Making the most of existing tools for the forced-alignment and segmentation of under-resourced languages

Stephen Nichols

stephen.nichols@phon.ox.ac.uk

Or: Hacking SPPAS

Stephen Nichols

`stephen.nichols@phon.ox.ac.uk`

This talk

- Discuss how the forced-aligner SPPAS can be adapted for use with languages currently lacking support
- Provide you with an introduction to a streamlined workflow for accomplishing this using data from Yoruba
- In my PhD project, I took this approach with three Bantu languages: Bemba, Lozi, Nyanja
- I am also currently doing so for two indigenous languages of Mexico: Amuzgo (San Pedro, Xochistlahuaca), Huave (San Francisco del Mar)

Forced-alignment

- Forced-alignment is the process by which an audio recording is aligned with its transcription (e.g. with a TextGrid in Praat)
- This typically involves not only alignment of sentences and words but also the segmentation of the speech stream into phones
- Automated alignment and segmentation is then typically manually corrected
- Nevertheless, this automation greatly speeds the process whole up (even just creating the right boundaries will do this)

Forced-aligners

- There are various options for forced-aligners, especially for more well-resourced languages, e.g.:
  - FAVE-align (English only)
  - Penn Phonetics Forced Aligner
  - (Web)Maus (30+ languages; also adaptable)
  - EasyAlign (French, Spanish and others)
  - Prosodylab-Aligner (English but adaptable)
  - Montreal Forced Aligner (various but also adaptable)

SPPAS

- SPeech Phonetization Alignment and Syllabification
  - http://sppas.org/
- Free and open source
- In-built support for various languages:
  - E.g. French, Catalan, Polish, Mandarin, Cantonese, Japanese, Korean
- But support for other languages can also easily be created

SPPAS

- The possibility of customised support new languages is especially useful when working on languages for which resources are limited – or non-existent
- But I can take advantage of existing resources such as SPPAS to save on time/labour as much as possible
- The rest of this talk will consist of a (somewhat rough-and-ready) walkthrough/demonstration which aims to build support for Yoruba
- (NB I don’t at all mean to imply that SPPAS is the only way to go!)

Demo

- In order to be able to support the forced-alignment of a language, SPPAS requires the following “resources”:
  - An acoustic model
  - A vocabulary file
  - A dictionary file
- For new languages, these must be created/supplied by the user
- Let’s do that for Yoruba!

Demo

- The resources required:
  - `/resources/dict/`
  - `/resources/models/`
  - `/resources/repl/` (not strictly necessary – a blank file will do)
  - `/resources/syll/` (only required for certain capabilities)
  - `/resources/vocab/`
- Make sure all files only have linefeeds (`LF`) and no carriage returns (`CR`) as end-of-line (EOL) characters

Demo

- In order to create the necessary resources, we first need to know what phones we will need to include in the model
- This will inform not only the model itself but also the so-called “dictionary” file
- Depending on your needs and the language in question, the “phones” used may be more phonologically- or phonetically-based
- You will also (probably) need some idea of the mapping between orthography (or romanisation etc.) and phones

(Standard) Yoruba phone(me) inventory

	Labial	Alveolar	Post-alveolar/ Palatal	Velar	Labial–velar	Glottal
Plosive	b	t d		k g	k͡p g͡b	
Affricate			d͡ʒ			
Fricative	f	s	ʃ			h
Glide			j		w	
Liquid		r l				
Nasal	m	n				

	Front	Back
High	i	u
High-mid	e	o
Low-mid	ɛ	ɔ
Low	a	

(+ nasalisation, vowel length and tone, albeit not all logical combinations are found)

(adapted from Prezdziecki 2005)

(Standard) Yoruba phone(me) inventory

	Labial	Alveolar	Post-alveolar/ Palatal	Velar	Labial–velar	Glottal
Plosive	b	t d		k g	p gb	
Affricate			j			
Fricative	f	s	ṣ			h
Glide			y		w	
Liquid		r l				
Nasal	m	n				

	Front		Back
High	i		u
High-mid	e		o
Low-mid	ẹ		ọ
Low		a	

(+ nasalisation, vowel length and tone, albeit not all logical combinations are found)

(adapted from Prezdziecki 2005)

(Standard) Yoruba phone(me) inventory

	Labial	Alveolar	Post-alveolar/ Palatal	Velar	Labial–velar	Glottal
Plosive	b	t d		k g	k_p g_b	
Affricate			dZ			
Fricative	f	s	S			h
Glide			j		w	
Liquid		r l				
Nasal	m	n				

	Front		Back
High	i		u
High-mid	e		o
Low-mid	E		O
Low		a	

(+ nasalisation, vowel length and tone, albeit not all logical combinations are found)

(adapted from Prezdziecki 2005)

./resources/models/models-yor/

- `config`, `wav_config` and `macros` can be copied and left unedited
- `monophones` should includes the list of phones found required for your language – in principle you can use any ASCII symbols but it's probably a good to broadly follow SPPAS (and you cannot use spaces between characters in a phone symbol) – this also includes symbols for fillers (see docs)
- `monophones.repl` should include replacement formulae between phones, for the most part, no changes are required – the filler symbol replacements can be left as is

`./resources/models/models-yor/`

- The next thing is a more tedious/work: `hmmdefs`
  - This is tells the aligner what kind of cues to look for for each phone in the audio being aligned and segmented
  - Each phone in the model requires a definition in this file
  - (And phones not in the model don’t need to be in `hmmdefs`)
  - (Some already existing models may have biphones but for our sake we’re not going to worry about them)

./resources/models/models-yor/

- Each phone definition begins `~h "…"` and is followed by the definition itself enclosed in the tag `<BeginHMM>` … `<EndHMM>`
- For a completely new phone (e.g. `k_p`), I'd recommend simply choosing the closest sound acoustically (e.g. `p`)
- Even if this does mean sometimes essentially having duplicate entires (e.g. `b` for `g_b` but keeping `b` too)
- You can pick and choose from the various languages already supported – the closer the fit, the better the alignment but really anything close will be of help

./resources/models/vocab/

- Next you will need to create a “vocabulary” file
- This should be a text file with the extension `.vocab`
- Include filler symbols e.g. `dummy`, `#`, `*`, `+`, `@@` on separate lines at the top of the file
- Then populate this with words from the new target language in orthographic/romanised form
- (This can be (partly) automated and built cumulatively)

./resources/models/dict/

- You will also need to create a “dictionary” file
- This is based on the vocab file with a phonetic transcription (using the phone-symbol based transcription scheme just developed)
- Add `[ ]` after the orthographic word then add the transcription
- Put spaces between each phone, e.g. `Abíkẹ́ [] a b i k E`
- (This too may be (partly) automated and built cumulatively)

Automated cumulative `.vocab` and `.dict` generation

- For languages with phonetically transparent orthographies, it is possibly to – at least in part – not only automate the building of but also cumulatively build the vocabulary and dictionary files
- This can be done with the help of an R script (or similar in your preferred programming language)

Get aligning!

- With all of the necessary resources created, now we’re ready to actually align our data with SPPAS
- Remember, the alignment requires not only the audio file but also a transcript – each “IPU” should be entered on a separate line
- Out the other end, you will get a series of TextGrids for use in Praat (other formats are available)
- Of these, `*-merge.TextGrid` is typically the most useful

More on aligning

- As well as aligning and segmenting everything in one fell swoop, it is possible to pre-prepare the IPU segmentation and *then* feedback into SPPAS
- This might be useful if you only want to look at a subset of words or if you're aligning "dirtier" recordings, e.g. from fieldwork
- The filler symbols such as `+` and `dummy` may also come in handy

Summary

- Today I’ve shown you that by making the most of existing tools we can achieve forced-alignment and segmentation of languages which are otherwise under-resourced
- Namely, this is possible – and not too difficult – with SPPAS
- You will have seen how this can be applied to Yoruba (at least pending further refinement)

Ẹ ṣeun!

‘Thank you!’