UKDW

Universitas
Kristen Duta
Wacana

Dr. phil. Lucia D. Krisnawati

Word Processing:

Syllabifier & Language Identifier

04 September 2017

Outline

- Recap
- motivations
- Syllabification
- Language Identifier

Recap

Text normallization processes:

- Tokenization:
  - The task of chopping a given text into pieces called *token*.
  - But | what | are | thoughts | well | we | are | all | have | them
- Stemming: a crude heuristic process that chops off the end of words in the hope of achieving the base form, and often includes the removal of derivational affixes eg:
  - menyapu, disapu, tersapu, sapuan → sapu but got nyapu, sapu
- Lemmatization: the process of removing inflectional ending properly with the use of a vocabulary and morphological analysis of words, to remove inflectional endings and to return the base or dictionary form of a word, eg
  - Coming, came, come, comes, cometh → come
  - menyapu, disapu, tersapu, sapuan → sapu

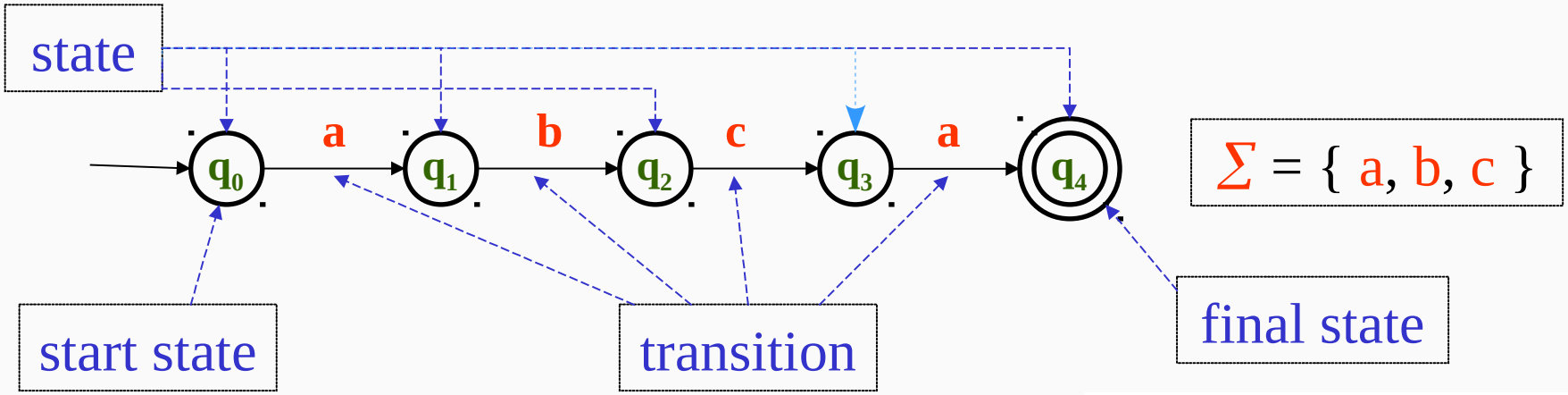
Recap

Regular Expression:

Let Σ be an alphabet. The regular Expression over the alphabet Σ is defined recursively as follows:

- Ø and ε are regular expressions
- A is a regular expression for each a ∈ Σ,
- If r and s are regular expreessions, then so are r ∪ s , r · s, and r*
- No other sequences of symbols are regular expressions

Recap



$\Sigma = \{ a, b, c \}$

State	Input a	b	c
0	1	⊘	⊘
1	⊘	2	⊘
2	⊘	⊘	3
3	4	⊘	⊘
4	⊘	⊘	⊘

$S = q_0$ $F = \{q_5, q_8\}$

Transition function $\delta: \Phi \times \Sigma \rightarrow \Phi$

$\delta(q_0,c)=q_1$

$\delta(q_0,e)=q_3$

$\delta(q_0,l)=q_6$

$\delta(q_1,l)=q_2$

$\delta(q_2,e)=q_3$

$\delta(q_3,a)=q_4$

$\delta(q_3,v)=q_9$

$\delta(q_4,r)=q_5$

$\delta(q_6,e)=q_7$

$\delta(q_7,t)=q_8$

$\delta(q_8,t)=q_9$

$\delta(q_9,e)=q_4$

Exercise

On-Hand exercise:

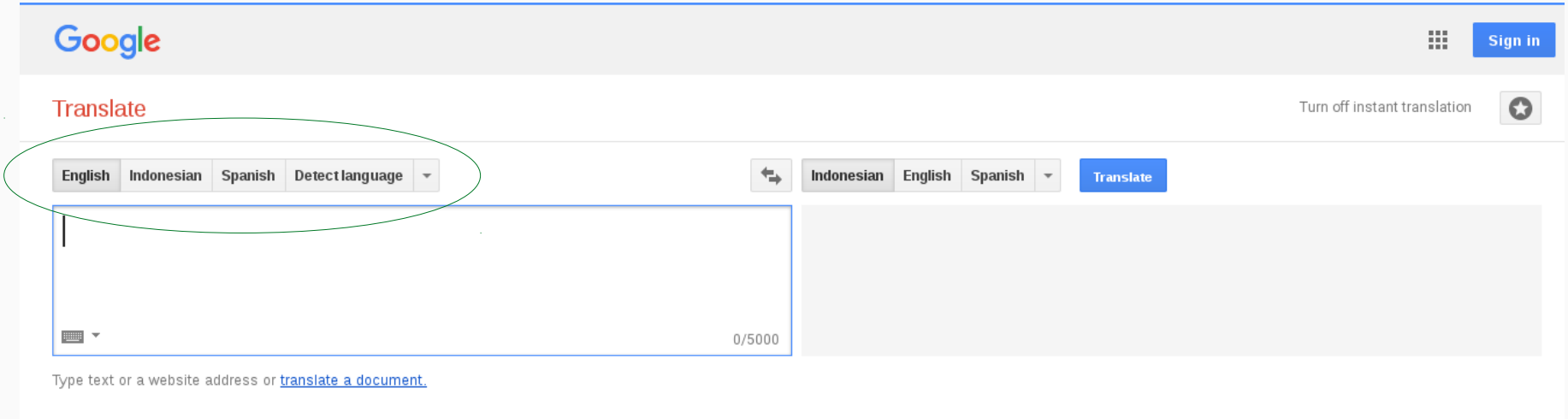
- Write an FSA for time-of-day expression like ‘eleven o’clock’, twelve-thirty or a quarter to ten

Motivation for Syllabification

- A preprocessing process in a more complex applications:
  - Speech processing: speech recognition & synthesis
  - Predicting word:
    - Identifying new names (Warintarawej et al, 2011) created by industries & businesses
    - Spelling correction
    - Transliteration
  - Information extraction
  - An approach in language identifier

Motivation for Language Identification

- Human-computer interaction in natural language
- A preprocessing step in:
  - Information retrieval, text mining
  - Machine translation, email interception
  - Speech processing, dialogue system
  - Plagiarism & text reuse detection, Text Forensics



Syllabification

A syllable is

- a unit of sounds composed of a vowel and/or the consonants that cluster around this central peak
  - ba + pak → bapak I + bu → ibu
- typically made up of a syllable nucleus (most often a vowel) with optional initial and final margins (typically, consonants).

A syllabification is:

- A process of segmenting a word or token into syllables
  - bapak → ba – pak (ID)
  - Mbledos → mble – dos (JV)
  - Scream → scream (EN)

Syllabification

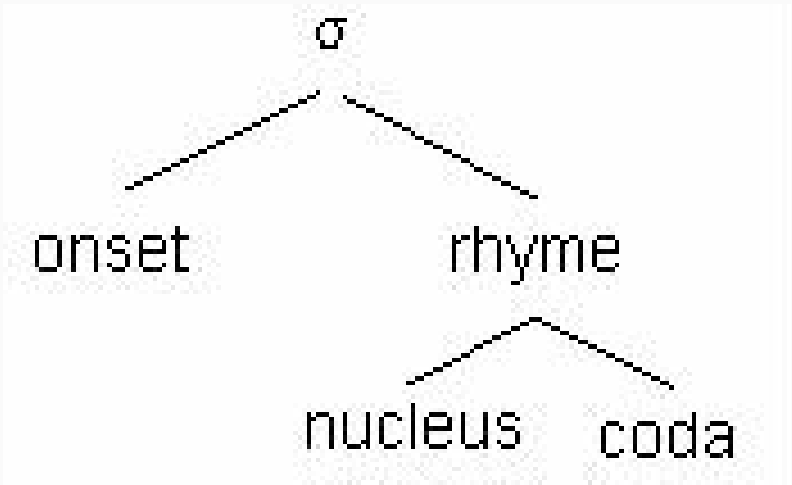
Syllable structure

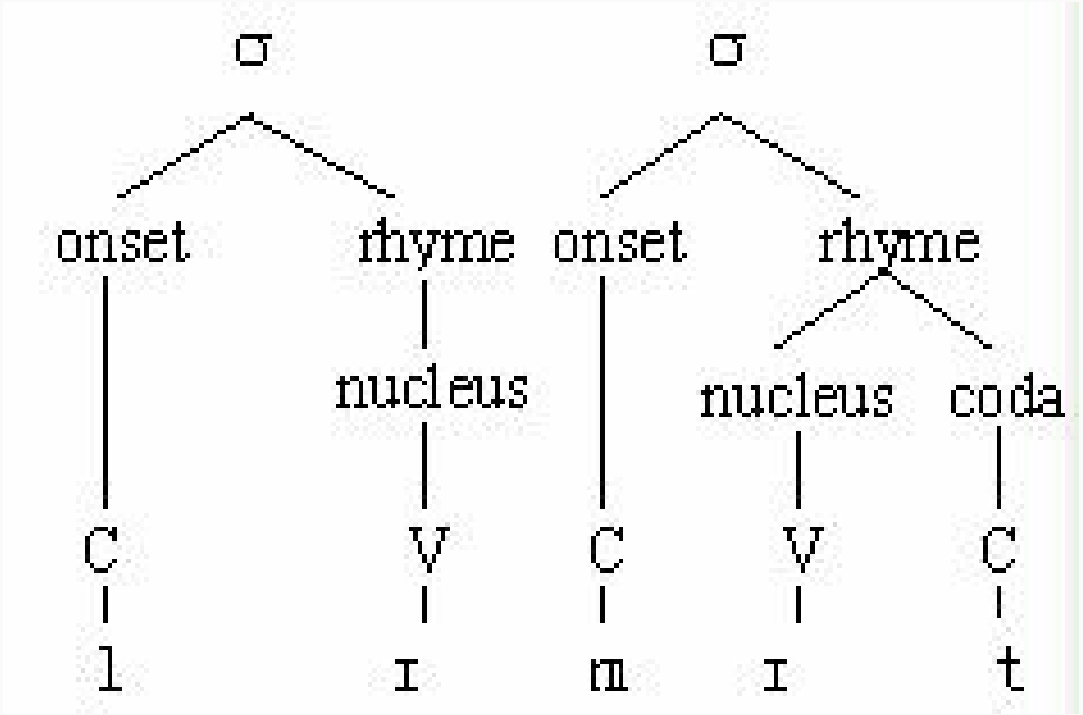
- Is the combination of allowable segments and typical sound sequences
- Is language specific
- The syllable structure comprises of Onset and Ryme

Parts	Description	Optionality
Onset	Initial segment of a syllable	optional
Ryme: - nucleus - coda	- Central segment of a Syllable - closing segment of a syllable	obligatory

Syllabification

Syllable structure:





Syllabification

Points to Ponder

- Look at the theory on syllable structure
- Discuss it with your friend whether it is applicable to:
  - Bahasa Indonesia
  - Any vernacular languages in Indonesian archipelago, eg. Javanese, Balinese, etc
- Is there any drawback of this theory?

Syllabification

Phonotatics

- A field which studies the rules of combining phones into a larger phonological segment – a syllable
- Indonesian has in total 11 syllable structures comprising of 4 basic structures and 7 variations of the former
- Indonesian syllable structures (Kushartanti et. Al, 2007):
  - Basic: V, VK, KV, KVK
  - Variation: VKK, KKV, KKVK, KVKK, KKVKK, KKKV, KKKVK
- Javanese Syllable has 8 structures (Wedhawati, et al, 2006):
  - V, VK, KV, KVK, KKV, KKVK, KKKV, KKKVK
  - KKVKK : **nyang**-king, me-**nyang**, **ndong**-kel, ke-cem-**plung**
  - Both Javanese & English have syllable structure begun with 3 consonants – KKKV-(k) but their consonant patters are different
    - Javanese starts with nasal consonants /ŋ/:: **ndre**-mi-mil, **mbya**-ya-ki
    - English starts with /s/ : **spray**, **squad**-ron, **split**

Syllabifier

Study Case I

- Source: Arifin & Tiun. Predicting Malay Prominent Syllable
- Motivation:
  - The lack of high quality in speech synthesis, a.k.a Text-To-Speech (TTS), in managing the prosodic speech elements such as intonation, accentuation, prosodic phrases, rhythm, hesitation, etc
- Goal:
  - Predicting the prominent syllable for improving the prosody elements in TTS
  - Classifying the speech syllables into prominent and non-prominent syllable using Support Vector Machine (SVM)

Study Case I

- Data/Documentation
  - A UMTK-MSS corpus consisting 400 sentences in Malay syntax-prosodic speech text
  - Select 50 sentences randomly from the corpus:
    - Learning model: 30 sentences
    - Test data: 20 sentences
  - The native speaker → Malay native speaker from Selangor
  - Corpus specification : Selangor dialect, educated background
  - Corpus specification & annotation were concerned to support the quality speech corpus

Study Case I

- Feature Selection:
  - Feature I: **Part of Speech (POS)**
    - using open source POS tagger written by Tan Tien Ping
    - Tag 8 classes: N, V, Adj, Adv, Pron, prep, conj, Interjection
  - Feature II: **Syllable types**
    - Defining 4 syllable types with their values

Syllable Type	Value
C	1
V	2
CV	3
VC	4

  - Feature III: **Syllable length**
    - Syllable length is measured by calculating each character in the syllable

Study Case I

- Feature IV: Syllable position
  - Coding the syllable position in a word
  - eg. makan → ma-kan, 2 positions only that are start & end
  - Defining 3 positions only: start, middle, end
  - Konverting this position into an integer value

Syllable Position	Value
Start	1
Middle	2
Final	3

Study Case I

- Feature V: **Word Position**
  - Looking at the index of a word in the whole text
  - The value of word position is marked after the phrasal break or the end of token
  - Defining only 3 positions : start, middle, final

Word Position	Value
Start	1
Middle	2
Final	3

Syllabifier

Study Case I

- Finding the prominent syllable by the way of classification
- Classifier applied:
  - Support Vector Machine (SVM)
  - Naive Bayes
- SVM data format:
  - Classifier is run, the data should be annotated
  - The format : label, index, index value
  - The label refers to the classes defined

[Label] [index1]: [value1] [index2]: [value2]....
[Label] [index1]: [value1] [index2]: [value2]....

Input sentence: "Saya *makan nasi"
Syllabification: sa.ya ma.*kan na.si

Study Case I

- SVM data format

Prominent/ Non-Prominent	Feature 1	Value	Feature 2	Value	Feature 3	Value	Feature 4	Value	Feature 5	Value	Info (#)
-1	1	1	2	3	3	2	4	1	5	1	sa
-1	1	1	2	3	3	2	4	3	5	1	ya
-1	1	3	2	3	3	2	4	1	5	2	ma
+1	1	2	2	3.1	3	3	4	3	5	2	kan
-1	1	2	2	3	3	2	4	1	5	3	na
-1	1	2	2	3	3	2	4	3	5	3	si

(Arifin & Tiun, 2013)

Study Case I

- Results

Features	Support Vector Machine (SVM)	Naïve Bayes (NB)
Part-Of-Speech	89.1583 %	Naïve Bayes (NB)
Syllable Type	89.0157 %	88.4451 %
Syllable Length	89.0157 %	88.4451 %
Syllable Position	89.0157 %	88.4451 %
Word Position	89.0157 %	88.4451 %

(Arifin & Tiun, 2013)

Syllabifier

Ruled-based syllabifier:

- The example rules:
  - If 2 consonants come between two vowels in a word, divide syllables between the consonants. eg.
    - KVKKV^{+} → KVK-KV
    - coffee → cof-fee
    - Kampung → kam-pung
  - If there is one consonant between two vowels in a word, divide the syllables after the first vowel. eg. bonus → bo-nus
    - KVKVK → KV-KVK
    - bonus → bo-nus

Syllabifier

Rule-based syllabifier:

- Breaking up and defining words in accordance to the defined rules
  - Suppose we have rules as shown in the table below, how would you syllabify the given word patterns?
  - How do you construct rules for your syllabifier to solve the input of : pa-ktu-ba-gus, se-pa-kabo-la?

Word patterns/ words	Rules for syllabifier
Kvkkvk/keplak	If 2 consonants come between two vowels in a word, divide syllables between the consonants
kvkvkvk/kudanan	If there is 1 consonant between two vowels in a word, divide the syllables after the first vowel.
kvvkvkkv/soeharto	If there are 2 vowels together that don't represent a long vowel sound or a diphthong, divide the syllables between the vowels.

Language Identifier

Introduction

- Language identifier is designed to focus on:
  - Detecting monolingual document
  - Detecting multilingual documents
- The approaches for language identifier falls into 2 categories
  - Statistical approaches
    - Use probability models to predict the language
    - Require corpora in specific language(s) as the basis of inference
  - Non-statistical approaches
    - depend on linguistic aspects for the knowledge base
    - Need sufficient knowledge on symbols such as diacritics and symbols

      eg. diacritics: ä, ê, ñ, è

      symbols: ß, Æ, β, γ, σ , Ж, Д

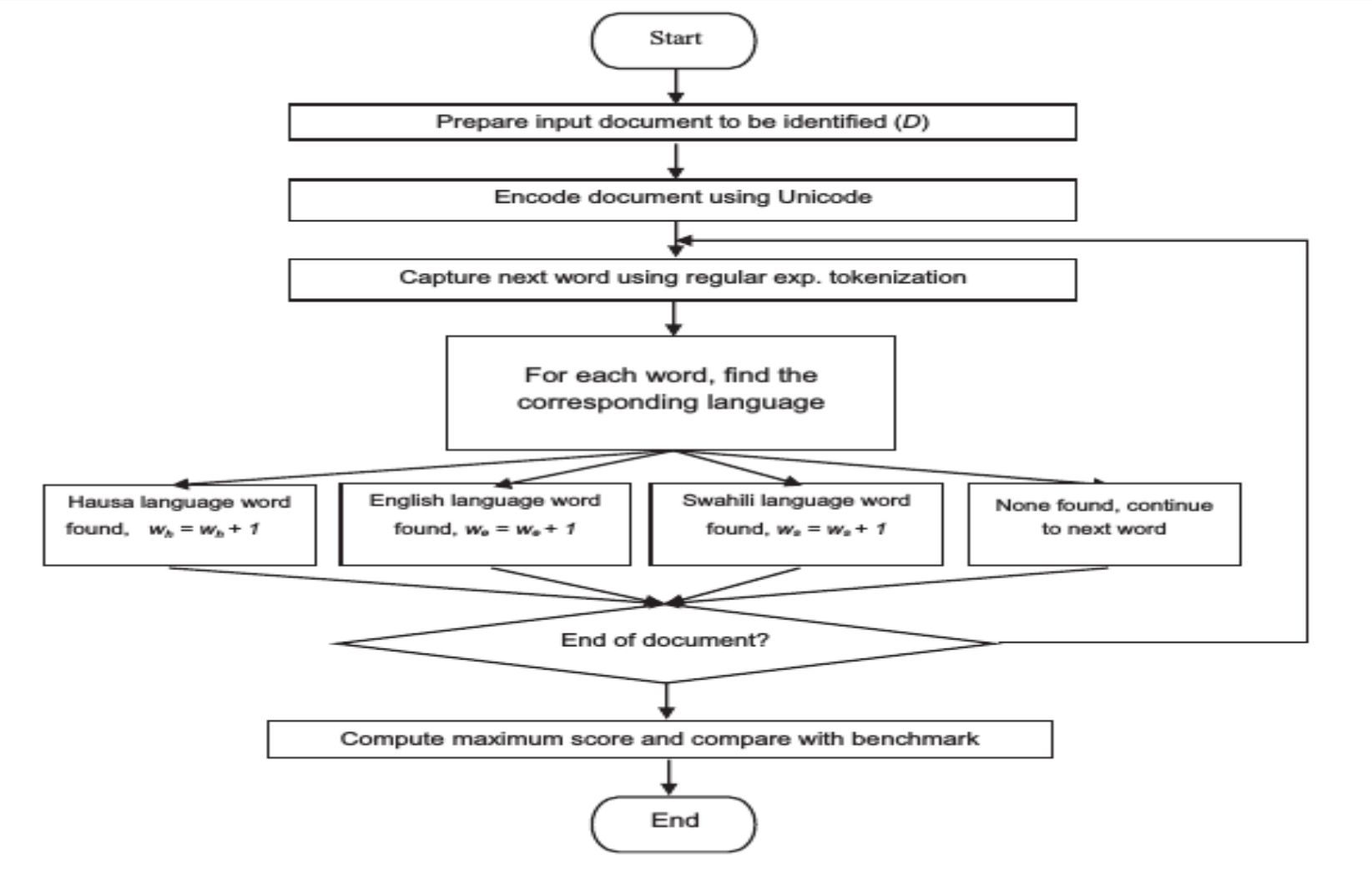
Non-statistical Approaches

- Some of non-statistical approaches
  - Lexicon-based language identification
  - Rule-based language identification
- Lexicon-based language identification(LBLI)
  - Lexicon is a list of vocabulary of a language or a branch of knowledge
  - The main principle of LBLI is to identify a language of a text if it contains most words of a lexicon from a specific language
  - LBLI needs a lexicon(s) from specific language
    - English lexicon, Javanese lexicon, Indonesian lexicon
  - The search : $Lg_w = \{w | w \in V \cap D(w)\}$

Study Case II

- Selamat & Akosu (2016). Word-length algorithm for language identification of under-resourced languages
- Method: lexicon-based
  - Identifying 3 languages: English, Hausa, & Swahili
  - Generating the language profiles:
    - Building a corpus for each language
    - For each document in a language do:
      - Tokenization
      - Eliminating duplicate words → a list of unique tokens
      - The token will be inputted into spelling checker
  - The search: $Lg_w = \{w | w \in V \cap D(w)\}$

Study Case II: The flowchart of LBLI



(Selamat & Akuso, 2016)

Study Case II

- The algorithm of LBLI (Selamat & Akosu):
  - Step 1: Input training texts and test texts.
  - Step 2: Generate the training profiles and test profiles.
  - Step 3: Compute binary matrix for test profile using all training profiles.
  - Step 4: With the binary matrix as input, determine the highest score using the training profiles.
  - Step 5: If the highest training profile's score is greater or equal to the benchmark set by the user, then that determines the language of the test profile. Otherwise the language of the test profile is unknown.

Study Case II

- Computing the binary matrix
  - The binary matrix is based on Boolean method:
    - 1 if the word in test profile is found in lexicon & training profile
    - 0 if the word in test profile could not be found

$$\text{Score}_D = \begin{cases} wh = wh + 1 \text{ if selected Hausa word is found} \\ we = we + 1 \text{ if selected English word is found} \\ ws = ws + 1 \text{ if selected Swahili word is found} \\ \text{Continue if none is found} \end{cases}$$

  - Wh is the weight for Hause, we → English, & ws → Swahili
- The desired output is computed:

$$\text{SPLID} = \max\left(\sum_{i=1}^{n} wh_i, \sum_{i=1}^{n} we_i, \sum_{i=1}^{n} ws_i\right).$$

Study Case II

- Lexicon-based model with word length statistics
  - Is used to reduce the computation time of LBLI
  - In LBLI 1 token in a test doc is compared to all tokens in training set
    - Wasting time & computer resource
  - Improvement:
    - Each word in test doc is measured its length and compared to the words in training set with the same length only.
    - V=vocabulary, D=dataset, n= number of words in D

$$\text{Score} = \begin{cases} 1 & \text{if } (w \in V \text{ and } D(w)) \\ 0 & \text{otherwise} \end{cases}, \quad \text{Score } (L:D) = \sum_{i=1}^{n} \text{score}(X_i) \quad \text{Score } (\%) = 100 * \sum_{i=1}^{n} \text{score}(x_i)/n$$

Statistical Approach

- Study Case III
  - Source: Takçi & Ekinci, 2012
  - Identifying 9 European languages including Turkey
  - The feature set is built from 26 letters & diacritics

‘A’, ‘À’, ‘Á’, ‘Â’, ‘Ã’, ‘Ä’, ‘Å’, ‘Æ’, ‘B’ , ‘C’, ‘Ç’, ‘D’, ‘E’, ‘È’, ‘É’, ‘Ê’, ‘Ë’, ‘F’, ‘G’, ‘Ğ’, ‘H’, ‘İ’, ‘I’, ‘Ì’, ‘Í’, ‘Î’, ‘Ï’, ‘J’, ‘K’, ‘L’, ‘M’, ‘N’, ‘Ñ’, ‘O’, ‘Ò’, ‘Ó’, ‘Ô’, ‘Õ’, ‘Ö’, ‘P’, ‘Q’, ‘R’, ‘S’, ‘Ş’, ‘T’, ‘U’, ‘Ù’, ‘Ú’, ‘Û’, ‘Ü’, ‘V’, ‘W’, ‘X’, ‘Y’, ‘Ÿ’, ‘ß’, ‘Ø’, ‘Z’, ‘’’’

  - Corpus: European Corpus Initiative(ECI) → multilingual corpus
  - Text normalization: eliminating punctuation marks, spaces, digits
  - Preprocessing: segmenting text into smaller chunks: 30, 60, 90, 100, 120, 180 & 500 characters
  - For each size, 200 texts are prepared

Study Case II: Statistical Approach

- Weighting vector
  - Each text is transformed into a list of letters
  - Compute the frequency of each letter in documents of a specific language
  - Mathematical notation used:
    - i → language index j → document index
    - k → letter index l → #language
    - n_i → #training documents in ith language
    - m → # letters
    - c_i →# letters weight factor vector for ith language
    - $_{cik}$ → weight factor for k^{th} letter in i^{th} language
    - fr_{ij} → frequency vector for j^{th} doc in i^{th} language
    - d_{ij} → weighted frequency vector for j^{th} doc in i^{th} language

Study Case II: Statistical Approach

- A single letter is weighted using equation 1:

$$c_{ik} = \log\left(\frac{\sum_{j=1..n_i} fr_{jk}}{\sum_{i=1..l}\sum_{j=1..n_i} fr_{ijk}} * 100\right) \quad [1]$$

$$c_i = (c_{i1}, c_{i2}, \ldots, c_{im})$$

- Transformed valued are obtained with equation 3

$$d_{ij} = fr_{ij} x\ c_i \quad [3]$$

- The next phase is classification usingKernel-Support Vector Machine (SVM), Multiplayer Perceptron (MLP), & Linear Discrimionant Analysis (LDA)

References

Arifin, N.A., & Tiun, S. (2013). Predicting Malay Prominent Syllable, In *Prosedia technology II*. Elsevier, pp. 861-869;

Kushartanti, Yuwono, U., & Lauder, M.R. (2007). P*esona Bahasa: Langkah Awal memahami Linguistik*. Jakarta: Gramedia.

Selamat, A., & Akuso, N. (2016). Word-length algorithm for language identification of under-resourced languages. In Journal of King Saud University – Computer and Information Sciences, vol 28, pp. 457-469

Takçi, H., & Ekinci, E. (2012) Minimal feature set in language identification and finding suitable classification method with it, in Procedis Technology 1, Elsevier, 444-448

Warintarawej, P., Laurent, A., Pompidor, P., Cassanas, A., & Laurent, B. (2011). *Classifying Words: A Syllables-based Model rnational Workshop on Text-based Information Retrieval – TIR,* DEXA 2011.

Wedhawati, et.al. 2006. *Tata Bahasa Jawa Mutakhir*. Yogyakarta: Penerbit Kanisius.