Fuzzy search for historical records of Aboriginal languages

Sasha Wilmoth[1,2]
Simon Hammond[2]
Alice Kaiser-Schatzlein[2]

[1]University of Melbourne
[2]Appen

Background

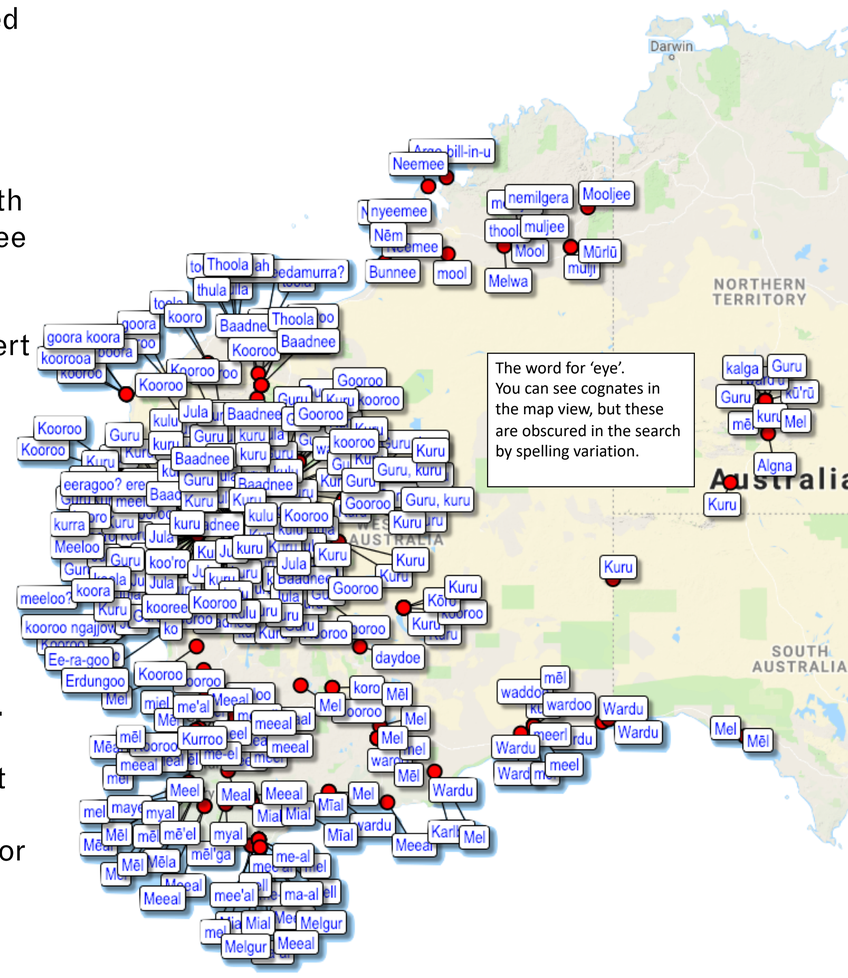
The Digital Daisy Bates Project (by CoEDL CI Nick Thieberger) digitised over 23,000 pages of handwritten Aboriginal language materials from the early 1900s, mostly from Western Australian languages. Much of this is surveys, with English words and phrases and their translation in the local language. The collection has many authors, and predates the development of standard orthographies for Aboriginal languages. The database is in XML, according to Text Encoding Initiative (TEI) standards.

Our goal was to make this database more easily searchable by all users, given the extreme lack of standardisation in spelling.

The solution

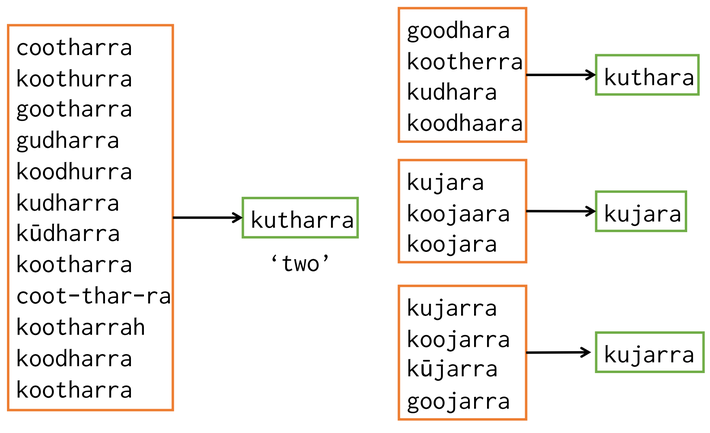
A Python script runs over each XML file as it is uploaded to the Daisy Bates website. The script standardises spelling, followed by generation of all possible variants that might result from the mis-hearing of unfamiliar sounds by English speakers. The website then searches across all of these variants, and is able to find words with many different original spellings. We use the ElementTree package in Python to handle XML trees.

Examples shown below are Pitjantjatjara/Western Desert words.



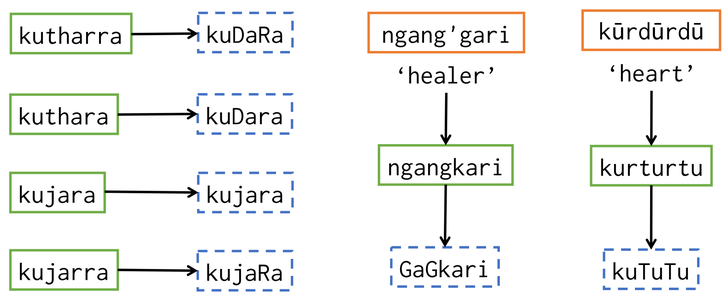
Step 1

The first step is to convert the original spellings (in all their wackiness) to a 'standard Australian' voiceless orthography. This is done using 100+ regular expressions.



Step 2

Next we convert digraphs to arbitrary single characters. This prevents unlikely variants from being generated later on. For example, for a word transcribed as *marlu*, it would generate *malu*, and from there also generate *malyu*, and *marlyu*. We therefore need unique symbols for each phone.



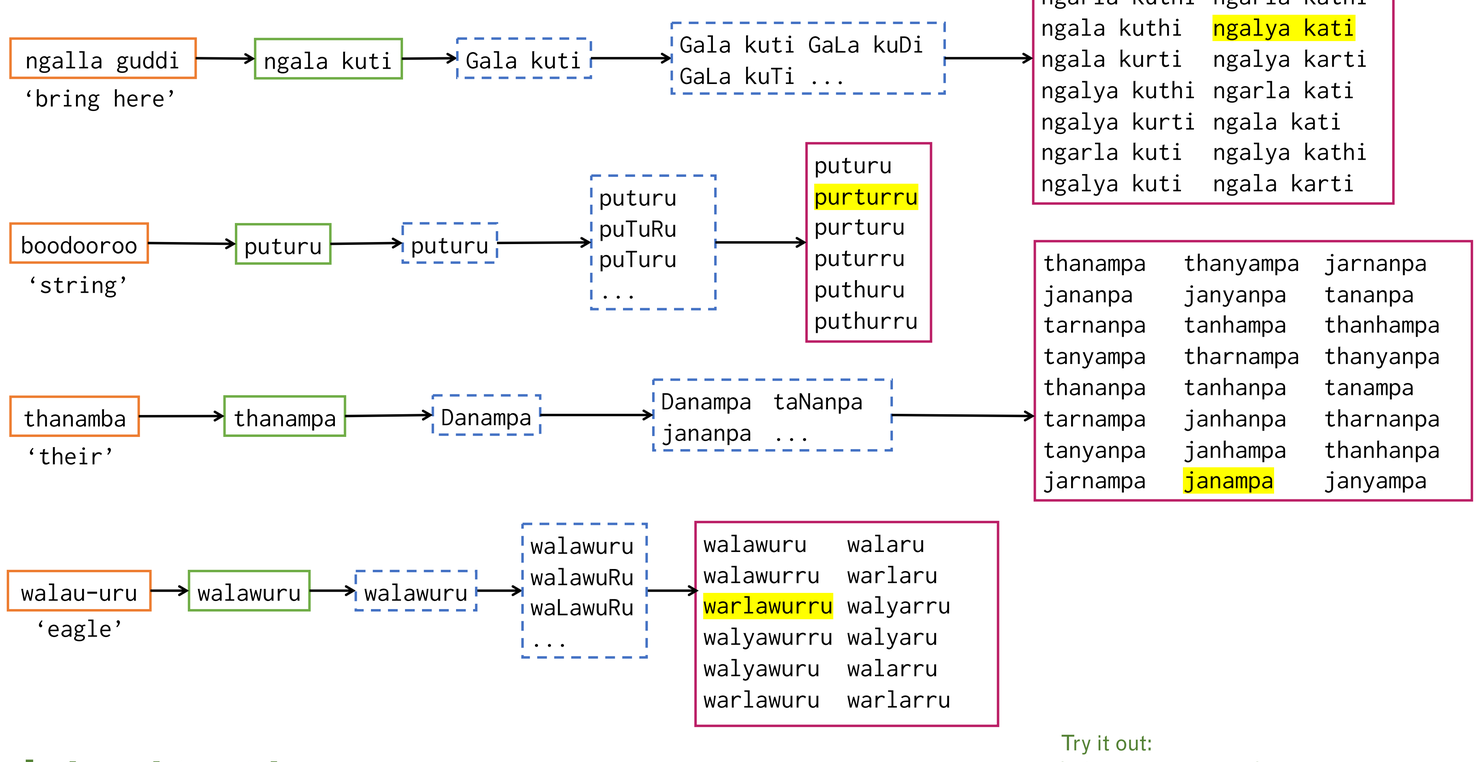
Step 3

Aboriginal languages have many sounds not found in English. In this step, we created a list of sounds that might be hard for an English speaker to perceive or transcribe. This file includes things like:

- retroflex, palatal, and dental consonants are often transcribed as alveolar (<n> instead of <rn, ny, nh>)
- initial velar nasals are often transcribed as <n>
- the vowel /a/ is often transcribed as <u>
- intervocalic <ng> could be a velar nasal (/ŋ/), or it could be a velar nasal + stop (/ŋk/)

Step 4

This file is then used to iteratively generate variant spellings, before converting the word back into digraphs (Step 2 in reverse). Some long phrases have several hundred possible variants!



Step 5

After some final checks, the script adds the full list of variants to the XML document. This is automatically uploaded to the Digital Daisy Bates website. The search looks in all elements behind the scenes, so if you search for *kujarra*, *Koodharra* will show as a result.


```
<cell><term ref="#two"><choice>
          <orig>Koodharra</orig>
</choice></term>
```

```
<cell><term ref="#two"><choice>
          <orig>Koodharra</orig>
          <corr>kutharra</corr>
          <reg>kuthara</reg>
          <reg>kujarra</reg>
          <reg>kutara</reg>
          <reg>kutarra</reg>
          <reg>kujara</reg>
</choice></term>
```

Limitations

1) The website assumes the search input has a standard spelling. (This is an easy fix and on the to-do list.)
2) Step 4 doesn't generate all possible variants when there is more than one instance of a grapheme in a word. For example, the entry <nganana> 'we' generates *nganhanha*, *nganyanya* and *ngarnarna*. The correct spelling would be *nganarna*, but the script doesn't make combinations of different types of 'n'. This is technically not a problem, but it's too computationally intensive to be practical (we left it running on Appen's servers overnight and it couldn't even get through 3 files!).

Try it out:
bates.org.au/search
(Search within language words, with fuzzy search on)

Get the code:
gitlab.com/swilmoth/daisybatesnormaliser

Get in touch:
swilmoth@student.unimelb.edu.au