Problem Space Specification: Browser-Based Linguistic Fieldwork Utilities

Luke D. Gessler[1] John Irby[2] Nakkul Sreenivas[3] Yugank Singhal[4]
ldg3fa[1], jmi8fs[2], ys5abl[4], ns3kb[3]
@ virginia.edu

1. Abstract:

Although many field linguists have made progress in documenting some endangered languages, very many of the world's languages remain unrecorded. Well-designed software has the potential to facilitate and magnify the work of field linguists. A few good software solutions are available, but there is reason to believe many fieldworkers find them cumbersome or unintuitive. Other solutions are discouraged because they are platform-specific. This paper describes a prototype built to improve upon these systems and its evaluation.

2. Design Components

The two components to be evaluated are the revised search functionality and the bulk import feature. The current search functionality present in existing field linguistics systems does not allow for search filters or complex queries. Typically, texts are only allowed to be filtered based on title, or in some cases author. The proposed improvements to the search system will offer more search options, such as language, dialect, grammatical structure, region, thematic tags (e.g. “story”, “folktale”), and grammatical category tags provided by the author or verified users. The filter will also make use of fuzzy string matching algorithms that take into account variance in spellings and transliteration of words. Evaluating this feature will help to identify possible issues with the existing search functionality, as well as improvements to our prototype.

The bulk import options is not currently a feature offered by existing field linguistics systems. Authors currently have to import each text individually, instead of a single import for multiple texts. Adding this feature could prove difficult, as many users may be familiar with the existing process. In conjunction with the search queries, a bulk import of common texts is easier to group and organize for a more efficient and thorough search in the future. Evaluating this feature will give insight into improving learnability for new features within the system.

3. Experimental Design

Search Functionality:

- Lucid and testable hypothesis:

  There is no statistically relevant difference between the time and number of inputs to complete a search using the existing search interface and the new interface.

- Independent Variables:

  The independent variables would be the test participant, the search query information given, and the system that is being tested on.

- Dependent Variables:

  The dependent variables would be the time taken to complete a search query, the number of inputs to the system to complete a search query, and the success or failure value for each query.

- Subject Selection:

  Ideally, at least two to three of each user group will be selected. When testing field linguists, it will be important to test two field linguists that have unrelated research. When testing native speakers, it is also important to sample individuals that speak different languages.

- Controlling bias:

  The main source of bias will be in the level of experience with the system. To help curb this bias, test participants will be allowed to practice on both the new interface and old interface for 5 minutes before the evaluation. Participants will also complete a qualitative survey regarding their level of experience and which aspects of each interface they preferred.

- Statistical analysis:

  One method to analyze the data collected would be to test against a confidence limit, such as testing the null hypothesis at the .05 level. Another analytical tool would be plotting the mean number of actions and search time of each user group on an interval scale. Considering the relatively small sample size for each group, performing a Fisher's Exact Test would be appropriate. If each category has more than five tested users (may be difficult given the specific nature of the product) then a Chi-Square Test can be used instead.

- Interpretation:

  The data can easily be represented graphically, via bar graphs for each user group. Another possibility would be a linear regression relating user experience on a nominal scale (obtained from the user survey) to performance in the evaluation. The major analysis tool will be comparing the means of each user group on the original system and the prototype. The user survey will also help to inform possible changes to the prototype.

Bulk Import:

- Lucid and testable hypothesis:

  There is no statistically relevant difference between the time and number of inputs to import a set number of texts using the existing import interface and the new bulk import.

- Independent Variables:

  The independent variables would be the test participants, the selected texts to upload, the file format to use in the bulk upload, and the number of texts to upload.

- Dependent Variables:

  The dependent variables would be the time is takes to upload all of the selected texts for each trial, as well as the number of inputs to the system for each trial.

- Subject Selection:

  Subject selection will primarily focus on researchers and field linguists, as they are the users of the import functionality. Among these users, testing from different backgrounds, as well as different experience levels is important. An example for researchers could be testing a tenured professor as well as an associate professor to gain insight into how each of them use the system.

- Controlling bias:

  The major source of bias will, once again, be the level of experience with the system. Allowing the participants to do a “test run” where they can try to use the bulk import feature will give them practice to help familiarize them with the new feature before the evaluation. Another method for controlling bias will be a qualitative survey which will address experience with the system. In particular, a question will address if the user has found a preferred method to import texts quickly using the existing system and how that method compares to the prototype.

- Statistical analysis:

  Similarly to the search analysis tools, one way to analyze the data would be to test against a .05 confidence level. Depending on the size of each of the user groups, a Fisher’s Exact Test or Chi-Square Test would be appropriate. Plotting and comparing the means for each user group, as well as utilizing an interval scale could prove useful.

- Interpretation:

  Plotting the mean user inputs and import times for each user group could provide a visual comparison of the data. In this evaluation, the user surveys will prove vital, as experienced users may have methods and techniques that can help to inform later design decisions.

4. Design Prototype

A possible prototype would be a rudimentary version of the final system. This would be a high-fidelity digital prototype, with the user account/home interface available. This would include a functional copy of the bulk import feature through the user's hub, as well as the search functionality. The database for this prototype would be a dummy database, filled with various texts to test for the search evaluation. The search functionality would include search fields including form, meaning, date, and speaker, which could then be extended to more advanced searches in following prototypes. These would enable a variety of test searches in the evaluation, requiring the use of different fields to quickly complete the search task.

The bulk import functionality would be implemented to accept JSON and XML. The functionality would also be extended to allow for the upload of multiple files and compressed file formats (ie- .rar, .zip). The bulk import tool would be located in a user's home view via a navigation bar. This would allow for comparisons between data entry using various file formats, as well as testing the learnability and efficiency of the bulk import tool itself.

5. Prototype

Our prototype was made to test improvements to the search functionality. This search prototype will be compared to the existing search functionality of one existing online fieldwork utility called LingSync. Our prototype allows searching texts by form or by meaning, making finding texts intuitive and fast. We also implemented a search filter to allow for more efficient and accurate searches.

Search is implemented with an AngularJS front-end and a JSON backend. The JSON backend will hold all the fields, such as form, meaning, date, speaker, etc, while Angular will be able to call a "get" request to the JSON values, and then display them dynamically on a webpage. Typing into the search will have Angular filter out the needed words into a separate array and then displaying it on the webpage. The filter our Angular prototype uses is a "fuzzy" filter, also known as an Approximate String Matching. In Approximate String Matching, Angular will search based on words which adds a letter, deletes a letter, or substitutes a letter to the word being searched.

The Git Repository for the project is located here: https://github.com/lgessler/OLF
The prototype can be viewed here: http://lgessler.github.io/OLF/
(Note: if website is unavailable, please clone repo into webserver.)

6. Evaluation

For the evaluation, participants were asked to complete a series of searching tasks, 3 in total, consisting of finding various texts in each system. Participants completed the search first in LingSync, then in our prototype. The number of inputs given to the system, as well as the time to complete each search task were recorded. After completing the task, the user completed a series of four survey questions about the test.

Task 1: Find the entry in the database that has the form *rinaywan.*

Task 2: Find from that means 'I feel like dancing a lot'.

Task 3: You only partially remember a phrase *Noqa** qa**rinaykuni*, where the asterisks are letters you have forgotten. You aren't sure whether it means 'I feel like yelling at you' or 'I feel like yelling at myself'. Please find this form in the database.

Participant Number / Task Number	**Number of Inputs (Prototype / Existing System)**	**Time (Prototype / Existing System)**	**Familiarity with Existing System (1 to 5)**	**Frustration with Prototype (1 to 5)**	**Frustration with Existing System (1 to 5)**	**Do you feel you completed the task? (Yes/No)**
Participant 1 / Task 1	8 / 9	3.6 seconds / 8.39 seconds	1	1	2	Yes
Participant 1 / Task 2	19 / 26	7.05 seconds / 9.41 seconds	1	1	1	Yes
Participant 1 / Task 3	13 / 61	11.32 seconds / 188.91 seconds	1	1	5	Yes
Participant 2 / Task 1	8 / 9	3.5 seconds / 30.06 seconds	1	1	2	Yes
Participant 2 / Task 2	15 / 26	5.91 seconds / 11.68 seconds	1	1	1	Yes
Participant 2 / Task 3	19 / 203	10.91 seconds / 207.15 seconds	1	2	5	Yes
Participant 3 / Task 1	9 / 1	9.29 seconds / 34.71 seconds	1	1	4	Yes

Participant 3 / Task 2	4 / 26	20.74 seconds / 19.3 seconds	1	1	1	Yes
Participant 3 / Task 3	5 / 10	40.65 seconds / 10 seconds	1	2	1	Yes
Participant 4 / Task 1	10 / 10	8.91 seconds / 9.85 seconds	1	1	2	Yes
Participant 4 / Task 2	25 / 28	16.02 seconds / 11.89 seconds	1	1	3	Yes
Participant 4 / Task 3	37 / 65	37.45 seconds / 43.27 seconds	1	2	4	Yes
Participant 5 / Task 1	9 / 11	4.33 seconds / 4.7 seconds	4	1	1	Yes
Participant 5 / Task 2	17 / 27	3.87 seconds / 5.4 seconds	4	1	3	Yes
Participant 5 / Task 3	15 / 19	10.08 seconds / 71.87 seconds	4	1	4	Yes

Task	**Average Inputs on Prototype**	**Average Inputs on Existing System**	**Average Time on Prototype**	**Average Time on Existing System**
Task 1	8.8	8	5.926 seconds	17.542 seconds
Task 2	16	26.6	10.718 seconds	11.536 seconds
Task 3	17.8	71.6	22.082 seconds	104.24 seconds

7. Design Considerations

The results of our evaluation seem to indicate that the “fuzzy” search filter is beneficial to the overall design. Task 3 was designed to test this feature to determine if it was an improvement. On average, participants took less time to complete task 3 on the prototype and used fewer inputs to complete it. Some results did indicate that there are additional features that can be added to the prototype in later development.

One interesting result came from participant 3, who utilized a scrollbar interface in the LingSync system that was not present in the prototype. This caused their completion of task 3 on the LingSync system to be much faster and use fewer inputs than those of the other participants. Participant 5 also utilized Ctrl+F to search through the LingSync interface, as they were familiar with a drop down list of all the search terms. Both of these interactions indicate that perhaps adding the option of a drop-down list that users can use Ctrl+F or a scrollbar to navigate would prove beneficial to some user groups. Another possible design addition is the use of additional fields. These could be author, date, or some sort of tag filtering. This would give users more options to quickly filter their search results.

One negative aspect of the LingSync system that the participants addressed during the survey was that the system was difficult to navigate and slow. Load times consumed a large amount of the search time for participants that chose to use the search bar. This is also seen in the higher scores that LingSync received for frustration in completing task 3.