Transforming Semi-Structured Indigenous Dictionary into Machine-Readable Dictionary

Bali Ranaivo-Malançon[1], Suhaila Saee[1,2], Rosita Mohamed Othman[1] and Jennifer Fiona Wilfred Busu[1]
[1]Faculty of Computer Sciences and Information Technology,
[2]Institute of Social Informatics and Technological Innovations,
Universiti Malaysia Sarawak.
ssuhaila@unimas.my

***Abstract*—Creating a machine-readable dictionary for an indigenous language is not an easy process and thus, transforming an existing indigenous dictionary into a machine-readable dictionary is one approach to speed up the process. This paper presents the sequential transformation of two bilingual Sarawak indigenous dictionaries, Melanau-Mukah-Malay and Iban-Malay dictionaries, from their initial semi-structured form into their structured representation. The transformation makes use of an OCR to convert the original PDF format of the dictionaries into HTML files, which is then analysed by the Python HTMLParser to extract only the content of the dictionaries. The extracted content is saved in plain text file. To understand the original structure of each dictionary, the textual units in the plain text file are converted into generic symbols. The observation of the collocations of the generic symbols yields to the writing of regular expressions that can delimit each dictionary element. The result is a machine-readable dictionary stored in comma-separated values format. The inspection of each column in the comma-separated values file indicates that the written regular expressions offer a good coverage of the different dictionary elements present in the studied dictionaries. Therefore, the proposed sequential transformation is efficient in accomplishing the conversion of a semi-structured indigenous dictionary into a structured machine-readable dictionary.**

***Index Terms*—Machine-Readable Dictionary; Semi-Structured Data; Indigenous Dictionary; Regular Expressions; Python.**

I. Introduction

Creating a machine-readable dictionary (MRD) for an indigenous language is a long-term process requiring a large amount of lexical knowledge, human resource, and a sufficient financial support. Thus, transforming an existing indigenous dictionary into an MRD is one approach to speed up the process. This paper presents the different steps needed for the transformation.

Numerous bilingual dictionaries of indigenous languages of Sarawak exist as either in printed or electronic form, which is usually the electronic version of the printed one. Bilingual MRDs are useful for many applications such as word-processing assistance, machine translation, generating parallel corpora [1], etc. However, transforming the electronic version of a printed dictionary into an MRD is not straightforward and can be challenging. The transformation process requires a good strategy to be re-usable for any similar type of dictionaries, with a minimum cost of processing and human intervention. To illustrate the application of the proposed transformation method, two bilingual Sarawak indigenous dictionaries, Melanau-Mukah-Malay and Iban-Malay dictionaries, are used as the case-study. "A bilingual dictionary consists of an alphabetical list of words or expressions in one language (the 'source language') for which, ideally, exact equivalents are given in another language (the 'target language')." [2]. Hence, the source languages are Melanau-Mukah and Iban and the target language is Malay Standard. The input dictionaries are in PDF (Portable Document Format) and their content is semi-structured that makes their transformation difficult. The target dictionaries (MRDs) are structured and stored in CSV (Comma-Separated Values) format. The proposed transformation process consists of converting the PDF file into a plain text file for programming language purpose, and then at identifying automatically the different textual units corresponding to dictionary elements for final storage. Since each indigenous dictionary has its own structure and information, the identification of the dictionary elements is the only step that is dictionary-dependent. The other steps (transformation of PDF into plain text and transformation of annotated plain text into CSV) are totally generic. In general, a dictionary has four types of structures. The megastructure concerns the entire structure of the dictionary. The macrostructure relates to the organisation of the dictionary (or lexical) entries. The microstructure concerns the consistent organisation of lexical information within lexical entries. The mesostructure refers to the set of relations that exist between lexical entries. To be able to identify each of these structures as well as the sub-structures, a rigorous and systematic method needs to be determined. The dictionary elements that need to be identified are the elements of the microstructure. Usually, they correspond to the headword, the pronunciation, the part of speech (POS) tag, and the various senses. However, other information can be available such as the etymology, examples, hyphenation, translations in a target language, etc.

II. Related Work

The creation of a dictionary for indigenous, under-resourced, and endangered languages is part of their documentation. Today, with the spread of computer and smart devices, a printed dictionary is no more attractive for end-users, who are more interested in accessing information in a fast way. Thus, a dictionary needs to be in a digital form and the content needs to be accessible through "intelligent" search such as searching for all lexical entries sharing the same root (e.g. read, reader, reading, etc.) or belonging to a specific POS like verb. An MRD can offer the answers to these queries. However, creating an MRD from scratch is a long process, and thus some researchers started to convert the electronic versions of printed dictionaries into MRDs.

A bilingual dictionary (English-Slovene) published by DZS and stored in SGML (Standard Generalised Markup Language) was converted into a TEI SGML dictionary format [3]. SGML is a standard set of codes for marking boldface, italics, etc., in ASCII text files. TEI (Text Encoding Initiative) provides standard SGML-based formats for various types of texts including dictionary data. The authors used Omnimark LE, the light version of Omnimark, a conversion programming language for SGML. To perform the conversion from DZS SGML DTD into TEI.dictionary DTD, 44 actions were written [3].

A script language called ABET (APL Bidict Extraction Tool) based on APL programming language has been developed to convert online bilingual dictionaries to MRDs [4]. The extraction comprises four steps: text pre-processing, header and body identification, post-processing, and recursive header and body parsing. As stated by the authors, “the longest ABET extraction script to date was a mere twenty-four lines”, which “matches the data so well”.

Balabanova and Ivanova [5] presented a method for transforming the Bulgarian valence dictionary into an XML (Extensible Markup Language) electronic version. “The electronic version has been built by restructuring the lexical entries into an HPSG style of information representation and by defining the document’s metalanguage.” [5]. HPSG (Head-Driven Phrase Structure Grammar) is a grammar formalism used in natural language processing for parsing the syntax of a sentence.

Apertium is a free and open-source platform for the development of rule-based machine translation systems. As such, the platform requires bilingual and multilingual dictionaries. To make the translations available on the Web, 22 Apertium bilingual dictionaries and lexicons have been converted into RDF (Resource Description Framework) [6]. RDF is the standard model for data interchange on the Web. The conversion goes through a few steps: analysis of the data, selection of relevant vocabularies, modelling, URIs design, generation, linking, and publication. URI stands for Uniform Resource Identifier, and it is a string of characters used to identify a resource on the Web. The authors indicated that “all the converted dictionaries (many of them covering under-resourced languages) are connected among them and can be easily traversed from one to another to obtain, for instance, translations between language pairs not originally connected in any of the original dictionaries.” [6].

The keen interest in endangered languages has brought Maxwell and Bills [7] to also consider the conversion of OCRed dictionaries into structured dictionaries. This recent work is very similar to the work reported in this paper: the same objective and the same programming language (Python). However, their proposed approach implies the intervention of human to correct the recognition of the OCR. While their separation of the different dictionary elements goes from the identification of lexical entry boundaries first before applying a finite state grammar to parse the internal structure of each lexical entry. The work is still in progress and no real result has been reported.

III. TRANSFORMATION INTO MRD

The transformation of the original PDF dictionary into structured CSV dictionary (i.e. the final MRD) goes through a series of processing. It starts from the textual extraction of the content of the dictionary, followed by the text analysis and annotation of the dictionary fields, and to finally store the annotated data into a structured format as illustrated in Figure 1.

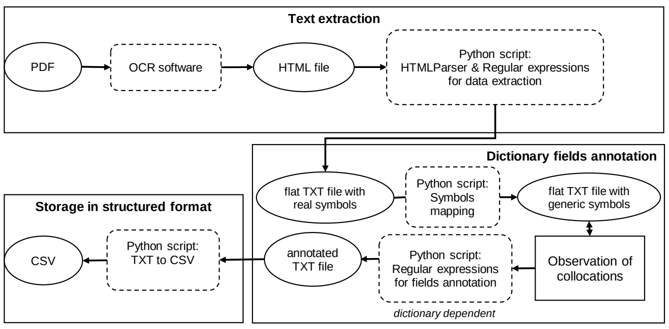

Text extraction
PDF
OCR software
HTML file
Python script: HTMLParser & Regular expressions for data extraction
Dictionary fields annotation
flat TXT file with real symbols
Python script: Symbols mapping
flat TXT file with generic symbols
Observation of collocations
Python script: Regular expressions for fields annotation
annotated TXT file
dictionary dependent
Storage in structured format
Python script: TXT to CSV
CSV

Figure 1: The sequential transformation

A. *Motivation: irregular structure and formatting in dictionaries*

A detailed examination of the two studied indigenous dictionaries reveals that they present irregularities in their structure and formatting. Therefore, a direct transformation of the dictionaries will certainly not generate the expected MRDs. The Melanau-Mukah dictionary presents its content in two columns (Figure 2) whereas the Iban dictionary presents it in eight columns per page (Figure 3).

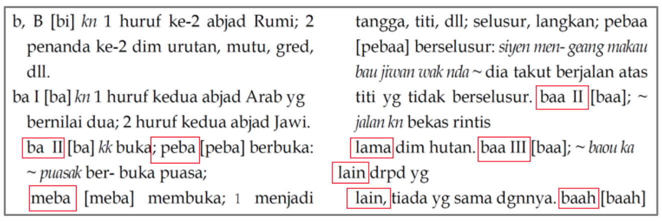
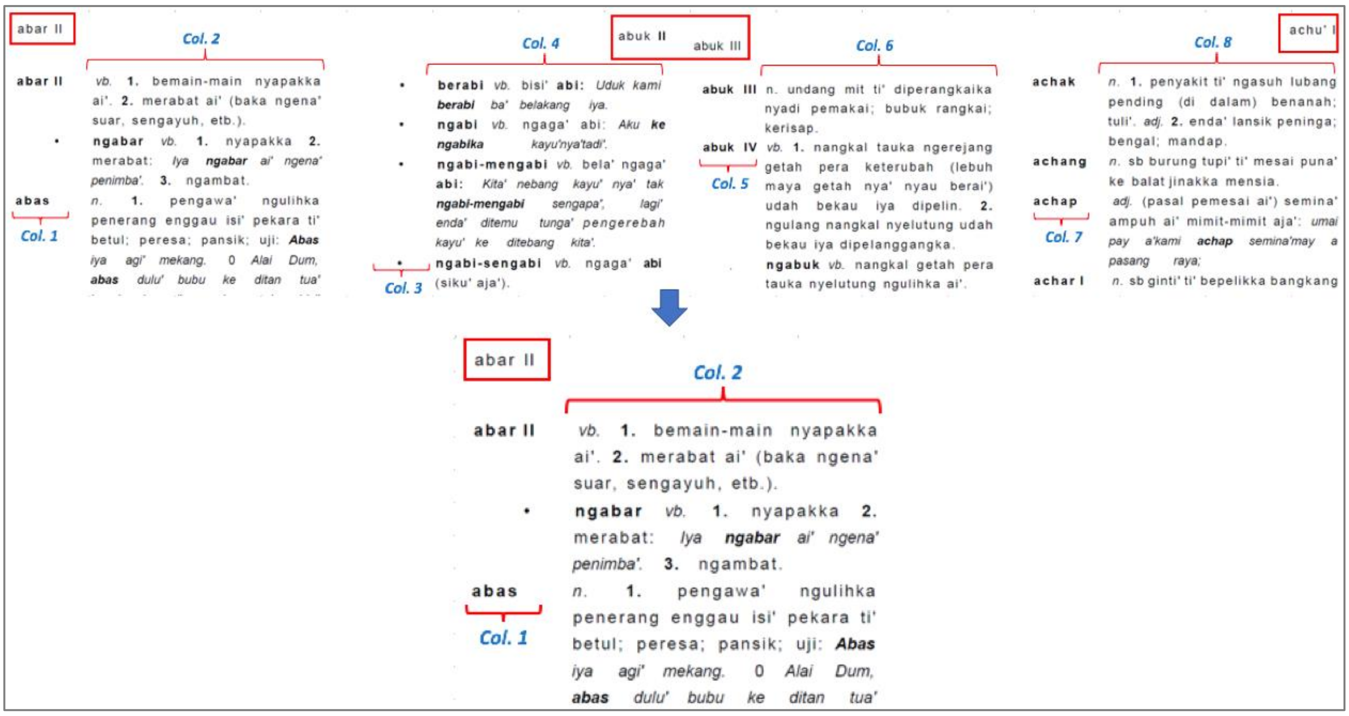

b, B [bi] *kn* 1 huruf ke-2 abjad Rumi; 2 penanda ke-2 dim urutan, mutu, gred, dll.
ba I [ba] *kn* 1 huruf kedua abjad Arab yg bernilai dua; 2 huruf kedua abjad Jawi.
ba II [ba] *kk* buka; peba [peba] berbuka: ~ *puasak* ber- buka puasa;
meba [meba] membuka; 1 menjadi tangga, titi, dll; selusur, langkan; pebaa [pebaa] berselusur: *siyen men- geang makau bau jiwan wak nda* ~ dia takut berjalan atas titi yg tidak berselusur. baa II [baa]; ~ *jalan kn* bekas rintis
lama dim hutan. baa III [baa]; ~ *baou ka*
lain drpd yg
lain, tiada yg sama dgnnya. baah [baah]

Figure 2: An excerpt of Melanau-Mukah dictionary

In Figure 2, from a very small excerpt of the Melanau-Mukah dictionary, nine irregularities (indicated by rectangles) can be found. The entry “ba II” does not have the same alignment as the entry “b, B”. This inconsistency of alignment can be seen again with the line “lain drpd yg”, which is aligned like a headword. The derived words, which are the sub-entries of a headword, are not formatted in the same way. The derived word “peba [peba]” is at the same line as the headword “ba II” and it is introduced after a semi-column “;”. However, the other derived word “meba [meba]” is in a newline. Not all headwords have the same position. The headwords “baa II”, “baa III”, and “baah” are positioned inside the description of the previous headword.


abar II
Col. 2
Col. 1
Col. 4
Col. 3
abuk II
abuk III
Col. 6
Col. 5
Col. 8
achu' I
Col. 7

abar II *vb.* **1.** bemain-main nyapakka ai'. **2.** merabat ai' (baka ngena' suar, sengayuh, etb.).
• **ngabar** *vb.* **1.** nyapakka **2.** merabat: *Iya* ***ngabar*** *ai' ngena' penimba'.* **3.** ngambat.
abas *n.* **1.** pengawa' ngulihka penerang enggau isi' pekara ti' betul; peresa; pansik; uji: ***Abas*** *iya agi' mekang.* 0 *Alai Dum,* ***abas*** *dulu' bubu ke ditan tua'*

• **berabi** *vb.* bisi' **abi:** *Uduk kami* ***berabi*** *ba' belakang iya.*
• **ngabi** *vb.* ngaga' abi: *Aku* ***ke ngabika*** *kayu'nya'tadi'.*
• **ngabi-mengabi** *vb.* bela' ngaga' **abi:** *Kita' nebang kayu' nya' tak* ***ngabi-mengabi*** *sengapa', lagi' enda' ditemu tunga' pengerebah kayu' ke ditebang kita'.*
• **ngabi-sengabi** *vb.* ngaga' **abi** (siku' aja').

abuk III n. undang mit ti' diperangkaika nyadi pemakai; bubuk rangkai; kerisap.
abuk IV *vb.* **1.** nangkal tauka ngerejang getah pera keterubah (lebuh maya getah nya' nyau berai') udah bekau iya dipelin. **2.** ngulang nangkal nyelutung udah bekau iya dipelanggangka.
ngabuk *vb.* nangkal getah pera tauka nyelutung ngulihka ai'.

achak *n.* **1.** penyakit ti' ngasuh lubang pending (di dalam) benanah; tuli'. *adj.* **2.** enda' lansik peninga; bengal; mandap.
achang *n.* sb burung tupi' ti' mesai puna' ke balat jinakka mensia.
achap *adj.* (pasal pemesai ai') semina' ampuh ai' mimit-mimit aja': *umai pay a'kami* ***achap*** *semina'may a pasang raya;*
achar I *n.* sb ginti' ti' bepelikka bangkang

abar II

abar II *vb.* **1.** bemain-main nyapakka ai'. **2.** merabat ai' (baka ngena' suar, sengayuh, etb.).
• **ngabar** *vb.* **1.** nyapakka **2.** merabat: *Iya* ***ngabar*** *ai' ngena' penimba'.* **3.** ngambat.
abas *n.* **1.** pengawa' ngulihka penerang enggau isi' pekara ti' betul; peresa; pansik; uji: ***Abas*** *iya agi' mekang.* 0 *Alai Dum,* ***abas*** *dulu' bubu ke ditan tua'*

Figure 3: An excerpt of Iban dictionary

In Figure 3, the very small excerpt of the Iban dictionary shows fewer irregularities as compared to the Melanau-Mukah dictionary. The only problem is to get rid of the heading texts (marked by a rectangle) during the conversion into HTML format. Another problem, which is the main challenge, is to obtain one column text from the eight columns. Overall, the structure and irregularities found in the two dictionaries are listed in Table 1.

Table 1
Summary of the structure and irregularities

	Melanau-Mukah dictionary	Iban dictionary
Layout	Two columns per page	Eight columns per page
Alignment	Inconsistent	Consistent
Derived words position	Inconsistent, indicated by different punctuations: ';' or ','	Consistent and indicated by a bullet (•) symbol
Whitespaces	Irregular spacing	Irregular spacing
Set of characters	48 characters: Latin alphabet (lower and upper-case), digit, 12 punctuations	40 characters: Latin alphabet (lower and upper-case), digit, 9 punctuations
Paired-punctuations	Presence of imbalanced punctuations; two types: () and []	No imbalanced punctuations; only one type: ()
Information provided	Headword, headword number, pronunciation, POS tag, sense number, sense definition, examples, derived words	Headword, headword number, POS tag, sense number, sense definition, examples

B. *Text extraction*

The objective of the text extraction is to convert the original readable PDF dictionary file into a long sequence of text symbols (i.e. a flat text) as shown in Figure 4. Firstly, the original readable PDF file is converted into an HTML file using ABBYY Finereader. Then, the HTML file is parsed using HTMLParser and Python regular expressions to extract the contents of specific HTML tags.

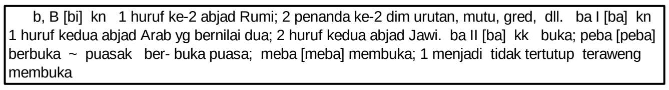

b, B [bi] kn 1 huruf ke-2 abjad Rumi; 2 penanda ke-2 dim urutan, mutu, gred, dll. ba I [ba] kn 1 huruf kedua abjad Arab yg bernilai dua; 2 huruf kedua abjad Jawi. ba II [ba] kk buka; peba [peba] berbuka ~ puasak ber- buka puasa; meba [meba] membuka; 1 menjadi tidak tertutup teraweng membuka

Figure 4: A flat text of dictionary content

ABBYY Finereader is a commercial optical character recognition (OCR) with the capability of recognising texts written in different languages such as Malay. Iban and Melanau languages are very close to the Malay Standard [8] and thus the use of Malay language for the recognition is the most appropriate one compared to other languages proposed by the OCR. HTMLParser is a Python programming language module that allows the parsing of text files formatted in HTML. The content of the dictionary lies between the two HTML tags <body> and </body>. Therefore, a Python script file was created to read an HTML file, parse it with HTMLParser, and then save the result in a text file (Figure 5). The saved data is then processed to combine all lines corresponding to the tag "Data:" in order to obtain the flat text as shown in Figure 4. The use of HTML format as an intermediary representation is to avoid all errors that the OCR may generate when transforming directly the PDF file into a plain text.

As each dictionary has its own microstructure, the conversion becomes dictionary-dependent. Therefore, for each studied dictionary, a set of regular expressions – corresponding to regular patterns in the dictionary – was written in Python to parse the text lines and divide them into dictionary fields. The process goes through three main steps: (1) converting the flat text into generic symbols, (2) observing the contextual environment of selected generic symbols using a collocation tools, and (3) writing the regular expressions for dictionary field separation.

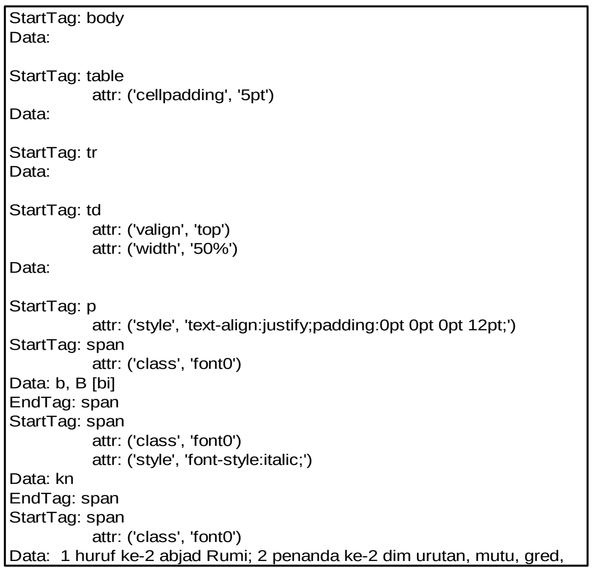
A Python script was created to map each textual unit into a corresponding generic symbol as indicated in Table 2. All punctuation symbols and any other symbols like bullet (•) remains as they are with whitespaces inserted around them. A sample of that conversion is shown in Figure 7.

Table 2
Generic symbols for text conversion

Symbol	Description	Examples
W	for any sequence of letters	nyapakka ⇒ W
D	for any sequence of digits indicating sense number	1. ⇒ D . 1 ⇒ D
%	for any POS tags	vb. ⇒ % . kn ⇒ %
$	for any Roman numerals as structural markers of headwords	II ⇒ $ IV. ⇒ $.

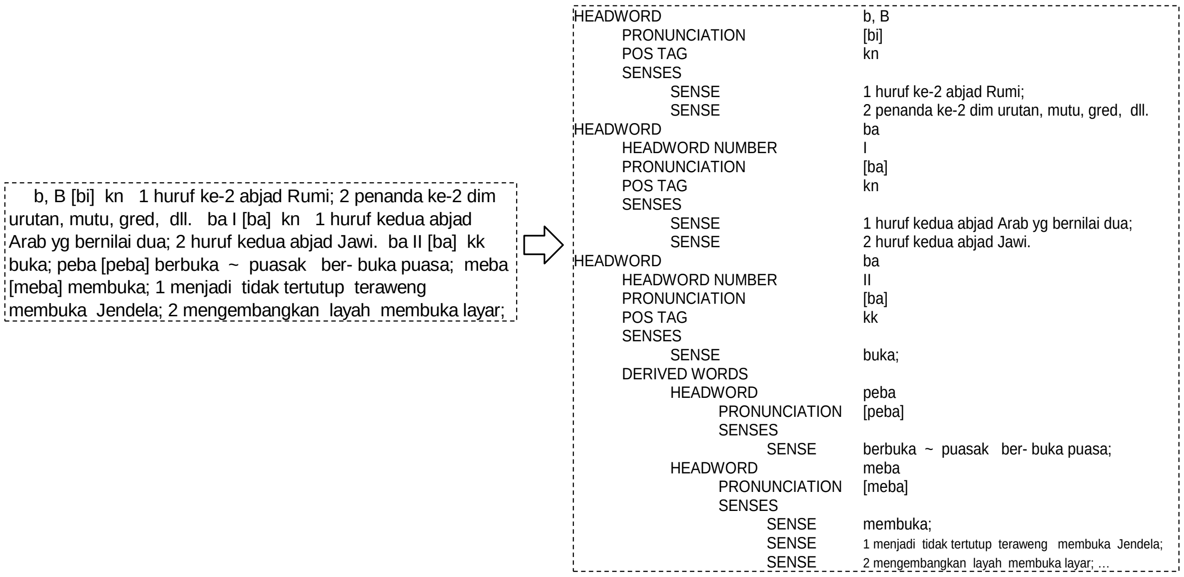
C. Dictionary fields annotation

The next step is to analyse and convert the flat text into separated fields that constitute the microstructure of the dictionary. The most challenging part is to find the regular patterns that can be used to separate the flat text into dictionary fields. Figure 6 illustrates the purpose of the separation. The indentations in that Figure have been added for the reader's convenience only.



```
StartTag: body
Data:

StartTag: table
            attr: ('cellpadding', '5pt')
Data:

StartTag: tr
Data:

StartTag: td
            attr: ('valign', 'top')
            attr: ('width', '50%')
Data:

StartTag: p
            attr: ('style', 'text-align:justify;padding:0pt 0pt 0pt 12pt;')
StartTag: span
            attr: ('class', 'font0')
Data: b, B [bi]
EndTag: span
StartTag: span
            attr: ('class', 'font0')
            attr: ('style', 'font-style:italic;')
Data: kn
EndTag: span
StartTag: span
            attr: ('class', 'font0')
Data:  1 huruf ke-2 abjad Rumi; 2 penanda ke-2 dim urutan, mutu, gred,
```

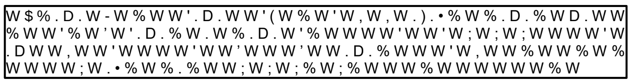
Figure 5: A sample of extracted HTML content



```
    b, B [bi]  kn   1 huruf ke-2 abjad Rumi; 2 penanda ke-2 dim
urutan, mutu, gred,  dll.   ba I [ba]  kn   1 huruf kedua abjad
Arab yg bernilai dua; 2 huruf kedua abjad Jawi.  ba II [ba]  kk
buka; peba [peba] berbuka  ~  puasak   ber- buka puasa;  meba
[meba] membuka; 1 menjadi  tidak tertutup  teraweng
membuka  Jendela; 2 mengembangkan  layah  membuka layar;
```

```
HEADWORD                                b, B
    PRONUNCIATION                       [bi]
    POS TAG                             kn
    SENSES
        SENSE                           1 huruf ke-2 abjad Rumi;
        SENSE                           2 penanda ke-2 dim urutan, mutu, gred,  dll.
HEADWORD                                ba
    HEADWORD NUMBER                     I
    PRONUNCIATION                       [ba]
    POS TAG                             kn
    SENSES
        SENSE                           1 huruf kedua abjad Arab yg bernilai dua;
        SENSE                           2 huruf kedua abjad Jawi.
HEADWORD                                ba
    HEADWORD NUMBER                     II
    PRONUNCIATION                       [ba]
    POS TAG                             kk
    SENSES
        SENSE                           buka;
    DERIVED WORDS
        HEADWORD                        peba
            PRONUNCIATION               [peba]
            SENSES
                SENSE                   berbuka  ~  puasak   ber- buka puasa;
        HEADWORD                        meba
            PRONUNCIATION               [meba]
            SENSES
                SENSE                   membuka;
                SENSE                   1 menjadi  tidak tertutup  teraweng   membuka  Jendela;
                SENSE                   2 mengembangkan  layah  membuka layar; …
```

Figure 6: From sequence of lines into dictionary fields



```
W $ % . D . W - W % W W ' . D . W W ' ( W % W ' W , W , W . ) . • % W % . D . % W D . W W
% W W ' % W ’ W ' . D . % W . W % . D . W ' % W W W W ' W W ' W ; W ; W ; W W W W ' W
. D W W , W W ' W W W W ' W W ’ W W W ’ W W . D . % W W W ' W , W W % W W % W %
W W W W ; W . • % W % . % W W ; W ; W ; % W ; % W W W % W W W W W W % W
```

Figure 7: A sample of converted text

The next step corresponds to the observation of the collocations of the symbols D, %, $, and punctuations to determine their contextual environment, that is, to find the most frequent symbols on their left and right sides. This information is used to write the most appropriate regular expression. For example, in the Iban dictionary, Roman numerals (% symbol) occurs 16 times and the symbols occurring on its left and right sides are displayed in Table 3. A simplified form of the Python regular expression that delimits the Roman numerals is then written as *newtext= re.sub(" I ", " <entrynum>I<\entrynum> ", text)*. The reading of the regular expression is as follows: substitute (sub) in the variable *text*, all expressions of the form " I " (a capital I surrounded by whitespaces) by " <entrynum>I<\entrynum> ", and save the result in the variable *newtext*.

Table 3
Left and right side of Roman numerals in Iban dictionary

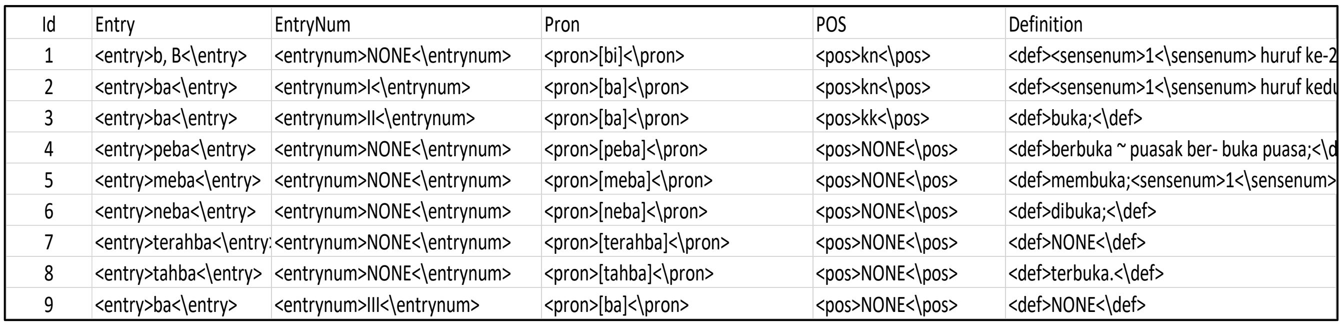
Left	Freq. Left	Freq. Right
Word (W)	13	2
POS tag (%)	0	14
Apostrophe (‘)	3	0

The observation of the collocations yields to the writing of eight and nine regular expressions for dividing the elements in the Iban and Melanau-Mukah dictionaries respectively.

D. MRD storage

Once each dictionary element is annotated, it can be extracted and stored in a more structured form. In this study, the final MRD is stored in CSV file. When the information is not provided in the original dictionary, the field is filled with the string “NONE”.

Table 4
Melanau-Mukah-Malay MRD stored in CSV format



Id	Entry	EntryNum	Pron	POS	Definition
1	<entry>b, B<\entry>	<entrynum>NONE<\entrynum>	<pron>[bi]<\pron>	<pos>kn<\pos>	<def><sensenum>1<\sensenum> huruf ke-2
2	<entry>ba<\entry>	<entrynum>I<\entrynum>	<pron>[ba]<\pron>	<pos>kn<\pos>	<def><sensenum>1<\sensenum> huruf kedu
3	<entry>ba<\entry>	<entrynum>II<\entrynum>	<pron>[ba]<\pron>	<pos>kk<\pos>	<def>buka;<\def>
4	<entry>peba<\entry>	<entrynum>NONE<\entrynum>	<pron>[peba]<\pron>	<pos>NONE<\pos>	<def>berbuka ~ puasak ber- buka puasa;<\d
5	<entry>meba<\entry>	<entrynum>NONE<\entrynum>	<pron>[meba]<\pron>	<pos>NONE<\pos>	<def>membuka;<sensenum>1<\sensenum>
6	<entry>neba<\entry>	<entrynum>NONE<\entrynum>	<pron>[neba]<\pron>	<pos>NONE<\pos>	<def>dibuka;<\def>
7	<entry>terahba<\entry:	<entrynum>NONE<\entrynum>	<pron>[terahba]<\pron>	<pos>NONE<\pos>	<def>NONE<\def>
8	<entry>tahba<\entry>	<entrynum>NONE<\entrynum>	<pron>[tahba]<\pron>	<pos>NONE<\pos>	<def>terbuka.<\def>
9	<entry>ba<\entry>	<entrynum>III<\entrynum>	<pron>[ba]<\pron>	<pos>NONE<\pos>	<def>NONE<\def>

IV. Conclusion and Future Work

This paper describes the three-sequential stages for transforming indigenous dictionaries saved in PDF file into structured dictionaries stored in CSV format. The most challenging part of the transformation is the automatic identification of dictionary fields in a sequence of strings. The proposed solution is the writing of a set of regular expressions obtained by the observation of the regular patterns in the sequence of strings. The research is still in progress but this paper has presented the outline of the proposed method. In near future, the transformation method will be applied to the complete indigenous dictionaries, and the result will be incorporated in Kirrkirr, a dictionary writing system, to visualise the errors, inconsistencies, and incompleteness.

Acknowledgements

This work is supported by Special Funding for Research Institutes through I03/SpFRI/1430/16/7. The authors thank Universiti Malaysia Sarawak for providing the resources used in the conduct of this study.

References

[1] Bakliwal, V. V. Devadath, and C. V. Jawahar, "Align Me: A framework to generate parallel porpus psing OCRs & bilingual dictionaries," *Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing*, Osaka, Japan, pp. 183-187, 2016.

[2] E. H. Klapicová, "Composition of the entry in a bilingual dictionary," *SKASE Journal of Theoretical Linguistics*, vol. 2, no. 3, pp. 57-74, 2005.

[3] T. Ejarvec, and N. Ide, "Markup enhancement: Converting CEE dictionaries into TEI, and beyond," In F. Kiefer, G. Kiss, & J. Pajzs (Ed.), *Papers in Computational Lexicography COMPLEX'99 Linguistics Institute*, Budapest: Hungarian Academy of Sciences, pp. 211-217, 1999.

[4] J. Mayfield and P. McNamee, "Converting On-Line Bilingual Dictionaries from Human-Readable to Machine-Readable Form," *SIGIR’02*, Tampere, Finland: ACM, 2002.

[5] E. Balabanova, and K. Ivanova, "Creating a machine-readable version of Bulgarian valence dictionary: (A case study of CLaRK system application)," *First Workshop on Treebanks and Linguistic Theories (TLT2002)*, Sozopol, Bulgaria, pp. 1-12, 2002.

[6] J. Gracia, M. Villegas, A. Gómez-Pérez, and N. Bel, "The apertium bilingual dictionaries on the web of data," *Semantic Web - Interoperability, Usability, Applicability*, pp. 1-10, 2017.

[7] M. Maxwell and A. Bills, "Endangered data for endangered languages: Digitizing print dictionaries," *Proceedings of the 2nd Workshop on the Use of Computational Methods in the Study of Endangered Languages*, Honolulu, Hawai‘i, USA: Association for Computational Linguistics, pp. 85-91, 2017.

[8] E.-L. Ng, A. W. Yeo, and B. Ranaivo-Malançon, "Identification of closely related indigenous languages: An orthographic approach," *Proceedings of the International Conference on Asian Language Processing (IALP)*, Singapore: IEEE, pp. 230-235, 2009.

[9] B. Karagol-Ayan, D. Doermann, and A. Weinbe, "Adaptive transformation-based learning for improving dictionary tagging," *11th Conference of the European Chapter of the Association for Computer Linguistics*, Trento, Italy: EACL, The Association for Computer Linguistics, pp. 257-264, 2006.

[10] A. A. Krizhanovsky, "Transformation of Wiktionary entry structure into tables and relations in a relational database schema," *Computing Research Repository (CoRR)*, 2010. Retrieved June 2017, from https://arxiv.org/abs/1011.1368

[11] M. Padró, N. Bel, and S. Necsulescu, "Towards the fully automatic merging of lexical resources: A step forward," *LREC 2012 Workshop on Language Resource*, Istanbul, Turkey: European Language Resources Association, pp. 8-14, 2012.