



Special thanks to
Kevin Scannell

UiT Norgga Árkálaš Universitehta
Romsa, Norway
francis.tyers@uit.no

Examples of code-switching segmentation

[en You're a] [**ga** Meiriceánach, cén fáth] [**en** are you] [**ga** foghlaim Gaeilge?!]
 @afaltomkins [**cy** gorfod cael bach o tan] [**en** though init]
en omg[**cy** mar cwn bach yn] [**en** black and tan] [**cy** a popeth,] [**en** even cuter!!]

- ## Example

Code switching: You're a [Meiriceánach, cén fáth] are you [foghlaim Gaeilge?!]
Quotations: The anthem starts with the words ['Mae hen wlad fy nhadau...']
Named entities: [Dr Jekyll] ha [Mr Hyde] embannet gant [Éditions Aber]
Interjections: Hey, that's great, [diolch yn fawr!]
Translations: Bloavezh mat d'an holl ! [Bonne anné à tous !]

Chunking algorithm

-
- Algorithm 1**
- *

Pair	Language	Statistics (%)	
		Tokens	Segments
Irish—English	Irish	332	40
	English	379	42
Welsh—English	Welsh	419	64
	English	378	66
Breton—French	Breton	388	54
	French	379	53

```

Require:  $s$  : sentence to chunk
1: buffer = [ ] /*Undecided expanding window of chunk*/
2: chunks = [ ] /*Decided labelled segment*/
3: buffer_language  $\leftarrow$  LANGPREDICT( $s[0]$ ) /* Language of first word */
4: flag  $\leftarrow$  0
5: for all  $w \in s$  do
6:   if LANGPREDICT( $w$ )=buffer_language then
7:     if flag = 1 then
8:       buffer  $\leftarrow$  buffer + [word_buffer, $w$ ]
9:       flag  $\leftarrow$  0
10:    else
11:      buffer  $\leftarrow$  buffer + [  $w$ ]
12:  if LANGPREDICT( $w$ )  $\neq$  buffer_language then
13:    if flag= 0 then
14:      flag  $\leftarrow$  1
15:      word_buffer  $\leftarrow$   $w$ 
16:      continue
17:    else
18:      chunks  $\leftarrow$  chunks + [(buffer,buffer_language)]
19:      buffer  $\leftarrow$  [word_buffer, $w$ ]
20:      buffer_language  $\leftarrow$  LANGPREDICT( $w$ )
21:      flag  $\leftarrow$  0
22:  if length(buffer)  $\neq$  0 then
23:    chunks  $\leftarrow$  chunks + [(buffer,buffer_language)]

```

Results

System		Irish—English		Welsh—English		Breton—French	
		Irish	English	Welsh	English	Breton	French
baseline	p	2.50	0.0	0.0	0.0	0.0	0.0
	r	2.56	0.0	0.0	0.0	0.0	0.0
<i>langid</i> -3character	p	5.00	14.29	0.0	21.21	1.85	20.75
	r	5.41	8.45	0.0	14.58	1.92	12.36
wordlist	p	32.50	28.57	26.69	40.91	57.41	33.96
	r	23.64	26.09	26.03	33.75	47.69	33.33
character bigram	p	32.50	35.71	23.44	19.70	57.41	52.83
	r	22.41	26.79	15.31	16.67	41.33	37.84
wordlist+character bigram	p	52.50	50.00	32.81	31.82	70.37	67.92
	r	38.18	43.75	24.14	25.61	57.58	57.14

- | System | Accuracy (%) | | |
|---------------------------|---------------|---------------|---------------|
| | Irish—English | Welsh—English | Breton—French |
| baseline | 42.76 | 42.16 | 44.07 |
| <i>langid</i> -3character | 57.24 | 45.92 | 43.16 |
| wordlist | 79.75 | 74.28 | 83.96 |
| character bigram | 81.29 | 65.62 | 76.79 |
| wordlist+character bigram | 85.79 | 72.40 | 88.79 |

..... **Word based prediction**

- Word are labeled according to their presence in the particular word list.

- In case of a confusion the word is added to the previous segment

.. Word-based prediction with character backoff ..

- Same as Word-based prediction, but in case of confusion this falls back to the Alphabet bi-gram approach.

.....**Baseline**.....

- Using *languid.py* labeled all the lines in a particular dataset according to the majority classification

.....*Langid* character trigram prediction.....

- Trigram probabilities from *langid* were taken into account.

- All other heuristics and chunking algorithm are same as for other methods

Conclusions

- We followed the footsteps of **CoNLL 2000** shared task on language independent named entity recognition.

- Divide the text into non-overlapping segments.

- **Precision** - percentage of correctly detected phrases.

- **Recall** - number of phrases in the data that were found by the chunker.

- A very preliminary investigation into subsegment language identification in Celtic language texts.

- We would like to include supervised methods and features talked about by King and Abney (2013)

- We would also like to check our methods with higher order n-grams and more options in backoff.

- Explore a lattice technique where each word is a lattice node and the inclusions of the words are done using probability.