KLPT – Kurdish Language Processing Toolkit

Sina Ahmadi
Insight Centre for Data Analytics
National University of Ireland Galway
ahmadi.sina@outlook.com

Abstract

Despite the recent advances in applying language-independent approaches to various natural language processing tasks thanks to artificial intelligence, some language-specific tools are still essential to process a language in a viable manner. Kurdish language is a less-resourced language with a notable diversity in dialects and scripts and lacks basic language processing tools. To address this issue, we introduce a language processing toolkit to handle such a diversity in an efficient way. Our toolkit is composed of fundamental components such as text preprocessing, stemming, tokenization, lemmatization and transliteration and is able to get further extended by future developers. This project is publicly available[1].

1 Introduction

Language technology is an increasingly important field in our information era which is dependent on our knowledge of the human language and computational methods to process it. Unlike the latter which undergoes constant progress with new methods and more efficient techniques being invented, the processability of human languages does not evolve with the same pace. This is particularly the case of languages with scarce resources and limited grammars, also known as less-resourced languages.

Various natural language processing (NLP) tasks are of pipeline architecture; that is, to address a specific task, a few other language processing tasks may be initially required (Manning et al., 2014). With the current advances in the open-source movements, more researchers and industrial developers are encouraged to share their knowledge in an open-source manner, accessible under certain conditions (Ljungberg, 2000). Therefore, the development of underlying tasks in NLP for a specific language will potentially pave the way for further contributions to the field, by either improving the current tools or further progress in new tasks. For instance, tokenization as a fundamental task is widely required in many other applications such as part-of-speech tagging, machine translation and syntactic analysis. Once addressed, future researchers can build upon it for more advanced tasks or eventually improve it.

Despite a plethora of performant tools and specific frameworks for NLP, such as NLTK (Loper and Bird, 2002), Stanza (Qi et al., 2020), Teanga (Ziad et al., 2018) and spaCy[2], the progress with respect to less-resourced languages is often hindered by not only the lack of basic tools and resources but also the accessibility of the previous studies under an open-source licence. This is particularly the case of Kurdish, a less-resourced Indo-European language that is the focus of the current paper. As an example, although the task of spell-checking and stemming for Kurdish have been addressed by many previous studies, (Jaf and Ramsay, 2014; Salavati and Ahmadi, 2018; Mustafa and Rashid, 2018; Saeed et al., 2018a; Hawezi et al., 2019) to mention but a few, none of them provides an implementation of their tool under any licence.

On the other hand, some previous studies use specific frameworks that are hardly integrable and inter-operable. For instance, (Walther and Sagot, 2010) and (Walther et al., 2010) describe their efforts in developing a large-scale morphological lexicon and a part-of-speech tagger for Kurdish within the *Alexina* framework under the LGPL-LR licence. Despite the valuable impact of this study in the field, for example in (Cotterell et al., 2017) and (Gökırmak and Tyers, 2017), the tool does not

[1] https://github.com/sinaahmadi/klpt

[2] https://github.com/explosion/spaCy

IPA	b	t͡ʃ	d͡ʒ	d	f	g	h	ʒ	k	l	ɫ	m	n	p	q	ɾ	r	s	ʃ	t	v	w	x	j	z	ʕ	ħ	ɣ	ʔ
Latin	b	ç	c	d	f	g	h	j	k	l	ł/ll	m	n	p	q	r	ř/rr	s	ş	t	v	w	x	y	z	‘/’e/ë	ḧ/‘h	ẍ/x	’
Arabic	ب	چ	ج	د	ف	گ	ه	ژ	ک	ل	ڵ	م	ن	پ	ق	ر	ڕ	س	ش	ت	ڤ	و	خ	ی	ز	ع	ح	غ	ئ

(a) Consonants

IPA	aː	æ	eː	ɪ	iː	oː	uː	ʊ	ʉ:
Latin	a	e	ê	i	î	o	û	u	ü
Arabic	ا	ە	ێ	▨	ی	ۆ	وو	و	ۊ

(b) Vowels

Table 1: A comparison of the Kurdish alphabets. Variations are specified with "/"

seem to be widely used in the subsequent projects. As such, projects such as (Jaf and Ramsay, 2014) and (Ahmadi and Hassani, 2020a) tackle the very same topic from scratch.

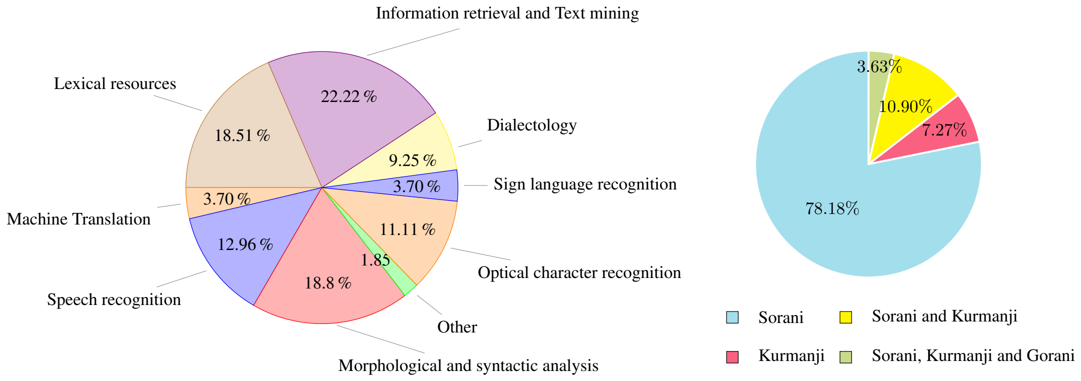
Language-specific toolkits have been previously designed for various languages, such as IceNLP for Icelandic (Loftsson and Rögnvaldsson, 2007), VnCoreNLP for Vietnamese (Vu et al., 2018), FudanNLP for Chinese (Qiu et al., 2013), PSI-Toolkit for Polish (Graliński et al., 2013) and ParsiPardaz for Persian (Sarabi et al., 2013). In the same vein, in order to facilitate the basic language processing tasks for Kurdish in an organized and methodical way and aware of the increasing importance of open-source and inter-operable tools for building more efficient systems and get further advanced in the field, we present KLPT–the Kurdish language processing toolkit. This toolkit is developed in Python and is composed of core modules and is extendable by future developers.

2 Kurdish Language

Kurdish belongs to the Northwestern branch of the Iranian languages within the Indo-European language family which is spoken by 20-30 million speakers in the Kurdish regions of Turkey, Iraq, Iran and Syria and also, among the Kurdish diaspora around the world (Ahmadi et al., 2019). The division of Kurdish into Northern Kurdish (or Kurmanji), Central Kurdish (or Sorani), Southern Kurdish and Laki, respectively with kmr, ckb, sdh and lki ISO 639-3 language codes, has been widely studied previously (Edmonds, 2013). Based on the structural differences between these, some scholars believe that they are distinct languages and therefore, refer to them as Kurdish languages (Kreyenbroek, 2005). On the other hand, it is also commonly believed by both scholars and Kurdish people that those are in fact different dialects of the Kurdish language (Haig and Matras, 2002; Matras, 2017). In this study, we remain with this theory and refer to them as Kurdish dialects. It is worth mentioning that despite the linguistic similarities of Zazaki, also known as Dimlî, and Gorani languages and the popular belief that they are dialects of Kurdish, studies show that they belong to the Zaza-Gorani language family which is independent from the Kurdish language (Paul, 1998; Jugel, 2014; Ahmadi, 2020c).

Kurdish has been historically written in various scripts, namely Cyrillic, Armenian, Latin and Arabic among which the latter two are still widely in use. Efforts in standardization of the Kurdish alphabets and orthographies have not succeeded to be globally followed by all Kurdish speakers in all regions (Tavadze, 2019; Haig and Matras, 2002; Aydoğan, 2012). As such, the Kurmanji dialect is mostly written in the Latin-based script while the Sorani, Southern Kurdish and Laki are mostly written in the Arabic-based script. That, not only scatters readers and speakers to communicate together, but also creates further challenges in processing the language (Esmaili, 2012; Ahmadi, 2019). Table 1 provides the Latin-based and Arabic-based Kurdish alphabets used for all the dialects.

Kurdish language is a highly inflectional language, particularly due to a high number of affixes and clitics (Ahmadi and Hassani, 2020b). Regarding nouns, although Sorani does not have gender or grammatical cases, it has a full article marking system for definite, indefinite and demonstrative in singular and plural forms (Jugel, 2014). On the other hand, Kurmanji has a fewer number of article markers for feminine and masculine genders (Thackston, 2006). With respect to the



(a) per sub-field in NLP

(b) per dialect

Figure 1: Proportion of publications related to Kurdish language processing

verbs, Kurdish has a few number of around 300 single-word verbs (Walther and Sagot, 2010), e.g. *kirdin/kirin* “to do”, which are inflected based on person (1,2,3, SG, PL), tense (past, present, future), aspect (indefinite, perfect, progressive, imperfective) and mood (indicative, subjunctive, conditional). Unlike Kurmanji, Sorani Kurdish does not have future tense and uses adverbs for this purpose. However, Kurdish extensively takes use of compound constructions for creating new verb forms, particularly with (Noun + Verb), (Adjective + Verb) and (Preposition + Verb) forms (Traida, 2007). For instance, *siław* ‘hi (n)’, *pîroz* ‘holy’ (adj) and *heł* (verbal particle denoting ‘up’) with the single-word verb *kirdin* can respectively form compound verbs *siław kirdin* “to greet”, *pîroz kirdin* “to congratulate” and *heł kirdin* “to turn on”. The stringing characteristic of the Arabic-based script of Kurdish further adds to this morphological complexity in such a way that several word forms may be concatenated together (Ahmadi, 2020b).

Regarding syntax, Kurdish has a subject–object–verb word order and is a null-subject (or pro-drop) language. The presence of grammatical markers for nominative and oblique cases varies within dialects and subdialects. For instance, in the Sorani subdialects of Sulaymaniyah and Erbil, respectively categorized as Southern Sorani and Northern Sorani by (Matras, 2017), the oblique case is marked differently. Another particularity of the Kurdish language is its morphosyntactic alignment in the past tense of transitive verbs. In such tenses, an ergative–absolute alignment occurs where the subject of intransitive verbs behaves like the patient of the transitive verb in the past (Haig, 1998; Karimi, 2014). Unlike Kurmanji which uses oblique cases for this purpose, Sorani only uses different pronominal markers to specify ergativity, therefore it is called split-ergative (Esmaili and Salavati, 2013). Except the past tenses, a nominative-accusative alignment is observed in other tenses.

Not being equally documented and used, Kurdish dialects have different levels of linguistic resourcefulness. In comparison to Sorani and Kurmanji which are widely used by the media and press, Southern Kurdish and Laki are under-documented and lack basic language resources such as electronic dictionaries and corpora (Fattah, 2000; Ahmadi et al., 2019; Ahmadi, 2020c).

3 Current State of Kurdish Language Processing

The earliest works in the field of Kurdish language processing date back to 2009. Our literature review indicates that some of these contributions fail to provide open-source solutions. Despite financial and scientific constraints in Kurdish language processing, the Kurdish Language Processing Project (KLPP) (Esmaili et al., 2013) in 2012[3] and Kurdish Basic Language Resource Kit (Kurdish BLARK) (Hassani, 2018) in 2014[4] have succeeded to promote an open-source vision based on research volunteering within the Kurdish scientific communities. However, the outcomes of these projects are mostly released in an unorganized manner for individual tasks.

In order to understand the current state of the Kurdish language in the realm of NLP and com-

[3] http://klpp.github.io

[4] https://kurdishblark.github.io

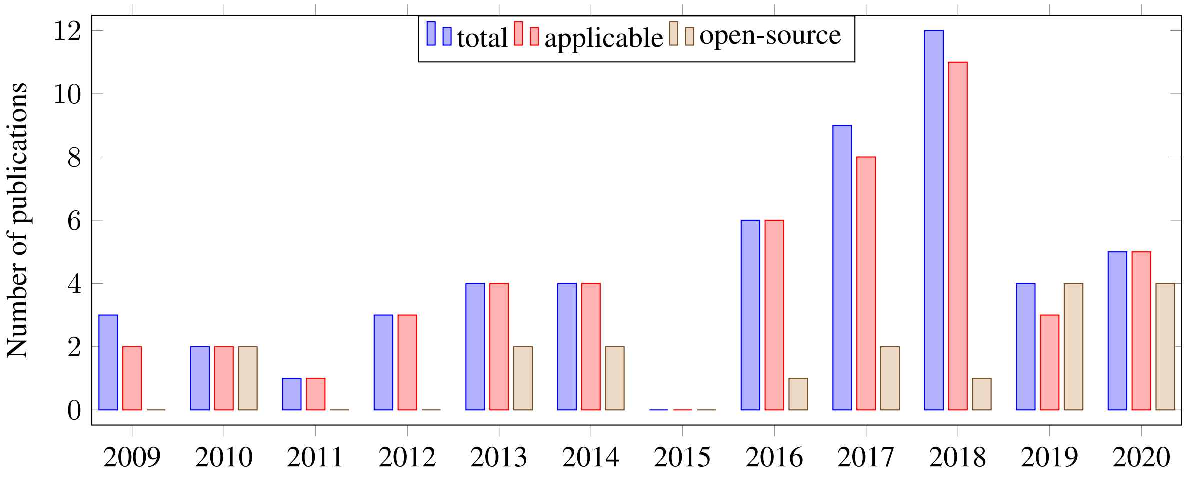


Figure 2: Number of scientific publications directly related to Kurdish language processing per year

putational linguistics, we reviewed the scientific publications that directly address an issue in those fields. A total number of 53 publications are collected from the widely-used academic databases and search engines such as Google Scholar[5], and then classified based on their discussed sub-fields which are illustrated in Figure 1. The Kurdish dialects are not evenly discussed in the previous studies, with Sorani making up a predominant proportion of almost 90%. Although a smaller proportion represents the Kurmanji dialect, no publication is found with respect to processing of the Southern Kurdish or Laki dialects. Regarding the research focus of the previous works, a range of NLP sub-fields has been addressed, particularly in text mining, morphological and syntactic analysis and, creation of lexical resources. We exceptionally included optical character recognition as it is of importance for converting printed material to electronic forms (Ahmadi et al., 2019). The full list of the surveyed papers can be found in Appendix A.2.

More importantly, we analyze previous publications from the following two perspectives:

- Open-source: Does the paper provide the discussed resource or tool under an open-source license? To this end, we verified the content of the papers and also, checked the Web, particularly major distributed version control systems such as GitHub[6], GitLab[7] and BitBucket[8].
- Applicability: Does the paper, implicitly or explicitly, propose an approach or methodology that can be applied to solve the same problem in the other dialects of Kurdish? For the choice of the word, we were inspired by (Årdal et al., 2011) where the possibility of applying common practices of software development for drug discovery are investigated. For instance, (Ahmadi et al., 2019) is deemed an applicable contribution where lexicographical resources can be created for other dialects. On the other hand, (Ahmadi, 2019) is not applicable to other dialects due to its ad-hoc solution for transliterating Sorani texts according to its phonological and phonetic rules.

Figure 2 provides the number of previous publications in the Kurdish language processing field per year, and specifies their open-source status and their applicability. Although most of these publications are applicable to other dialects, only 18 out of 53 of them provide their resources or tools under an open-source license. Among the open-source ones, 11 are outcomes of volunteering projects, KLPP and Kurdish-BLARK. Given the small number of non-scientific contributions, we did not include them in this survey. A few notable examples of such contributions are Kurdînûs[9], Vejin Dictionaries[10] and VejinBooks[11] which mostly focus on Sorani Kurdish and script conversion tasks.

[5] https://scholar.google.com
[6] https://github.com
[7] https://gitlab.com
[8] https://bitbucket.org
[9] https://github.com/aso-mehmudi/kurdinus
[10] https://lex.vejinbooks.com
[11] https://books.vejin.net

4 KLPT Architecture

KLPT is implemented in Python and is composed of four core modules with specific tasks. Although we were inspired by the functionality of relevant NLP toolkits, particularly NLTK and spaCy, no external library is used in this toolkit. Regarding the toolkit design, we followed the rules of scientific software development suggested by (Prlić and Procter, 2012) along with common practices in Python programming language. Figure 3 provides the structure of the toolkit. In order to facilitate the integration of variations specific to dialects and scripts and more importantly, to avoid hard-coding, required files are provided in the `data` folder. For instance, the data required for the `preprocess` module is imported from `preprocess.json`. In addition, third-party programs can be provided in `bin`. `test` and `docs` respectively contain test cases and project documentation. Regarding the latter, we use Sphinx documentation generator[12].

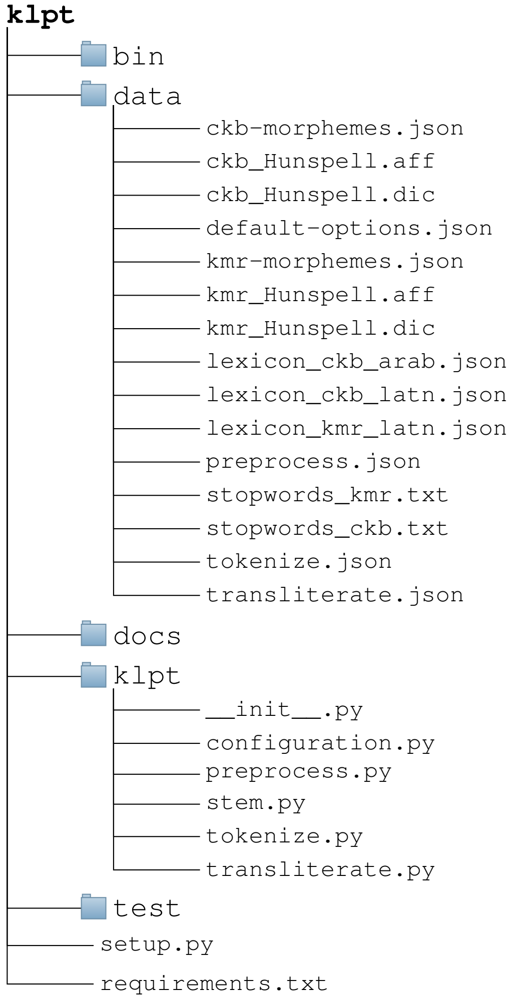
It is worth noting that each module within the `klpt` package has been previously studied and evaluated separately. Our goal is to introduce the functionality of the modules within the toolkit in this section.

4.1 Preprocess

Many keyboard layouts are specifically designed for Kurdish where different character encoding are assigned to visually-similar graphemes. In addition to the usage of non-Kurdish keyboards, such as Arabic, Turkish and Persian keyboards, such diversity creates abnormality across texts in Kurdish writing. For instance, the grapheme ی (*î/y*), can be represented as ي (U+064A), ى (U+0649), ﻲ (U+FEF2), ﻱ (U+FEF1) and ی (U+06CC), among which only the latter should be used in the Arabic-based script of Kurdish. Moreover, various writing conventions are used for each dialect and script. For instance, in Kurmanji, when dates are affixed with a morpheme, the suffix may be separated by ’, - or without any marker as in *2020’an*, *2020-an* and *2020an*.

To remedy such issues in an automatic and structured manner, the `preprocess` module provides two main functions: `normalize()` for normalizing encoding abnormalities by unifying characters in such a way that only one specific encoding is used for each grapheme and, `standardize()` which applies orthographic conventions to the text. For example, when *hêvî* ‘hope’ is suffixed with the vowel *a* (*Izafa*, meaning ‘of’), a semi-vowel *y* appears between the two vowels and is usually written as *hêviya* or *hêvîya* ‘hope of’. As the latter form is considered less ambiguous, this function converts the first form accordingly. Although defining a universal orthography for Kurdish is out of scope of our project, we believe that writing conventions and orthographies should be addressed to some extent. Therefore, in this initial version, we follow the writing conventions proposed by (Aydoğan, 2012) for Kurmanji and (Hashemi, 2016) for Sorani.

In addition to these two functions, `unify_numeral()` is provided to convert numerals, namely in Farsi (۰۱۲۳۴۵۶۷۸۹), Eastern Arabic (٠١٢٣٤٥٦٧٨٩) and Western Arabic (0123456789). Although we set the latter as default for all scripts, users will have the



```
klpt
├── bin
├── data
│   ├── ckb-morphemes.json
│   ├── ckb_Hunspell.aff
│   ├── ckb_Hunspell.dic
│   ├── default-options.json
│   ├── kmr-morphemes.json
│   ├── kmr_Hunspell.aff
│   ├── kmr_Hunspell.dic
│   ├── lexicon_ckb_arab.json
│   ├── lexicon_ckb_latn.json
│   ├── lexicon_kmr_latn.json
│   ├── preprocess.json
│   ├── stopwords_kmr.txt
│   ├── stopwords_ckb.txt
│   ├── tokenize.json
│   └── transliterate.json
├── docs
├── klpt
│   ├── __init__.py
│   ├── configuration.py
│   ├── preprocess.py
│   ├── stem.py
│   ├── tokenize.py
│   └── transliterate.py
├── test
├── setup.py
└── requirements.txt
```

Figure 3: Structure of KLPT

[12] https://www.sphinx-doc.org

choice to modify the numerals according to the administrations in the Kurdish regions. All these three functions are then evoked within `preprocess()` function which normalizes, standardizes and unifies the text according to the given arguments.

The general procedure followed in this module can be summarized as string replacement. For this purpose, we define regular expressions for each dialect and script. The regular expressions along with the character mappings are provided in `preprocess.json` in such an order that the intended normalization and standardization are carried out correctly. Although this module is not explicitly evoked within other modules, except in the `transliterate` module, it is recommended that the output of the preprocessing module be used as the input of other modules by the user.

4.2 Transliterate

Given the diversity of the alphabets used in Kurdish, transliteration is a necessity to facilitate the communication between speakers and is also beneficial to various NLP tasks, such as named-entity recognition and machine translation. Although Kurdish orthographies are phonemic, i.e. each grapheme is supposed to represent a single phoneme, transliterating characters within the alphabets is more challenging than it appears. This is particularly due to و (`U+0648`) and ی (`U+06CC`) in the Arabic-based alphabet which can be respectively mapped to '*u/w*' and '*î/y*'. For instance, و in بیور and کورت is transliterated as *bîwir* 'axe' and *kurt* 'short', respectively. Moreover, there is no grapheme for the vowel *i*, also known as *Bizroke* "the little furtive", in the Arabic-based script which creates further challenges in the morphological analysis of the language (Ahmadi, 2019).

In this module, we focus on transliterating Arabic-based and Latin-based scripts of Kurdish using Wergor transliterator[13] (Ahmadi, 2019). This tool uses a rule-based approach based on the phonological and syllabic characteristics of Kurdish for distinguishing double-usage characters, i.e. و and ی, and predicting the placement of *i*. Although the algorithm efficiently transliterates double-usage characters, it has been evaluated to detect *i* with a low accuracy of 39%.

4.3 Stem

Although the task of stemming has been previously addressed in the literature, no open-source viable solution was available for Kurdish. Therefore, we developed morphological rules containing combinations of Kurdish morphemes in Sorani and Kurmanji, and also an annotated lexicon containing lemmas with specific flags such as part-of-speech tags and stems. The morphological rules and the lexicons are then used to develop a morphological analyzer and spell-checker in Hunspell (Ooms, 2017) for Kurdish, where they are respectively known as affixes (`.aff`) and dictionary (`.dic`). Thanks to the wide usage of Hunspell in open-source text editors such as Apache OpenOffice, our development will be also beneficial for general purposes such as spell-checking in text editors. More importantly, we integrate Hunspell in KLPT for this module using a wrapper program[14].

The Stem module comes with two classes: `Stem` and `Spellcheck`. Although these two classes focus on two different tasks, they are provided in the same module as they are both based on the same implementation in Hunspell. Given a word, the `Stem` class provides four main functions, namely `stem()` for retrieving word-form stem, e.g. *kirdin/kirin* (do.INF) → *kir*, `lemmatize()` for lemmatization, e.g. *kirdbûm* (do.1SG.PST.PFV) → *kirdin*, `analyze()` for morphological analysis which returns a dictionary containing the flags according to Hunspell such as part-of-speech, terminal suffixes and inflectional suffixes and finally, `suffix_suggest()` which returns all the possible suffixes that can appear with a given lexeme. In addition to these, `generate()` will also be added to the module which generates a word-form given morphemes.

On the other hand, the `Spellcheck` class provides `check_spelling()` and `correct_spelling()` which are respectively used for spell checking (Boolean output) and spell correction. For instance, given خواردووماته (*xwardûmate*), `check_spelling()` detects that it is incorrectly written and a few suggestions are provided by `correct_spelling()`, among which خواردوومانه (*xwardûmane*) "(we) have eaten". The performance of the tool is further described in (Ahmadi, 2020d,a).

[13] https://github.com/sinaahmadi/wergor

[14] https://github.com/MSeal/cython_hunspell

4.4 Tokenize

Although both Arabic-based and Latin-based alphabets use spaces to delimit word boundaries, not all words correspond to a token in Kurdish. This is particularly due to the complex morphology, e.g. article marking suffixes, and the writing traditions. In the Arabic-based alphabet, there is a tendency to concatenate clitics, affixes and words together which results many tokens being written as one single word-form without any space as in هیواشیانە (*hîwaşyane*) "(it) is also their hope" which is composed of four tokens, noun *hîwa*, endoclitic *=ş*, pronominal enclitic *-yan* and present copula *e*. The Latin-based script, particularly when used for writing Kurmanji, respects word boundaries in a better way. For instance, the same phrase is written as "*hêvîya wan jî ew e*".

In this module, we use the tokenization approach proposed by (Ahmadi, 2020b). This approach uses an annotated lexicon with a morphological analyzer to tokenize words in Sorani and Kurmanji. Given the wide usage of compound forms in word formation in Kurdish, a lexicon is also provided for multi-word expressions (MWEs) and their possible forms, with and without space. That way, the inconsistencies in writing compound words is tackled efficiently. In addition to `mwe_tokenize()` and `word_tokenize()` which are respectively provided for the tokenization of words and MWEs, `sent_tokenize()` is a third function which tokenizes a given text into sentences based on punctuation marks. It is worth mentioning that words and MWEs are respectively separated by ▁ and ▁▁▁ by default which can be customized by the user.

4.5 Configuration

Given the combination of scripts and dialects of the input data, verification of the several configurations of each class can be complex. Therefore, we provide the `configuration` module which is used internally within the modules when an object of a class is initialized. This way, the class constructors validate the arguments by evoking this module and the error-handling is carried out only in the `Configuration` class.

For further clarification on the interaction of the individual modules within the KLPT package, Figure A.5 shows its package and class diagrams in the Unified Modeling Language (UML).

5 Usages

In this section, we provide basic usages of the application programming interface (API) of the KLPT package. The package is available on the Python Package Index (PyPI)[15] in Python 3.5 and later and, can be installed as follows:

```
pip install klpt
```

The installation of the package comes with the data files, i.e. `data` folder, and requirements which are also installed. Once the package installed, each module can be imported and used as described above. Figure 4 provides an example on how to work with various modules of the package.

```
>>> from klpt.preprocess import Preprocess
>>> from klpt.transliterator import Transliterate
>>> from klpt.tokenize import Tokenize
>>> from klpt.stem import Stem

# Preprocess module
>>> preprocessor = Preprocess("Sorani", "Arabic",
numeral="Latin")
>>> preprocessor.normalize("لە ســـــاڵەکانی ١٩٥٠دا")
لە ساڵەکانی 1950دا
>>> preprocessor.standardize("راستە لەو ووڵاتەدا")
ڕاستە لەو وڵاتەدا

# Transliterate module
>>> transliterator = Transliterate("Kurmanji", "Latin",
target_script="Arabic")
>>> transliterator.transliterate("rojhilata navîn")
'رۆژهلاتا ناڤین'

# Stem module
>>> stemmer = Stem("Sorani", "Arabic")
>>> stemmer.check_spelling("سوتاندبووت")
False
>>> stemmer.correct_spelling("سوتاندبووت")
('سووتاندبووت', 'سووتاندت', 'سووتاندن', 'سووتاند')
>>> stemmer.stem("سووتاندبووت")
('سووت',)
>>> stemmer.analyze("دیتبامن")
{'pos': 'verb', 'is': 'past_intransitive', 'stem':
'دی', ‘verb_stem': 'دیت', 'terminal_suffix': 'بامن'}

# Tokenize module
>>> tokenizer = Tokenize("Kurmanji", "Latin")
>>> tokenizer.word_tokenize("endamên encûmena wezîrên")
['▁endam▁ên', '▁encûmen▁a', '▁wezîr▁ên']
```

Figure 4: Basic usage of the KLPT package for the Sorani and Kurmanji dialects

6 Conclusion and Future Work

In this paper, we present KLPT, an open-source toolkit developed in Python and composed of core modules, namely `Preprocess`, `Stem`,

[15] https://pypi.org

`Tokenize` and `Transliterate` for processing the Sorani and Kurmanji dialects of Kurdish. In addition to the provided modules, the toolkit enables future researchers to contribute their work by extending the modules for more advanced tasks and other dialects. We believe that recognizing every single contribution to the toolkit is encouraging for researchers and also, beneficial to help Kurdish to pass over its less-resourced status.

As a future work, we would like to extend the current version to include syntactic and semantic parsing for Sorani and Kurmanji. Given the scarcity of resources regarding computational linguistics and natural language processing, we believe that the KLPT package will create a new field of interest for Kurdish linguists as well. Therefore, we are aiming at creating educational content to introduce the field to non-expert public too.

Acknowledgments

The author would like to thank his two colleagues, Dr. Kyumars Sheykh Esmaili and Dr. Hossein Hassani who respectively initiated the Kurdish Language Processing Project and Kurdish-BLARK. Despite the lack of financial support of Kurdish-related projects, these initiatives have made huge contributions thanks to volunteer researchers. Similarly, the constructive comments of the three anonymous reviewers were very useful and are much appreciated.

References

Roshna Abdulrahman, Hossein Hassani, and Sina Ahmadi. 2019. Developing a Fine-grained Corpus for a Less-resourced Language: the case of Kurdish. In *Proceedings of the 2019 Workshop on Widening NLP*, pages 106–109.

Roshna Omer Abdulrahman and Hossein Hassani. 2020. Using Punkt for Sentence Segmentation in non-Latin Scripts: Experiments on Kurdish (Sorani) Texts. *arXiv preprint arXiv:2004.14134*.

Sina Ahmadi. 2019. A Rule-based Kurdish Text Transliteration System. *Asian and Low-Resource Language Information Processing (TALLIP)*, 18(2):18:1–18:8.

Sina Ahmadi. 2020a. A Lemmatization System for Sorani Kurdish. under review.

Sina Ahmadi. 2020b. A Tokenization System for the Kurdish Language. In *the Proceedings of the Seventh Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2020)*.

Sina Ahmadi. 2020c. Building a Corpus for the Zaza–Gorani Language Family. In *the Proceedings of the Seventh Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2020)*.

Sina Ahmadi. 2020d. Hunspell for Sorani Kurdish Spell-checking and Morphological Analysis. under review.

Sina Ahmadi and Hossein Hassani. 2020a. Towards Finite-State Morphology of Kurdish. *arXiv preprint arXiv:2005.10652*.

Sina Ahmadi and Hossein Hassani. 2020b. Towards Finite-State Morphology of Kurdish. *(under review) ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*.

Sina Ahmadi, Hossein Hassani, and Kamaladdin Abedi. 2020. A corpus of the Sorani Kurdish folkloric lyrics. In *Proceedings of the 1st Joint Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL) Workshop at the 12th International Conference on Language Resources and Evaluation (LREC)*.

Sina Ahmadi, Hossein Hassani, and John P McCrae. 2019. Towards electronic lexicography for the Kurdish language. In *Proceedings of the sixth biennial conference on electronic lexicography (eLex)*. eLex 2019.

Abdulbasit Al-Talabani, Zrar Abdul, and Azad Ameen. 2017. Kurdish dialects and neighbor languages automatic recognition. *ARO-The Scientific Journal of Koya University*, 5(1):20–23.

Purya Aliabadi. 2014. Semi-automatic development of KurdNet, the Kurdish Wordnet. In *Proceedings of the ACL 2014 Student Research Workshop*, pages 94–99.

Purya Aliabadi, Mohammad Sina Ahmadi, Shahin Salavati, and Kyumars Sheykh Esmaili. 2014. Towards building kurdnet, the Kurdish Wordnet. In *Proceedings of the Seventh Global Wordnet Conference*, pages 1–6.

Christine Årdal, Annette Alstadsæter, and John-Arne Røttingen. 2011. Common characteristics of open source software development and applicability for drug discovery: a systematic review. *Health Research Policy and Systems*, 9(1):36.

Duygu Ataman. 2018. Bianet: A parallel news corpus in turkish, kurdish and english. *arXiv preprint arXiv:1805.05095*.

Mustafa Aydoğan. 2012. *Rêbera rastnivîsînê*. Weşanxaneya Rûpelê. Ziman. Rûpel.

Anvar Bahrampour, Wafa Barkhoda, and Bahram Zahir Azami. 2009. Implementation of three text to speech systems for Kurdish language. In *Iberoamerican Congress on Pattern Recognition*, pages 321–328. Springer.

Wafa Barkhoda, Bahram ZahirAzami, Anvar Bahrampour, and Om-Kolsoom Shahryari. 2009. A comparison between allophone, syllable, and diphone based TTS systems for Kurdish language. In *Signal Processing and Information Technology (ISSPIT), 2009 IEEE International Symposium on*, pages 557–562. IEEE.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, et al. 2017. CoNLL-SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection in 52 Languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30.

Fatemeh Daneshfar, Wafa Barkhoda, and Bahram Zahir Azami. 2009. Implementation of a Text-to-Speech System for Kurdish Language. In *Digital Telecommunications, 2009. ICDT'09. Fourth International Conference on*, pages 117–120. IEEE.

Özlem Batur Dinler and Nizamettin Aydin. 2018a. Extraction of the acoustic features of semi-vowels in the Kurdish language. *The Online Journal of Science and Technology-April*, 8(2).

Özlem Batur Dinler and Nizamettin Aydin. 2018b. Kurdish recognition system digit. *The Online Journal of Science and Technology*, 8(1):101.

Özlem Batur Dinler and Fatih Karabıber. 2017. Formant analysis of vowels in Kurdish language. In *Signal Processing and Communications Applications Conference (SIU), 2017 25th*, pages 1–4. IEEE.

Alexander Johannes Edmonds. 2013. The Dialects of Kurdish. *Ruprecht-Karls-Universität Heidelberg*.

Kyumars Sheykh Esmaili. 2012. Challenges in Kurdish text processing. *arXiv preprint arXiv:1212.0074*.

Kyumars Sheykh Esmaili, Donya Eliassi, Shahin Salavati, Purya Aliabadi, Asrin Mohammadi, Somayeh Yosefi, and Shownem Hakimi. 2013. Building a test collection for Sorani Kurdish. In *Computer Systems and Applications (AICCSA), 2013 ACS International Conference on*, pages 1–7. IEEE.

Kyumars Sheykh Esmaili and Shahin Salavati. 2013. Sorani kurdish versus kurmanji kurdish: An empirical comparison. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 300–305.

Kyumars Sheykh Esmaili, Shahin Salavati, and Anwitaman Datta. 2014. Towards Kurdish information retrieval. *ACM Transactions on Asian Language Information Processing (TALIP)*, 13(2):7.

Ismaïl Kamandâr Fattah. 2000. *Les dialectes kurdes méridionaux: étude linguistique et dialectologique*. Acta Iranica : Encyclopédie permanente des études iraniennes. Peeters.

Memduh Gökırmak and Francis M Tyers. 2017. A dependency treebank for Kurmanji Kurdish. In *Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017)*, pages 64–72.

Filip Graliński, Krzysztof Jassem, and Marcin Junczys-Dowmunt. 2013. PSI-toolkit: A natural language processing pipeline. In *Computational Linguistics*, pages 27–39. Springer.

Geoffrey Haig. 1998. On the interaction of morphological and syntactic ergativity: Lessons from Kurdish. *Lingua*, 105(3-4):149–173.

Geoffrey Haig and Yaron Matras. 2002. Kurdish linguistics: a brief overview. *STUF-Language Typology and Universals*, 55(1):3–14.

Dyako Hashemi. 2016. Kurdish orthography [In Kurdish]. `http://yageyziman.com/Renusi_Kurdi.htm`. Accessed: 2020-07-25.

Abdulla D Hashim and Fattah Alizadeh. 2018. Kurdish sign language recognition system. *UKH Journal of Science and Engineering*, 2(1):1–6.

Hossein Hassani. 2017a. Kurdish interdialect machine translation. In *Proceedings of the fourth workshop on NLP for similar languages, varieties and dialects (VarDial)*, pages 63–72.

Hossein Hassani. 2017b. A method for proper noun extraction in Kurdish. In *OASIcs-OpenAccess Series in Informatics*, volume 56. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Hossein Hassani. 2018. BLARK for multi-dialect languages: towards the Kurdish BLARK. *Language Resources and Evaluation*, 52(2):625–644.

Hossein Hassani and Rahel Kareem. 2011. Kurdish text to speech (ktts). *Designing for Global Markets*, 10:79–89.

Hossein Hassani and Dzejla Medjedovic. 2016. Automatic Kurdish dialects identification. *Computer Science & Information Technology*, 6(2):61–78.

Roojwan Sc Hawezi, Muhammed Y Azeez, and Ahmed A Qadir. 2019. Spell checking algorithm for agglutinative languages "Central Kurdish as an example". In *2019 International Engineering Conference (IEC)*, pages 142–146. IEEE.

Sardar Jaf. 2016. A simple approach to unify ambiguously encoded Kurdish characters. In *Proceedings of the International Conference Computational Linguistics in Bulgaria (CLIB 2016).*, pages 86–94. Institute for Bulgarian Language, Bulgarian Academy of Sciences.

Sardar Jaf and Allan Ramsay. 2014. A Stemmer and a POS tagger for Sorani Kurdish. In *6th International Conference on Corpus Linguistics (CILC-14)*. Gran Canaria, Spain, Cambridge Scholars.

Sardar Jaf and Allan Ramsay. 2016. A Rule-based Part-of-speech Tagger For Sorani Kurdish. *Input a Word, Analyze the World: Selected Approaches to Corpus Linguistics*, page 39.

Thomas Jugel. 2014. On the linguistic history of Kurdish. *Kurdish Studies*, 2(2):123–142.

Kanaan M Kaka-Khan. 2017. Building Kurdish chatbot using free open source platforms. *UHD Journal of Science and Technology*, 1(2):46–50.

Kanaan M Kaka-Khan. 2018. English to Kurdish Rule-based Machine Translation System. *UHD Journal of Science and Technology*.

Zina Kamal and Hossein Hassani. 2020. Towards Kurdish text to sign translation. In *Proceedings of the LREC2020 9th Workshop on the Representation and Processing of Sign Languages: Sign Language Resources in the Service of the Language Community, Technological Challenges and Application Perspectives*, pages 117–122, Marseille, France. European Language Resources Association (ELRA).

Yadgar Karimi. 2014. On the syntax of ergativity in Kurdish. *Poznan Studies in Contemporary Linguistics*, 50(3):231–271.

Philip G Kreyenbroek. 2005. On the Kurdish language. In *The Kurds*, pages 62–73. Routledge.

Patrick Littell, Kartik Goyal, David R Mortensen, Alexa Little, Chris Dyer, and Lori Levin. 2016. Named entity recognition for linguistic rapid response in low-resource languages: Sorani Kurdish and Tajik. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 998–1006.

Jan Ljungberg. 2000. Open source movements as a model for organising. *European Journal of Information Systems*, 9(4):208–216.

Hrafn Loftsson and Eiríkur Rögnvaldsson. 2007. IceNLP: A natural language processing toolkit for Icelandic. In *Eighth Annual Conference of the International Speech Communication Association*.

Edward Loper and Steven Bird. 2002. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70.

Shervin Malmasi. 2016. Subdialectal differences in Sorani Kurdish. In *Proceedings of the third workshop on nlp for similar languages, varieties and dialects (vardial3)*, pages 89–96.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Yaron Matras. 2017. Revisiting Kurdish dialect geography: Preliminary findings from the Manchester Database. `http://kurdish.humanities.manchester.ac.uk/wp-content/uploads/2017/07/PDF-Revisiting-Kurdish-dialect-geography.pdf`. [Online; accessed 04-Mar-2019].

Bayan Omar Mohammed. 2012. Uniqueness in Kurdish handwriting. *International Journal of Engineering & Computer Science IJECS-IJENS*, 12(06):42–50.

Bayan Omar Mohammed. 2013. Handwritten Kurdish character recognition using geometric discertization feature. *Volume*, 4:51–55.

FS Mohammed, L Zakaria, Nazlia Omar, and MY Albared. 2012. Automatic Kurdish Sorani text categorization using n-gram based model. In *Computer & Information Science (ICCIS), 2012 International Conference on*, volume 1, pages 392–395. IEEE.

Arazo M Mustafa and Tarik A Rashid. 2018. Kurdish stemmer pre-processing steps for improving information retrieval. *Journal of Information Science*, 44(1):15–27.

Jeroen Ooms. 2017. hunspell: High-Performance Stemmer, Tokenizer, and Spell Checker. *https://hunspell.github.io/*.

Ludwig Paul. 1998. The position of Zazaki among West Iranian languages. *Old and Middle Iranian Studies*, pages 163–176.

Andreas Prlić and James B Procter. 2012. Ten simple rules for the open development of scientific software. *PLoS Comput Biol*, 8(12):e1002802.

Akam Qader and Hossein Hassani. 2019. Kurdish (sorani) speech to text: Presenting an experimental dataset. *arXiv preprint arXiv:1911.13087*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*.

Xipeng Qiu, Qi Zhang, and Xuan-Jing Huang. 2013. Fudannlp: A toolkit for chinese natural language processing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 49–54.

Tarik A Rashid, Arazo M Mustafa, and A Saeed. 2017a. A robust categorization system for Kurdish Sorani text documents. *Inf. Technol. J*, 16(1):27–34.

Tarik A Rashid, Arazo M Mustafa, and Ari M Saeed. 2017b. Automatic Kurdish text classification using kdc 4007 dataset. In *International Conference on Emerging Internetworking, Data & Web Technologies*, pages 187–198. Springer.

Ari M Saeed, Tarik A Rashid, Arazo M Mustafa, Rawan A Al-Rashid Agha, Ahmed S Shamsaldin, and Nawzad K Al-Salihi. 2018a. An evaluation of Reber stemmer with longest match stemmer technique in Kurdish Sorani text classification. *Iran Journal of Computer Science*, 1(2):99–107.

Ari M Saeed, Tarik A Rashid, Arazo M Mustafa, Polla Fattah, and Birzo Ismael. 2018b. Improving Kurdish Web Mining through Tree Data Structure and Porter's Stemmer Algorithms. *UKH Journal of Science and Engineering*, 2(1):48–54.

Shahin Salavati and Sina Ahmadi. 2018. Building a Lemmatizer and a Spell-checker for Sorani Kurdish. In *Proceedings of the 8th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*, Poznan, Poland.

Shahin Salavati, Kyumars Sheykh Esmaili, and Fardin Akhlaghian. 2013. Stemming for Kurdish information retrieval. In *Asia Information Retrieval Symposium*, pages 272–283. Springer.

Zahra Sarabi, Hooman Mahyar, and Mojgan Farhoodi. 2013. ParsiPardaz: Persian language processing toolkit. In *ICCKE 2013*, pages 73–79. IEEE.

Abdusalam Abdulla Shaltooki and Mzhda Hiwa Hama. 2016. Sentiment analyses for Kurdish social network texts using Naive Bayes classifier. *Journal of Human Development*, 1(4):393–397.

Givi Tavadze. 2019. Spreading of the Kurdish Language Dialects and Writing Systems Used in the Middle East. *Bull. Georg. Natl. Acad. Sci*, 13(1).

Wheeler M Thackston. 2006. *Kurmanji Kurdish:A Reference Grammar with Selected Readings*. Harvard University.

Sandrine Traida. 2007. *Morphosyntactic Study of the compound verbs in Sorani Kurdish Étude morphosyntaxique des verbes composés (nom-verbe) en kurde (dialecte sorani) [in French]*. PhD thesis at the Université Paris 3 - Sorbonne Nouvelle.

Hadi Veisi, Mohammad MohammadAmini, and Hawre Hosseini. 2020. Toward kurdish language processing: Experiments in collecting and processing the asosoft text corpus. *Digital Scholarship in the Humanities*, 35(1):176–193.

Thanh Vu, Dat Quoc Nguyen, Mark Dras, Mark Johnson, et al. 2018. VnCoreNLP: A Vietnamese Natural Language Processing Toolkit. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 56–60.

Géraldine Walther and Benoît Sagot. 2010. Developing a large-scale lexicon for a less-resourced language: General methodology and preliminary experiments on Sorani Kurdish. In *Proceedings of the 7th SaLTMiL Workshop on Creation and use of basic lexical resources for less-resourced languages (LREC 2010 Workshop)*.

Géraldine Walther, Benoît Sagot, and Karën Fort. 2010. Fast development of basic nlp tools: Towards a lexicon and a pos tagger for Kurmanji Kurdish. In *International conference on lexis and grammar*, page 0.

Rasty Yaseen and Hossein Hassani. 2018. Kurdish optical character recognition. *UKH Journal of Science and Engineering*, 2(1):18–27.

Rina D Zarro and Mardin A Anwer. 2017. Recognition-based online Kurdish character recognition using hidden Markov model and harmony search. *Engineering Science and Technology, an International Journal*, 20(2):783–794.

Housam Ziad, John Philip McCrae, and Paul Buitelaar. 2018. Teanga: a linked data based platform for natural language processing. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

A Appendix

Reference	Year	Field	open-source	applicable	dialects
(Mohammed et al., 2012)	2012	Dialectology	no	no	Sorani
(Esmaili and Salavati, 2013)	2013	Dialectology	no	yes	Sorani, Kurmanji
(Hassani and Medjedovic, 2016)	2016	Dialectology	no	yes	Sorani, Kurmanji
(Malmasi, 2016)	2016	Dialectology	yes	yes	Sorani
(Al-Talabani et al., 2017)	2017	Dialectology	no	yes	Sorani, Kurmanji, Gorani
(Littell et al., 2016)	2016	Information retrieval and Text mining	no	yes	Sorani
(Hassani, 2017b)	2017	Information retrieval and Text mining	yes	yes	Sorani, Kurmanji
(Esmaili, 2012)	2012	Information retrieval and Text mining	no	no	Sorani
(Esmaili et al., 2014)	2014	Information retrieval and Text mining	yes	yes	Sorani, Kurmanji
(Jaf, 2016)	2016	Information retrieval and Text mining	no	yes	Sorani
(Rashid et al., 2017a)	2017	Information retrieval and Text mining	no	yes	Sorani
(Rashid et al., 2017b)	2017	Information retrieval and Text mining	no	yes	Sorani
(Ahmadi, 2019)	2019	Information retrieval and Text mining	yes	no	Sorani
(Saeed et al., 2018b)	2018	Information retrieval and Text mining	no	yes	Sorani
(Saeed et al., 2018b)	2018	Information retrieval and Text mining	no	yes	Sorani
(Mustafa and Rashid, 2018)	2018	Information retrieval and Text mining	no	yes	Sorani
(Saeed et al., 2018a)	2018	Information retrieval and Text mining	no	no	Sorani
(Ahmadi et al., 2020)	2020	Lexical resources	yes	yes	Sorani
(Esmaili et al., 2013)	2013	Lexical resources	yes	yes	Sorani
(Aliabadi et al., 2014)	2014	Lexical resources	yes	yes	Sorani
(Aliabadi, 2014)	2014	Lexical resources	no	yes	Sorani
(Veisi et al., 2020)	2020	Lexical resources	yes	yes	Sorani
(Ahmadi et al., 2019)	2019	Lexical resources	yes	yes	Sorani, Kurmanji, Gorani
(Abdulrahman et al., 2019)	2019	Lexical resources	yes	yes	Sorani
(Abdulrahman and Hassani, 2020)	2020	Lexical resources	yes	yes	Sorani
(Ataman, 2018)	2018	Lexical resources	yes	yes	Kurmanji
(Hassani, 2017a)	2017	Machine Translation	no	yes	Sorani, Kurmanji
(Kaka-Khan, 2018)	2018	Machine Translation	no	yes	Sorani
(Walther and Sagot, 2010)	2010	Morphological and syntactic analysis	yes	yes	Sorani
(Walther et al., 2010)	2010	Morphological and syntactic analysis	yes	yes	Kurmanji
(Salavati et al., 2013)	2013	Morphological and syntactic analysis	yes	yes	Sorani
(Jaf and Ramsay, 2014)	2014	Morphological and syntactic analysis	no	yes	Sorani
(Jaf and Ramsay, 2016)	2016	Morphological and syntactic analysis	no	yes	Sorani
(Gökırmak and Tyers, 2017)	2017	Morphological and syntactic analysis	yes	yes	Kurmanji
(Salavati and Ahmadi, 2018)	2018	Morphological and syntactic analysis	no	yes	Sorani
(Mustafa and Rashid, 2018)	2018	Morphological and syntactic analysis	no	yes	Sorani
(Ahmadi and Hassani, 2020a)	2020	Morphological and syntactic analysis	no	yes	Sorani
(Mohammed, 2012)	2012	Optical character recognition	no	no	Sorani
(Mohammed, 2013)	2013	Optical character recognition	no	yes	Sorani
(Shaltooki and Hama, 2016)	2016	Optical character recognition	no	yes	Sorani
(Zarro and Anwer, 2017)	2017	Optical character recognition	no	yes	Sorani
(Yaseen and Hassani, 2018)	2018	Optical character recognition	no	yes	Sorani
(Dinler and Aydin, 2018b)	2018	Optical character recognition	no	yes	Sorani
(Kaka-Khan, 2017)	2017	Other	no	yes	Sorani
(Hashim and Alizadeh, 2018)	2018	Sign language recognition	no	yes	Sorani
(Kamal and Hassani, 2020)	2020	Sign language recognition	yes	yes	Sorani
(Daneshfar et al., 2009)	2009	Speech recognition	no	yes	Sorani
(Barkhoda et al., 2009)	2009	Speech recognition	no	no	Sorani
(Bahrampour et al., 2009)	2009	Speech recognition	no	yes	Sorani
(Hassani and Kareem, 2011)	2011	Speech recognition	no	yes	Sorani
(Dinler and Karabıber, 2017)	2017	Speech recognition	no	no	Kurmanji
(Dinler and Aydin, 2018a)	2018	Speech recognition	no	yes	Sorani, Kurmanji
(Qader and Hassani, 2019)	2019	Speech recognition	yes	yes	Sorani

Table A.2: Classification of the publications in the field of Kurdish language processing

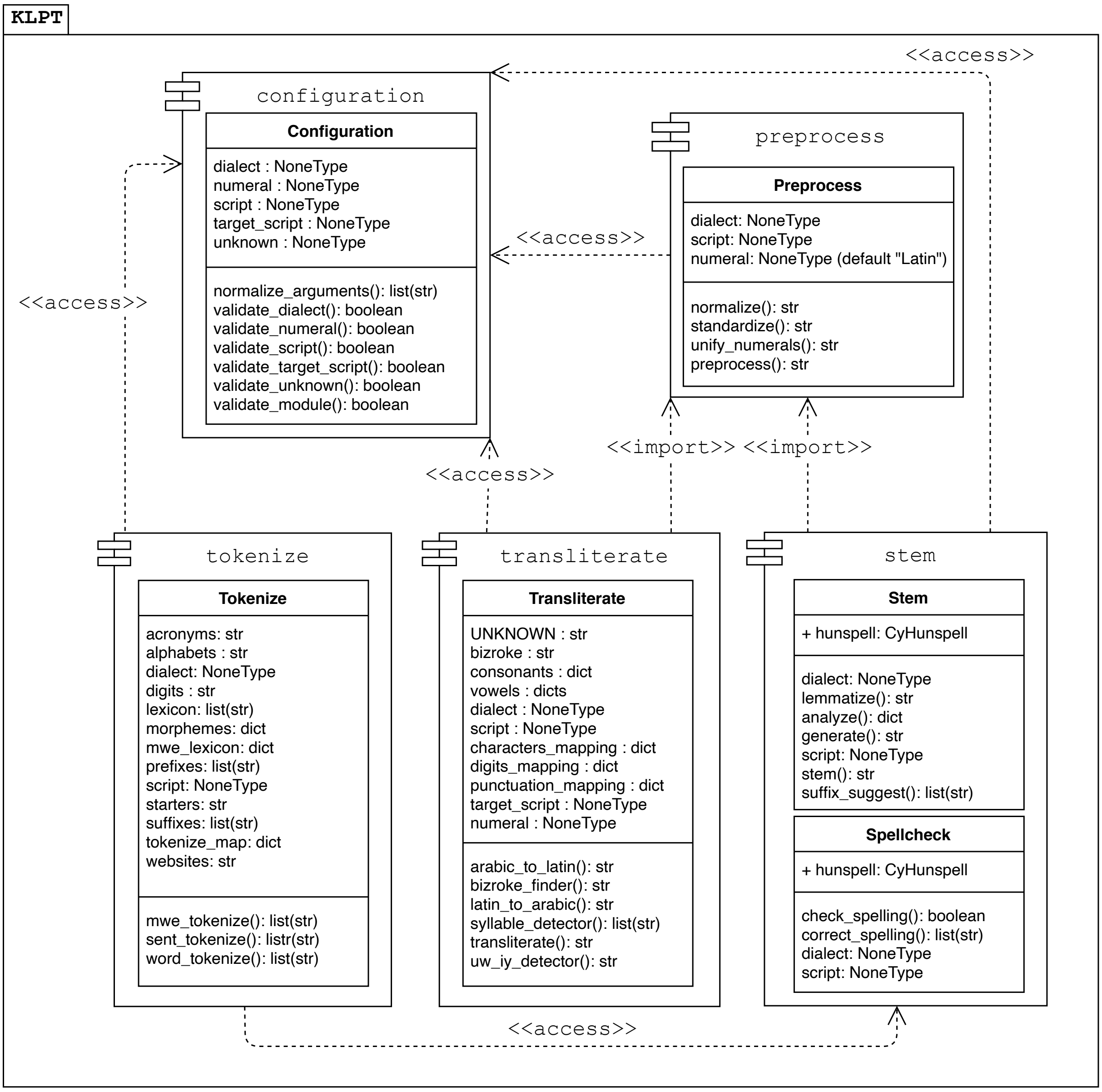


Figure A.5: The Package and class models of KLPT in the Unified Modeling Language (UML)