Common hurdles

• Hurdles particular to Arabic

• Some methods of approach:
  – Transcripts as plain-text or structured data files
  – Lexicon as plain-text or “tool-based” flat table
  – Lexicon as relational database (RDB)

• The next step -- now in progress:
  – Putting transcripts into the same RDB with the lexicon
What makes a good lexicon

• Full coverage for an adequately-sized corpus
  – List and describe all “word” tokens that are valid
  – Possibly add some common words not in the corpus
  – Token-count coverage of related corpora should be > 95%

• Consistent application of annotation conventions
  – Maximize the use of closed-set categorizations
  – Minimize the use of variant forms that “mean the same thing”
  – Centralize quality control activities among a small group of experts, working together closely

• Tight coupling between lexicon QC and transcripts
  – Error correction in the lexicon should propagate back to the transcripts
Common obstacles

• Axioms of manual transcription and annotation:
  – When a given task is done by N different people, there will be N different interpretations and techniques for the task.
  – The more often something must be done manually, the more mistakes will be made. (Typical minimum error rate is 5%)
  – For annotations involving unconstrained keyboard input, each person will create variant forms of a given annotation, and no two people will create the same form for it.
  – Most closed-set categorizations require a “miscellaneous” category. (Exception (?): “Is/Has X” vs. “Isn’t/Lacks X”)
  – For any particular annotation, a measurable percentage of instances within a corpus are indeterminate or ambiguous.
    • (“this guy” / ”the sky”)

Dec. 8, 2005
LDC Institute presentation 4
Common obstacles (cont.)

• Dependencies across layers of annotation
  – First layer: time segmentation of audio signal
  – Second layer: transcription of audio segments
    • Might catch segmentation errors
  – Third layer: building a lexicon from transcripts
    • Catches inconsistent spellings in transcription
  – Fourth layer: disfluency annotation
    • Catches segmentation and transcription errors
  – Fifth layer: treebanking
    • Potentially revises all previous layers

• **Worst case:** “Independent” layers create divergent versions of the same corpus.
Some Basic Problems with Arabic…

- The Arabic script-based writing system poses significant challenges for computational treatment.
  - Bidirectionality -- hard to render, harder to navigate and edit
  - Complex font with context-dependent rules for ligatures, glyph shape and character width

- Standard orthographic conventions represent an archaic form of the language.
  - Not “native” to any speaker of a current colloquial dialect
  - Colloquial (native) dialects have no standard orthography

- The absence of short vowels increases the difficulty.
  - Syntactic knowledge needed for correct word identification
  - Multiple meanings/pronunciations for a single written form
… and Special Problems with Colloquials

• Native speakers receive no formal instruction about their language -- no externalized grammar/analysis.
• Inherent phonological variability is unconstrained by - does not compete with - orthographic conventions or “correct speech”, so variants can have equal footing.
• Selection of consistent, appropriate spellings and morphological analyses for words entails deliberate and speculative choices.
• A systematic assessment of similarities and differences among colloquial dialects has yet to be done.
A Brief History of Implementation Details

Transcripts:

• Plain-text files and plain-text editors (bad old days)
  – Diverse information types are stored together “in-line”
  – Machine interpretation of content is difficult and brittle
  – Every corpus builder creates a new format
  – All data is manually editable -- nothing is safe

• Structured data files (XML) and specialized editors
  – Language content is always distinguishable from annotation
  – Many tools available for easy, reliable processing of data
  – Scope of format variation is constrained but not limiting
  – Scope of annotator effort is focused on appropriate tasks, and unrelated data is protected
Implementation Details (cont.)

Lexicons:

• Tab-delimited files and common (unix) shell tools
  – Tasks that are programmatic are fast, efficient, reliable; but:
  – Manual tasks are painful, and can break programmatic steps
  – Multi-stage manual work is especially risky, even with specialized tools for annotators/lexicographers

• Flat-table data and simple table-structured tools
  – Spreadsheets: easy to use and very capable, if entries are divided into reasonable sub-groups for handling/storage
  – Shoebox: has special attributes for linguists/lexicographers, but imposes its own set of limitations on what is possible
  – Data transfer across researchers/tools is simple and safe
Implementation Details: Lexicons (cont.)

• Relational Database (RDB)
  – Freely available servers are stable, easy to install, well documented, and can readily be made network-accessible.
  – Scalable to any size of lexical inventory with little or no effect on performance (speed)
  – Supports any appropriate conceptual model for lexicon creation, with configurable access permissions for users
  – Supports a wide rage of “sanity constraints” on input data (uniqueness, data type, string length, numeric min/max, …)
  – Provides lots of flexibility at the initial design stage and at any time thereafter (tables/fields can be added, modified)
  – Structured Query Language (SQL) provides a standardized, stable user interface for inserting, updating, retrieving data.
RDB Caveats

• Building a lexicon is always a complex process. RDB and SQL do not make it simpler (just more stable), and do require more technical expertise.

• Validating 50 K lexical entries is an inescapably long process. RDB will eliminate some delays and setbacks, but cannot reduce the basic effort required.

• Errors and failures are still possible -- on any scale.
Examples of Earlier Lexicons

• **Callhome**: tab-delimited, from transcripts & dictionaries
  
  $aGGAlaB $@GG%l@//$@GG%lit 010 $aGGAlaB:noun+fem-sg//$aGGAlaB:adj+fem-sg

• **Nahuatl** (Jon Amith): based on Shoebox and fieldwork
  – Includes multiple dialects in a single DB, with indexed audio
  – Web enabled for maintenance, expansion and pedagogy

• **CELEX**: relational tables for lemmas vs. word forms, covering frequency, morphology, pronunciation, syntax
  – Distributed as a set of cross-referenced flat tables
  – Includes comprehensive documentation (~150 pages)
The Next Step (where we are now)

- Take one colloquial Arabic dialect at a time
- Create or acquire conversational transcripts
  - Time-stamped “turns” that index the associated audio
  - “Skeletal” orthography (no short vowels)
  - Can also include pronunciations (short vowels) as a separate layer, but this is not essential
- Load the transcripts into database tables
- Add morphology/POS/gloss annotations
- Review, revise and refine, then dump tables into the publishable lexicon and transcripts
Database Table Structure

lex
- lex_revision
  - rwd_id int(11)
  - rgroup varchar(250)
  - rgroupsz int(11)
  - rdate datetime
  - rby varchar(30)
  - rorth1 varchar(100)
  - rorth2 varchar(200)
  - rseglbls varchar(250)
  - rcanon varchar(250)
  - rlglss varchar(250)
  - rdialec varchar(100)
  - rword_stat varchar(90)
  - rorth1_stat varchar(90)
  - rorth2_stat varchar(90)
  - rseglbls_stat varchar(90)
  - rcanon_stat varchar(90)
  - rlglss_stat varchar(90)
  - rdialec_stat varchar(90)

lex
- word_id (PRI) int(11)
- orth1 (MUL) varchar(100)
- orth2 varchar(200)
- segorth varchar(250)
- segblsls varchar(250)
- segmorph varchar(250)
- canon varchar(250)
- lgloss varchar(250)
- dialect varchar(100)
- rawfreq int(11)
- docfreq int(11)
- word_stat varchar(90)
- orth1_stat varchar(90)
- orth2_stat varchar(90)
- segorth_stat varchar(90)
- segblsls_stat varchar(90)
- canon_stat varchar(90)
- lgloss_stat varchar(90)
- dialect_stat varchar(90)

trans_file
- file_id (PRI) varchar(50)
- processed datetime
- source varchar(30)

trans_turn
- turn_id (PRI) int(11)
- tfid int(11)
- bgns word varchar(10)
- endsec varchar(10)
- chan char(1)
- spkr varchar(32)

trans_word
- twrd_id (MUL) int(11)
- ttrn_id (MUL) int(11)
- segnum tinyInt(4)
- w_stat varchar(30)

morph
- morph_id (PRI) int(11)
- segpos varchar(60)
- segtxt (MUL) varchar(200)
- mgloss varchar(250)
- dialect varchar(100)
Loading the tables

• For each transcript file:
  – Check for entry in trans_file, insert or update as needed
  – For each turn:
    • Check for entry in trans_turn, insert or update as needed
    • Delete entries (if any) from trans_word for this turn
    • For each word token:
      – Check for entry in lex, insert if needed
      – Add new entry to trans_word, citing turn-id, word-id, seq.number
      – Set “special feature” field in trans_word if token was uncertain
        “((this)) ((guy))“ or mispronounced “*nuclear”
Adding Morphology/POS/Gloss (MPG) Annotations

• Pull distinct words (skeletal “green” orthography) from lex table, sorted by frequency of occurrence in trans_word table (highest frequency first).

• Present one word at a time to an annotator, showing:
  – Skeletal (“green”) orthography
  – All associated vocalized (“yellow”) forms
  – Concordance drawn from token occurrences in turns

• Annotator provides:
  – “Canonical” vowelization
  – Segmentation into morphemes
  – Association of POS label to each morpheme
  – English gloss for each morpheme (and for word as a whole)
ABUMORPH Annotation Interface
created by Hubert Jin
Vetting / Validating MPG Annotations

• Summary reports of morph and lex entries:
  – Alternate sortings by POS labels and orthography
  – Including frequency of occurrence
• Web-based query tool with login access limited to lexicographers:
  – Generic query-generator for finding items and sets in either the lex table or the morph table
  – Listing of lex or morph table entries with links to listings of element occurrences, and links to “Entry Editor” form
  – Entry Editor supports modification in place, creation of new entry based on modified current entry, and merging of current entry into some other entry
Main Lexicon Search Page

- Search either the lex table or the morph table
- Use exact match, SQL “like” or regular-expression match
- Use single search criterion or two conjoined criteria:
  - “A and B”
  - “A and not B”
  - “A or B”
- Each “Search” button brings a separate pop-up window of “Search Results”
- Each pop-up is re-used on subsequent searches
### Lexicon Search Results Page

Click on an ID to see turns containing that entry. Click on a stat value to edit the entry.

<table>
<thead>
<tr>
<th>ID</th>
<th>Agreen</th>
<th>Bgreen</th>
<th>Ayellow</th>
<th>Byellow</th>
<th>N</th>
<th>POS_stat</th>
<th>W.Gloss</th>
<th>M.Gloss</th>
<th>Asegorth</th>
<th>Bsegorth</th>
<th>POS</th>
<th>Morph-IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>4925</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>0</td>
<td></td>
<td>ySuwfiwn</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5288</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>ySuwfuwn</td>
<td>0</td>
<td></td>
<td>ySuwfuwn</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>70454</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>ySuwfuwn</td>
<td>1</td>
<td>pass1</td>
<td>they see [masc.pl.]</td>
<td>see, look at, check, show [masc.pl.]</td>
<td></td>
<td></td>
<td></td>
<td>IV3MP IVYSUFF_SUBJP 1879 1503 1535</td>
</tr>
<tr>
<td>70455</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>يشوفون</td>
<td>ySuwfuwn</td>
<td>8</td>
<td>pass1</td>
<td>they see [masc.pl.]</td>
<td>see, look at, check, show [masc.pl.]</td>
<td></td>
<td></td>
<td></td>
<td>IV3MP IVYSUFF_SUBJP 1879 1503 1535</td>
</tr>
</tbody>
</table>

Click HERE to mark all these entries as VALID

- “ID” links produce pop-up of transcript concordance page
- “POS_stat” links produce pop-up of lex-entry editor page
- “N” shows current frequency of word occurrence in transcripts
• “Turn-ID” link fetches audio segment for the turn
• Transcription errors involving the target word can be corrected (so far, only word replacement is supported)
• Separate interface will be needed for word deletion/insertion
Lexicon Entry Editor Page

- Change Skeletal ("green") or Pronunciation ("yellow") spelling
- Change morphological composition and/or word gloss
- Update in place, or add as a new lex entry, or merge into some other existing lex entry (that is, render this entry obsolete)
Morph Search Results Page

- “ID” links to Lex Search Results to show all lex entries containing this morph entry
- “M_stat” links to a pop-up morph entry editor page
- “NLx” = number of lex entries currently using this morpheme
Morpheme Entry Editor Page

Modify the morph table entry for morph_id: 1879

<table>
<thead>
<tr>
<th>Field</th>
<th>Arabic</th>
<th>Btext</th>
<th>Change</th>
<th>New Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>segtxt</td>
<td>ي</td>
<td>Yi</td>
<td>to:</td>
<td>Yi</td>
</tr>
<tr>
<td>segpos</td>
<td>IV3MP</td>
<td></td>
<td>to:</td>
<td>IV3MP</td>
</tr>
<tr>
<td>mgloss</td>
<td>they [masc.pl.]</td>
<td></td>
<td>to:</td>
<td>they [masc.pl.]</td>
</tr>
<tr>
<td>morph_id</td>
<td>1879</td>
<td></td>
<td>merge into morph_id:</td>
<td>1879</td>
</tr>
<tr>
<td>Create new entry</td>
<td></td>
<td></td>
<td>Update in place</td>
<td>Send to DB</td>
</tr>
</tbody>
</table>

- Change the orthography, POS label and/or gloss
- Update in place, or create as a new entry, or merge all lex references to this entry so that they refer to some other morph entry instead (that is, render this entry obsolete)