Computer-Assisted Language Comparison in Practice

Tutorials on Computational Approaches to the History and Diversity of Languages

Making a Lexibank Dataset from Lee's "Phonological Features of Caijia" from 2023

Johann-Mattis List
Chair for Multilingual Computational Linguistics
University of Passau

Caijia is a very interesting Sino-Tibetan language variety. It has been documented only recently, it seems to belong to the Sinitic branch of Sino-Tibetan, but shows some archaic features that have led to some controversies among scholars regarding its proper affiliation, and detailed comparative analyses of the language in comparison with other Sino-Tibetan languages are still in their infancy. This little study demonstrates how a first published wordlist of Caijia (Lee 2023) can be prepared for the inclusion in the Lexibank repository.

1 Introduction

According to Lee (2023), Caijia is a Sino-Tibetan language that has only recently been discovered. First documentation efforts of the language reach back to the 1980s, and scholars have discussed since then, where in the branch of Sino-Tibetan languages to place the language variety. Lee provides evidence for the close affiliation of Caijia with the Sinitic branch of Sino-Tibetan, consisting of the Chinese dialects. Lee's study is accompanied by a Caijia wordlist that provides word forms for the majority of the 250 concepts that were used to assemble the data for the phylogenetic study of Sino-Tibetan by Sagart et al. (2019).

Lee archived the data on Zenodo and provided an additional analysis of the data by segmenting the phonetic transcription into speech sounds. This way of handling the data makes it very easy for us to conver the dataset to the formats required by the Cross-Linguistic Data Formats initiative (CLDF, Forkel et al. 2018). Since CLDF itself is the core format for wordlists used in Lexibank (Blum et al. 2025), the large repository providing standardized access to numerous mono- and multilingual wordlists for several

Computer-Assisted Language Comparison in Practice
Volume 8, Number 1
URL: https://calc.hypotheses.org/8581 DOI: 10.15475/calcip.2025.1.6

Published on 25/06/2025

thousand of the world's languages, it is straightforward to prepare Lee's Caijia data for inclusion in Lexibank. In the following, I will quickly show how this can be done.

2 Initial Preparations

In order to create a CLDF dataset that is compatible with Lexibank, we start by creating a couple of basic files. These include the folder `raw`, in which we place the raw data, the file `metadata.json`, which contains the basic information on the data, a BibTex file containing the BibTex-entry for the original source, also placed into the folder `raw`, and a MarkDown-file `contributors.md`, listing the contributors in tabular form. Lee's original supplement is archive with Zenodo in the form of an Excel file. From this file, I manually extracted the basic sheet and stored it in tab-separated format in a file called `data.tsv`, placed in the folder raw. The metadata file looks as follows.

```
{
  "id": "leecaijia",
  "title": "CLDF dataset derived from Lee's
           \"Phonological Features of Caijia\" from
           2023",
  "license": "CC-BY-4.0",
  "url": "https://doi.org/10.5281/zenodo.5544225",
  "citation": "Lee, Man Hei (2023): Phonological features
              of Caijia that are notable from a
              diachronic perspective.
              Journal of Historical Linguistics.
              DOI:
              https://doi.org/10.1075/jhl.21025.lee",
  "conceptlist": "Sagart-2019-250"
}
```

From this file, we can see that the ID that we give to the dataset is leecaijia, we also add a stanardized title for the dataset, that corresponds to most titles that we coin for CLDF datasets in Lexibank. These contain the name of the original author, the year of the original publication, and a short title of the original publication. As license, we choose CC-BY-4.0, the common license used by Zenodo. The citation cites the original publication in human-readable form, and the URL provides a link to the DOI of the Zenodo archive. Finally, we add as concept list for the Concepticon project (List et al. 2025) the identifier of the concept list that Lee used for the Caijia data.

The BibTeX file looks as follows, providing the major information on the original publication in computer-readable format.

```
@article{Lee2023,
    doi = {10.1075/jhl.21025.lee},
    url = {https://doi.org/10.1075/jhl.21025.lee},
    year = {2023},
    author = {Lee, Man Hei},
    volume = {13},
    number = {1},
    pages = {82-114},
    title = {Phonological features of Caijia that are
             notable from a diachronic perspective},
    journal = {Journal of Historical Linguistics}
}
```

The file contributors.md lists the contributors for the dataset. These will be processed by the Lexibank conversion routine and defines how data are handled on Zenodo. It is therefore important to be clear about the roles that different people play in the creation of datasets. Our file looks as shown below.

```
# Contributors

Name               | GitHub user | Description      | Role
 ---               | ---         | ---              | ---
Lee, Man Hei       |             |author, fieldwork | Author
Johann-Mattis List | @LinguList  | maintainer       | Editor
```

What is important here is that we assign the role of the major author to the original data creator (Man Hei Lee). I placed myself as an editor of the dataset, in order to indicate that I am the person responsible for the CLDF edition of the data. This shows that the CLDF version of the data is treated as a new edition, with the editor being responsible for certain aspects of this edition (e.g. the transparent standardization of the data).

3 Converting the Data to CLDF

3.1 Initial Preparations

In order to convert the data to CLDF, we must add Python files to the repository. We first create a setup.py file that allows us to install the data as a software package (while we usually do not need to do this, the file helps us to install all necessary packages later on and facilitates the integration).

```
from setuptools import setup
import json

with open("metadata.json", encoding="utf-8") as fp:
    metadata = json.load(fp)

setup(
    name='lexibank_leecaijia',
    py_modules=['lexibank_leecaijia'],
    include_package_data=True,
    url=metadata.get("url",""),
    zip_safe=False,
    entry_points={
        'lexibank.dataset': [
            'leecaijia=lexibank_leecaijia:Dataset',
        ]
    },
    install_requires=[
        "pylexibank&amp;amp;gt;=3.0.0"
    ],
    extras_require={
        'test': [
            'pytest-cldf',
        ],
    },
)
```

In this file, the only information that varies is the name of the dataset (which will be given the namespace `lexibank_leecaijia` from inside Python, after successful installation of the dataset as a Python package. To allow for a successful integration of tests -- making sure that basic aspects of the data remain unchanged -- we furthermore add the file `setup.cfg`, offering information on the integration of the pytest package that we use to validate CLDF data interactively (the content of this file remains unchanged across all CLDF datasets).

```
[tool:pytest]
testpaths = test.py
addopts =
    --cldf-metadata=cldf/cldf-metadata.json
```

The test itself can be written to the file test.py. Here, we typically use some very basic tests that check for the number of word forms (this test must be adjusted to individual datasets).

```
def test_valid(cldf_dataset, cldf_logger):
    assert cldf_dataset.validate(log=cldf_logger)
```

```
def test_forms(cldf_dataset):
    assert len(list(cldf_dataset["FormTable"])) == 238


def test_parameters(cldf_dataset):
    assert len(list(cldf_dataset["ParameterTable"])) == 234


def test_languages(cldf_dataset):
    assert len(list(cldf_dataset["LanguageTable"])) == 1
```

3.2 Installing Packages

In order to install CLDFBench (https://pypi.org/project/cldfbench, Forkel and List 2020) along with the PyLexibank plugin (https://pypi.org/project/pylexibank, Forkel et al. 2024), you must ensure to have created a fresh virtual environment in Python and activated it properly. Once this has been done, we can install both packages by installing `pylexibank` with the help of the Python package manager (note that instead of the command `python` you may need to type `python3`, depending on your system).

```
$ python -m pip install pylexibank
```

This will install PyLexibank with all its dependencies and enable you to run CLDFBench as a command `cldfbench` from the commandline.

3.3 Preparing the Major Script for CLDF Conversion

We can now prepare the major Lexibank script. This script is called `lexibank_leecaijia.py` and it consists of one basic class `Dataset` that handles the major conversion through individual commands. I will present the script in individual blocks. First, we import the libraries that we need for the conversion.

```
import pathlib
import attr
from clldutils.misc import slug
from pylexibank import Dataset as BaseDataset
```

Among the imports, `pathlib` is needed for internal reasons, `attr` as well, and `slug` is needed for the formatting of the concept identifiers. The major import that handles the conversion is the `Dataset` class from `pylexibank` that we import with the alias `BaseDataset`.

```
class Dataset(BaseDataset):
    dir = pathlib.Path(__file__).parent
    id = "leecaijia"
    writer_options = dict(keep_languages=False,
```

```
                                    keep_parameters=False)
```

The core of this class is the method `cmd_makecldf`. This method is triggered when passing the script name as argument from the commandline along with the command `cldfbench lexibank.makecldf` that was activated when installing CLDFBench with the PyLexibank plugin. The method typically must take care of four distinct tasks. It must (1) add the sources from the BibTeX file, (2) standardize the concepts by creating a concept list that is linked to Concepticon, (3) standardize the list of languages by linking languages to Glottolog (https://glottolog.org, Hammarström et al. 2025), and (4) adding individual word forms to the form table, preferably in segmented form with phonetic transcriptions conforming to the standard version of the International Phonetic Alphabet that we defined as part of the Cross-Linguistic Transcription Systems project (https://clts.clld.org, List et al. 2022).

We start by adding the sources to the command.

```
    def cmd_makecldf(self, args):
        # (1) add bib
        args.writer.add_sources()
        args.log.info("added sources")
```

As can be seen, the integrated functionalities passed with the writer class as an attribute of the variable args offers us convenient basic functions to handle data automatically. In this case, the file raw/sources.bib is simply copied to the newly created folder cldf, once running the CLDF conversion. We use the integrated logger args.log to provide information on the status of the conversion.

We can now add the concepts as our second step of the CLDF conversion.

```
        # (2) add concepts
        concepts = {}
        for concept in self.conceptlists[0].concepts.values():
            idx = concept.id.split("-")[-1] + "_" +
slug(concept.english)
            args.writer.add_concept(
                    ID=idx,
                    Name=concept.english,
                    Concepticon_ID=concept.concepticon_id,
                    Concepticon_Gloss=concept.concepticon_gloss
                    )
            concepts[concept.id] = idx
        args.log.info("added concepts")
```

Here, we define a dictionary concepts that takes as key the ID of the original concept list by Sagart et al. (2019) and as value the internal identifier for concepts that we use in the table for concepts (file `cldf/parameters.csv`, after successful CLDF

creation), and the form table (file `cldf/forms.csv` after successful CLDF creation). The reason is that Lee's data uses the Concepticon ID to identify the concepts, allowing us to use it as the basic information from which we link the internal identifier for concepts inside the CLDF dataset that we want to create to the external reference catalog Concepticon with its Concepticon IDs and Concepticon Glosses as reference points. We use the function `writer.add_concept` to add the concept to the CLDF dataset (this creates one row of the parameter table that stores the concepts along with links to Concepticon). Access to the content of the Concepticon concept list by Sagart et al. (2019) is established via the information in the file `metadata.json`, that provides the identifier of the concept list that we can now access via the attribute conceptlists, where we find the concept list in position 0 (`self.conceptlists[0]`).

In order to add the languages to a CLDF dataset, we often proceed in a similar fashion, adding languages in a loop. In this case, however, there is but one language, so we can add the language without an extra loop.

```
        # (3) add language
        args.writer.add_language(
                ID="Caijia",
                Glottocode="caij1234",
                Name="Caijia"
                )
        args.log.info("added languages")
```

We can now begin adding the forms. Here, we must carry out some extra operations, as we must correct some of the entries in Lee's original data file. Lee does not use the superscript [ʰ] to indicate aspiration, but instead uses a plain [h]. Furthermore, Lee writes tones as normal numbers, while CLDF requires tone numbers as superscripts. We therefore start from creating a dictionary that corrects the sounds in question in an individual manner.

```
        # (4) add forms
        # provide corrections for the data
        corrections = {
                "tsh": "tsʰ",
                "tɕh": "tɕʰ",
                "kh": "kʰ",
                "th": "tʰ",
                "ph": "pʰ",
                }
```

We now read in the data (again using predefined functionalities from CLDFBench and PyLexibank).

```
        # read in data
```

```
        data = self.raw_dir.read_csv(
                "data.tsv",
                delimiter="\t", dicts=True)

        # extend corrections
        for i, row in enumerate(data):
            form = row["Caijia (segments  separated)"]
            for s, t in zip("12345", "¹²³⁸⁹"):
                form = form.replace(s, t)
            row["Segments"] = [corrections.get(c, c) for c in
                               form.split()]
```

Now, we can add the data in a final loop. Note that the source key (`Lee2023`) must be identical with the key assigned in the BibTeX file. The `Parameter_ID` that we pass is the internal concept identifier that we created above, and since we only deal with one language variety, we pass our identifier Caijia directly as `Language_ID`.

```
        # add data
        for entry in data:
            if entry["Id"].strip():
                cid = entry["Id"]
            if entry["Caijia"].strip() != "/":
                args.writer.add_form_with_segments(
                        Language_ID="Caijia",
                        Parameter_ID=concepts[cid],
                        Value=entry["Caijia"],
                        Form=entry["Caijia"],
                        Segments=entry["Segments"],
                        Source="Lee2023"
                        )
```

4 Running the CLDF Conversion

In order to run the CLDF conversion routine, we only need to open a terminal file in the folder where the data and the additional scripts and files reside and then type the conversion command from the commandline. Before, however, we must make sure that we have access to the three reference catalogs Glottolog, Concepticon, and CLTS. The easiest way to guarantee this access is to run a command that downloads the data and stores their location in a configuration file.

```
$ cldfbench catconfig
```

If this does not work, you must download the catalogs individually (information can be found in the supplement to Blum et al. 2025) and point to their locations from the

shell command. With all reference catalogs at our disposal, we can now run the conversion procedure.

```
$ cldfbench lexibank.makecldf --glottolog-version=v5.2.1
--concepticon-version=v3.4.0 --clts-version=v2.3.0
lexibank_leecaijia.py
```

This will create something similar to the following output.

```
INFO    running _cmd_makecldf on leecaijia ...
INFO    added sources
INFO    added concepts
INFO    added languages
INFO    file written: /home/mattis/data/datasets/lexibank-
new/leecaijia/cldf/.transcription-report.json
INFO    Summary for dataset /home/mattis/data/datasets/lexibank-
new/leecaijia/cldf/cldf-metadata.json
- **Varieties:** 1 (linked to 1 different Glottocodes)
- **Concepts:** 234 (linked to 234 different Concepticon concept
sets)
- **Lexemes:** 238
- **Sources:** 1
- **Synonymy:** 1.02
- **Invalid lexemes:** 0
- **Tokens:** 1,057
- **Segments:** 42 (0 BIPA errors, 0 CLTS sound class errors, 42
CLTS modified)
- **Inventory size (avg):** 42.00
INFO    file written: /home/mattis/data/datasets/lexibank-
new/leecaijia/TRANSCRIPTION.md
INFO    file written: /home/mattis/data/datasets/lexibank-
new/leecaijia/cldf/lingpy-rcParams.json
INFO    ... done leecaijia [20.0 secs]
```

From this output, we can see that the conversion was successful. Importantly, the phonetic transcriptions, the concept mappings, and the linking of the language to Glottolog all passed the test. The command creates in addition a `.zenodo.json` file that can be used to provide important citation information (as defined in `metadata.json`) as well as information on the author and the editor of a given dataset to Zenodo, when uploading the data there (https://zenodo.org).

5 Conclusion

The data prepared in this sample study is but a single wordlist file, but it offers us many interesting possibilities for further research. On the one hand, every additional language

that we add to big databases like CLICS (Tjuka et al. 2025) and Lexibank (Blum et al. 2025) brings us closer to a satisfying coverage of the world's languages. More importantly, however, Lee's data gives us access to a language variety whose origin has not been completely established so far and is still disputed by various scholars. Given that the data is readily coded in CLDF now, it would be easy to use computational methods, such as the approach by Blum et al. (2025), in order to check how the Caijia will be classified by automated approaches. Additionally, the data could be easily integrated in the database by Sagart et al. (2019) and properly annotated with the help of computer-assisted tools (List et al. 2025). No matter what study one wants to carry out with the data, the fact that Lee shared the wordlist in such a clean format made it extremely easy to integrate the data into Lexibank. Having been integrated with the thousands of wordlist that are already part of the repository, several additional investigations of Caijia have now been facilitated.

References

Blum, Frederic and Herbold, Steffen and List, Johann-Mattis (forthcoming): From Isolates to Families: Using Neural Networks for Automated Language Affiliation. In: Proceedings of the Association for Computational Linguistics 2025. 1-12. https://doi.org/10.48550/arXiv.2502.11688

Blum, Frederic and Barrientos, Carlos and Englisch, Johannes and Forkel, Robert and Greenhill, Simon J. and Rzymski, Christoph and List, Johann-Mattis (forthcoming): Lexibank 2: pre-computed features for large-scale lexical data [version 1; peer review: 3 approved]. Open Research Europe 5.126. 1-19. https://doi.org/10.12688/openreseurope.20216.1

Forkel, Robert and List, Johann-Mattis and Greenhill, Simon J. and Rzymski, Christoph and Bank, Sebastian and Cysouw, Michael and Hammarström, Harald and Haspelmath, Martin and Kaiping, Gereon A. and Gray, Russell D. (2018): Cross-Linguistic Data Formats, advancing data sharing and re-use in comparative linguistics. Scientific Data 5.180205. 1-10. https://www.nature.com/articles/sdata2018205

Forkel, Robert and List, Johann-Mattis (2020): CLDFBench. Give your Cross-Linguistic data a lift. In: Proceedings of the Twelfth International Conference on Language Resources and Evaluation. 6997-7004. http://www.lrec-conf.org/proceedings/lrec2020/pdf/2020.lrec-1.864.pdf

Forkel, Robert and Simon J Greenhill and Hans-Jörg Bibiko and Christoph Rzymski and Tiago Tresoldi and List, Johann-Mattis (2024): PyLexibank. The Python curation library for Lexibank [Software Library, Version 3.5.0]. Geneva: Zenodo. https://pypi.org/project/pylexibank/

Hammarström, Harald, Martin Haspelmath, Robert Forkel, and Sebastian Bank. 2025. *Glottolog [Dataset, Version 5.2.1]*. Leipzig: Max Planck Institute for Evolutionary Anthropology. https://glottolog.org

Lee, Man Hei (2023): Phonological features of Caijia that are notable from a diachronic perspective. Journal of Historical Linguistics . 13.1. 82-141. https://doi.org/10.1075/jhl.21025.lee

List, Johann-Mattis and Anderson, Cormac and Tresoldi, Tiago and Forkel, Robert (2022): Cross-Linguistic Transcription Systems. Version 2.3.0. Jena: Max Planck Institute for the Science of Human History. https://clts.clld.org

List, Johann-Mattis and Tjuka, Annika and Blum, Frederic and Kučerová, Alžběta and Barrientos Ugarte, Carlos and Rzymski, Christoph and Greenhill, Simon J. and Robert Forkel (2025): CLLD Concepticon [Dataset, Version 3.4.0]. Leipzig: Max Planck Institute for Evolutionary Anthropology. https://concepticon.clld.org

List, Johann-Mattis and van Dam, Kellen Parker and Blum, Frederic (2025): EDICTOR 3. An Interactive Tool for Computer-Assisted Language Comparison [Software Tool, Version 3.1]. Passau: MCL Chair at the University of Passau. https://edictor.org

Sagart, Laurent and Jacques, Guillaume and Lai, Yunfan and Ryder, Robin and Thouzeau, Valentin and Greenhill, Simon J. and List, Johann-Mattis (2019): Dated language phylogenies shed light on the ancestry of Sino-Tibetan. Proceedings of the National Academy of Science of the United States of America 116. 10317-10322. https://doi.org/10.1073/pnas.1817972116

Tjuka, Annika and Forkel, Robert and Rzymski, Christoph and List, Johann-Mattis (2025): Advancing the Database of Cross-Linguistic Colexifications with New Workflows and Data. arXiv 2503.11377. 1-14. [Preprint, under review, not peer-reviewed] https://doi.org/10.48550/arXiv.2503.11377

Supplementary Material

Code and data are curated on GitHub (https://github.com/lexibank/leecaijia, Version 1.3) and archived with Zenodo (https://doi.org/10.5281/zenodo.15687121).

Funding Information

This project has received funding from the European Research Council (ERC) under the European Union's Horizon Europe research and innovation programme (Grant agreement No. 101044282). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.