Computer-Assisted Language Comparison: State of the Art

By comparing the languages of the world, we gain invaluable insights into human prehistory, predating the appearance of written records by thousands of years. The traditional methods for language comparison are based on manual data inspection. With more and more data available, they reach their practical limits. Computer applications, however, are not capable of replacing experts' experience and intuition. In a situation where computers cannot replace experts and experts do not have enough time to analyse the massive amounts of data, a new framework, neither completely computer-driven, nor ignorant of the help computers provide, becomes urgent. Such frameworks are well-established in biology and translation, where computational tools cannot provide the accuracy needed to arrive at convincing results, but do assist humans to digest large data sets. In this talk, we will illustrate what we consider the current state of the art of computer-assisted language comparison, by presenting a workflow that starts from raw data and leads up to a stage where sound correspondence patterns across multiple languages have been identified and can be readily presented, inspected, and discussed. We illustrate this workflow with help of a dataset on Hmong-Mien languages, which has so far not yet been analyzed in this way. Our illustration is furthermore accompanied by Python code and instructions on how to make use of additional web-based tools we developed, so that users can replicate our workflow or apply it for their own purposes.

1 Introduction

1.1 The Gap between Computational and Traditional Historical Linguistics

The proposal of new, fancy, and shiny quantitative methods applied to handle problems in historical linguistics has created a gap between what one could call "classical" approaches to historical language comparison and the "new and innovative" automatic approaches. Classical linguists are often skeptical of the new approaches, partly because the results differ from those achieved by classical methods (Anthony and Ringe 2015, Holm 2007), but also because the majority of the new approaches work in a black box fashion and do not allow inspecting the concrete findings in detail. Computational linguists, on the other hand, complain about classical historical linguists' lack of consistency when applying the classical methods.

1.2 Computer-Assisted Disciplines

The use of computer applications in historical linguistics is steadily increasing. With more and more data available, the classical methods reach their practical limits. At the same time, computer applications are not capable of replacing experts' experience and intuition, especially when data are sparse. If computers cannot replace experts and experts do not have enough time to analyse the massive amounts of data, a new framework is needed, neither completely computer-driven, nor ignorant of the assistance computers afford. Such computer-*assisted* frameworks are well-established in biology and translation. Current machine translation systems, for example, are efficient and consistent, but they are by no means accurate, and no one would use them in place of a trained expert. Trained experts, on the other hand, do not necessarily work consistently and efficiently. In order to enhance both the quality of machine translation and the efficiency and consistency of human translation, a new paradigm of computer-assisted translation has emerged (Barrachina et al. 2008: 3).

1.3 Computer-Assisted Language Comparison

Following the idea of computer-assisted frameworks in translation and biology, a framework for computer-assisted language comparison (CALC) could be the key to reconcile classical and computational approaches in historical linguistics. Computational approaches may still not be able to compete with human experts, but when used to pre-process the data with human experts systematically correcting the results, they can drastically increase both the efficiency and the consistency of the classical comparative method.

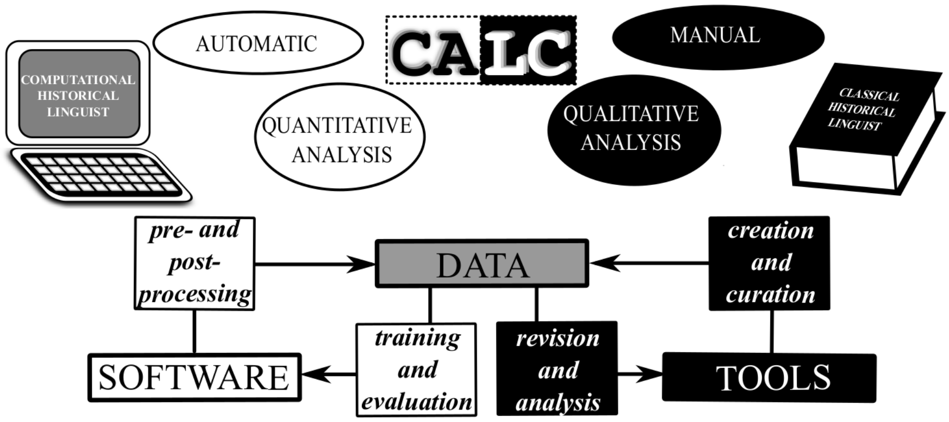


Figure 1: Basic idea of data managment within the CALC framework.

The basic idea behind computer-*assisted* as opposed to computer-*based* language comparison is to allow scholars to do qualitative and quantitative research are done at the same time. In order to allow scholars to do this, **data must always be available in *machine-* and *human-readable* form.** Figure 1 shows a tentative workflow for the CALC framework, in which data is constantly passed back and forth between computational and classical linguists.

Three different aspects are essential for this workflow:

(a) New software allows for the application of transparent methods which increase the accuracy and the application range of current methods and also treat the peculiarities of specific language families (like, e.g., Sino-Tibetan).

(b) Interactive tools provide an interface between human and machine, allowing experts to correct errors and to inspect the automatically produced results in detail.

(c) Specific data is used to test and train the software algorithms.

2 Workflows for Computer-Assisted Language Comparison

2.1 Overview

Our workflows for computer-assisted language comparison have so far been intensively tested on a small set of 8 Burmish languages, which we investigated in collaboration with Nathan W. Hill, who was responsible for the qualitative investigation of the data and for the common discussion of new computer-assisted methods which were then implemented by Johann-Mattis List (see Hill and List 2017 for an exemplary discussion of some of the new approaches). Our experience with the Burmish project by now allows us to set up a first workflow that starts from raw data and leads up to the explicit identification of correspondence patterns across multiple languages. At the moment, List and Hill develop the workflow further to account also for (semi)-automatic reconstructions, but in this talk, only the identification of correspondence patterns will be discussed.

2.2 Details of the Workflow

Our workflow currently comprises 5 different stages, in which we successively lift linguistic data from their raw form in which we can find them in wordlists and tables published in dictionaries and field-work notes, up to a level where correspondence patterns across cognate words have been automatically identified and can be qualitatively inspected by the scholar.

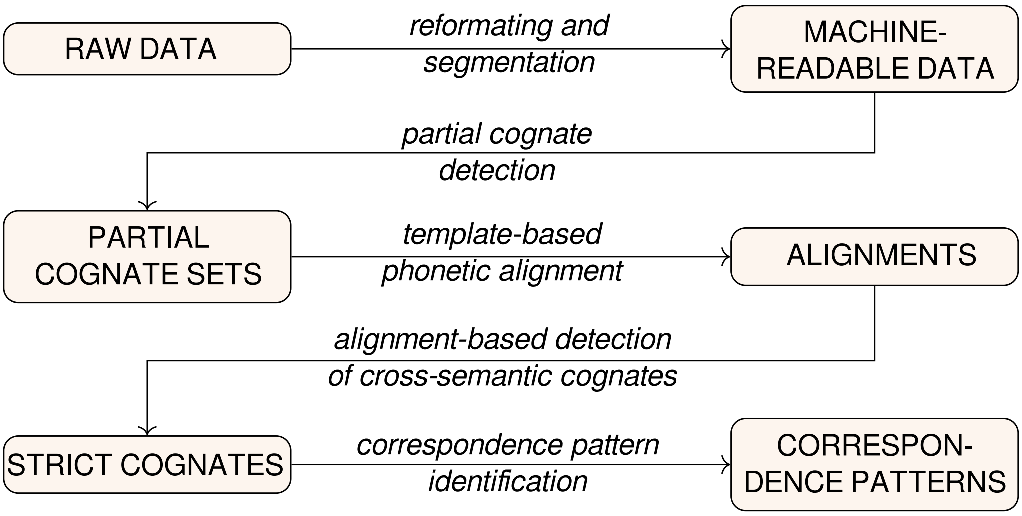


Figure 2: Current state-of-the-art workflow developed in collaboration of different research groups working in computer-assisted frameworks.

Although the workflow can be carried out almost completely without any manual intervention by a linguist, we emphasize that this workflow explicitly *allows* for expert intervention at *any* of the five stages. While, in our experience, specific care is required when lifting the data the first time to machine-readable format, it should further be noted that *all* steps of the workflow profit from human intervention, since none of the automatic methods currently available to us could spot all patterns in linguistic data without over- or underestimating their importance for linguistic reconstruction.

Our workflow starts from *raw data*, including tabular data from fieldwork notes or data published in books and articles, which we re-organize and re-format in such a way that the data can be processed by our tools. Once we have *machine-readable data*, we can use methods for automatic cognate detection (List et al. 2016b) in order to infer *partial cognates* across the languages in our data. Having inferred cognates, we can now also align the data in the cognate sets. While we could use phonetic alignment approaches discussed in the literature (List 2014), we now use a new approach, based on phonotactic templates, which has the advantage of being much faster and accurate when dealing with alignments for South-East-Asian languages. Once having identified the alignments, we start to search automatically for cognates *across* different concepts. Since all automatic methods *need* to start searching for cognates within the same concept slot (otherwise, there would be too many false positives), our new method, which makes used of a systematic comparison of readily aligned cognate sets, systematically searches for cognates independent of their meaning. The improved, cross-semantic cognate sets, which are all readily aligned, have the specific property of being *strict*: no cognate set could compare two morphemes from the same language which would differ in their pronunciation. (List 2018) calls these cognate sets *regular*, but in discussions with Nathan Hill, we decided that *regular* is probably not the best term, as they can well be wrong, so we call them *strict* now. Once strict cognates have been identified, we use the new algorithm for the automatic inference of sound correspondence patterns across multiple languages by List (2019) to infer the correspondence patterns in the data.

In Section 3, we will provide detailed examples how all steps of the workflow interact, using a relatively recent collection of linguistic data on Hmong-Mien languages (Chén 2012) for this purpose.

2.3 Materials and Methods for the Workflow Illustration

The data we use to illustrate our workflow in the next section was originally collected by Chén (ibid.), and later added in digital form to the Wiktionary project. Chén's collection of *frequent terms* (*chángyòng cíbiǎo* 常用词表, pp. 567-862) comprises 885 different concepts translated into 25 varieties of Hmong-Mien. In Figure 3, we contrast one exemplary page from Chéns book with the data as it has been prepared by the Wiktionary users. We can see that the data is essentially the same, but that the rows and columns of the tabular form have been swapped.

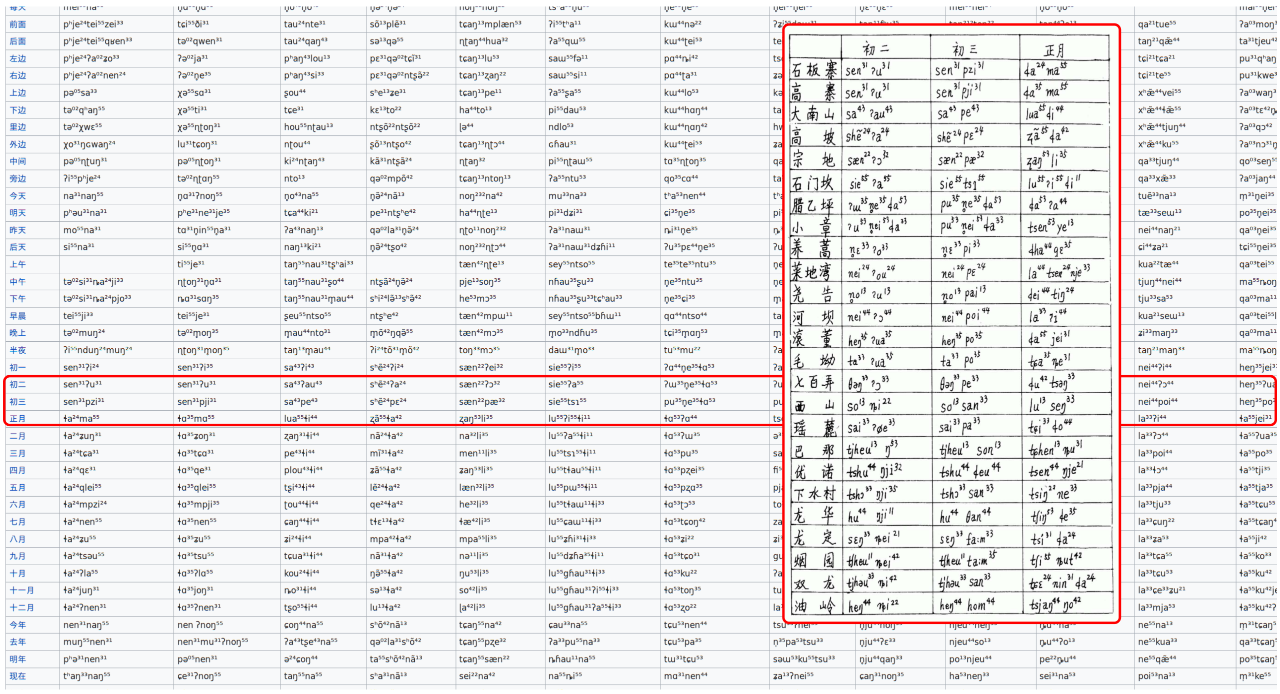


Figure 3: Contrasting Chén's original data with the table in Wiktionary

All methods have either been implemented and published before, or are shared along with the slides and the handout for this talk. Since this is work in progress, however, we warn users that both data and code will be in flux for some time, but we will make sure that both data and code can always be readily analyzed with our tools. All code, the data we use, and installation instructions can be found at `https://github.com/lingpy/calc-workflow`. We ask those interested in testing our methods to use our issue-tracker on GitHub in case they face difficulties of any kind. «««< HEAD In this talk, we present the workflow with a subset of 10 varieties of the Hmong-Mien languages in Chén's sample, for which we selected a subset of 313 concepts. The concepts were selected by checking the overlap with the current 504 concept list of the Burmish Etymological Database project (headed by Nathan W. Hill, data online at `https://dighl.github.io/burmish`). The languages were selected for some general reasons, like lexical coverage, geographic distribution, or basic diversity, but not with the specific "eye" of a historical linguist who would select languages to explore the history of a language family. We would be glad about any additional recommendations, if scholars feel competent to give us advice in this context. The geographic locations are shown in the Figure 4.

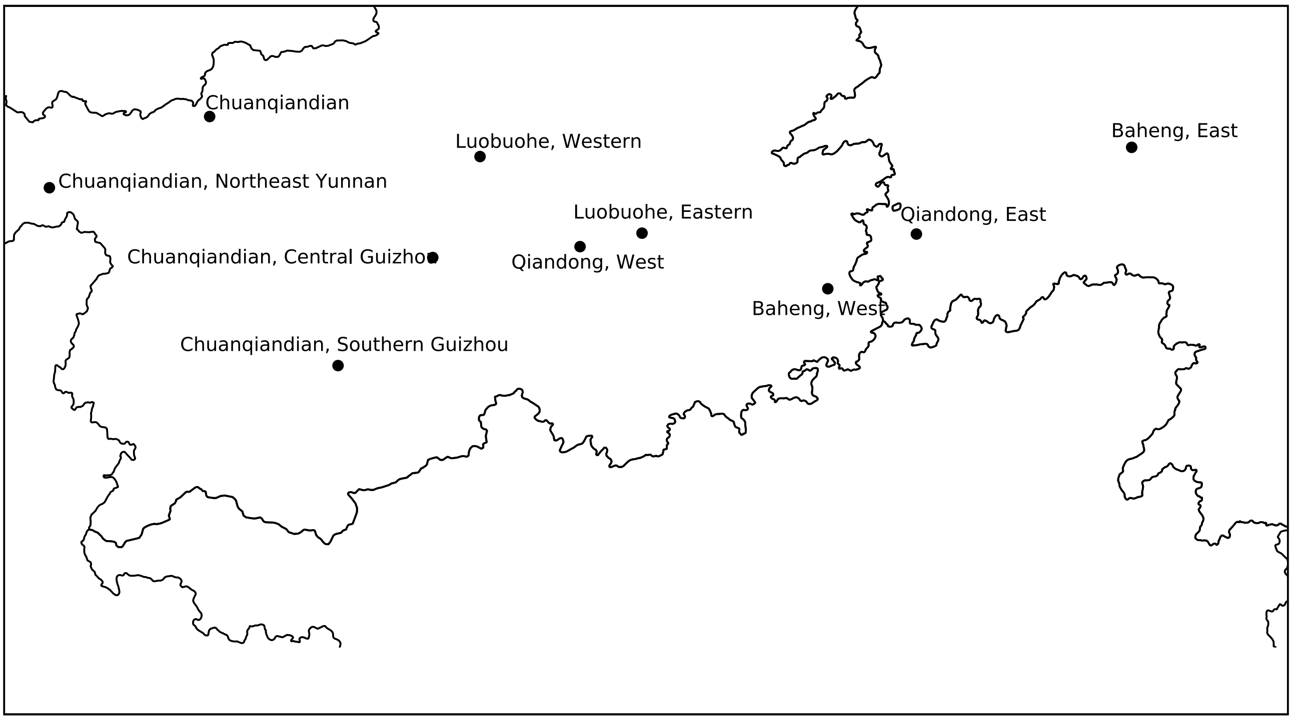


Figure 4: Language geographic locations

3 Illustration of the Workflow

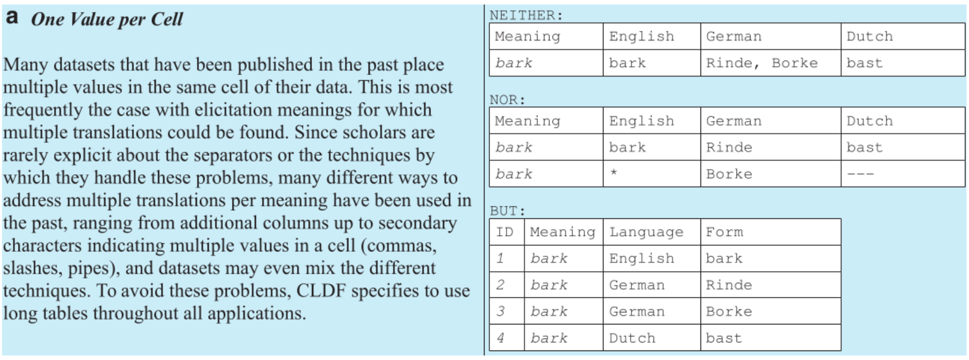
3.1 From Raw Data to Segmented Data

When comparing languages within a computer-assisted framework, with the goal of identifying sound correspondence patterns in the data, we need to make sure that our data is machine-readable at first. If the data is not machine-readable, we can neither use web-based tools like EDICTOR which make it easy to edit the data *manually* (List 2017), nor can we use computational tools, like LingPy (List et al. 2018b), which can help us a great deal in identifying cognate sets and aligning our data.

A first problem for many researchers is to get used to our formats for data representation. In contrast to the typical style used by scholars, we do not use simple tables, with languages in a row and concepts in a column, or vice versa, but instead a so-called long-table format, in which we reserve a *row* in a table for each word, and add a header, which tells us what the cells in each column contain in terms of the data. This long-table format reflects the rule of "One Value per Cell", as stated by the Cross-Linguistic Data Formats initiative (Forkel et al. 2018), reproduced in Figure 5.

As a second rule, we have certain format specifications that make it easier form machines to deal with our input. This includes

- the use of a *segmented* form of IPA transcriptions, in which a space is used to separate distinct sounds from each other, to give the computer direct information on whether symbol combinations are meant to reflect one sound (e.g., affricates, such as [ts, tʃ]), or multiple sounds (compare German *Handschuh* [h a n t ʃ uː] vs. German *Tschüss* [tʃ y s]),
- them use of morpheme segmentation markers (we use a +) to indicate morpheme boundaries, which is straightforward when working with many morpheme-syllabic SEA languages, in which morphemes coincide with syllables,



a ***One Value per Cell***

Many datasets that have been published in the past place multiple values in the same cell of their data. This is most frequently the case with elicitation meanings for which multiple translations could be found. Since scholars are rarely explicit about the separators or the techniques by which they handle these problems, many different ways to address multiple translations per meaning have been used in the past, ranging from additional columns up to secondary characters indicating multiple values in a cell (commas, slashes, pipes), and datasets may even mix the different techniques. To avoid these problems, CLDF specifies to use long tables throughout all applications.

NEITHER:

Meaning	English	German	Dutch
bark	bark	Rinde, Borke	bast

NOR:

Meaning	English	German	Dutch
bark	bark	Rinde	bast
bark	*	Borke	---

BUT:

ID	Meaning	Language	Form
1	*bark*	English	bark
2	*bark*	German	Rinde
3	*bark*	German	Borke
4	*bark*	Dutch	bast

Figure 5: Long-table format instead of condensed formats with multiple values per cell.

- a clear-cut account on the concepts in our data, as they serve as the initial comparanda, so each concept needs to be given a clear-cut definition, and our preferable starting points are concept lists which are translated into the languages to be investigated, as opposed to pre-selected accounts on potential etymological items.

We indicate words in the computer-readable form, by adding a column called `TOKENS` in which data is segmented with a space to distinguish different sounds, and with the plus-symbol to distinguish different morphemes.

Thus, our original data consists of a text-file, separated by tabstop, with the first row serving as a header, and the following rows providing information for one word per language. Our software requires the following columns to be submitted:

- `ID`: numerical identifier, greater than 0,
- `DOCULECT`: name of the language,
- `CONCEPT`: some gloss for the concept,
- `TOKENS`: the morpheme and sound-segmented form of the data.

We recommend also to add a column called `VALUE`, containing the original data, as well as a column `FORM`, which shows the original data but corrected for multiple values per cell. The software usually automatically creates a form `IPA`, which is not necessarily used, but a legacy form that will be replaced by the `FORM` in future updates. Additional values are then consistently added by our workflow and will be discussed later.

We offer procedures to ease the conversion of the data to the required formats. While the creation of long-table formats is usually done by applying a custom script, we use *orthography profiles* to create morpheme-segmented IPA representations for our `TOKENS` column from the original data (Moran and Cysouw 2018). Orthography profiles are a very straightforward way to convert raw data to space-separated IPA representations. An orthography profile can be thought of as a simple text file with two or more columns in which the first represents the values as you find them in your data (i.e., non-IPA transcriptions, etc.), and the other columns allowing you to convert the sequence of characters that you find in the first column. So in brief, you have a source-pattern and a replacement pattern, for example, the one shown in Table 1. With such a replacement pattern, an input string `čashaa` would on the one hand be segmented into `č a sh aa` and at the same time, it would be converted to `tʃ a ʃ aː`. We now offer an online demo of orthography profiles at `http://calc.digling.org/profile`, which can be used to test and apply customized orthography profiles.

Grapheme	IPA
č	tʃ
ž	dʒ
th	tʰ
dh	d̤
sh	ʃ
a	a
aa	aː

Table 1: Very simple orthography profile example.

SUMMARY

- Data must be machine-readable in order to be amenable for computer-assisted analyses.
- Data must specifically be segmented, both with respect to the morpheme boundaries and the boundaries between distinct sounds.
- Data must be provided in form of a *long table* with some specific column headers, providing all relevant information.
- Computer-assisted tools help to prepare the data for computer-assisted processing.

3.2 From Segmented Data to Cognate Sets

Once the data is segmented and provided in the long table format as it is required by the LingPy software package, as described in our tutorial (List et al. 2018a), we can use LingPy's partial cognate detection method to infer partial cognates in our linguistic data. Partial cognates are hereby understood as cognate assessments *per morpheme* in our data, as opposed to cognate assessments *per word*. While it has always been clear to scholars working in the field of South-East Asian linguistics that cognacy should rather be assigned on the level of the morpheme than on the level of full words, given that the high degree of compounding would easily complicate the identification of cognate relations, automatic methods, and specifically phylogenetic reconstruction approaches usually still assume a rather naive one-word-one-cognate relation (List 2016).

In our framework, we explicitly address this problem by adopting a numerical annotation format in which each morpheme instead of each word form is assigned to a specific cognate set (Hill and List 2017). This framework is illustrated in Figure 6, where we contrast word forms for "yesterday" in five Burmish varieties, indicating their detailed "cognate relations". In the first "traditional" style of cognate coding, we would proceed in a *strict* way, only allowing those words which are completely cognate in all their morphemes to be judged as cognates. In the second, *loose* cognate annotation, we judge all words that are in a *connected component* in our shared morpheme network to be cognate, and in the last column, we show our explicit coding of partial cognacy, in which each morpheme is assigned to one cognate set.

The software package LingPy offers a straightforward algorithm to detect and annotate partial cognates in datasets formatted as long tables. This algorithm by List et al. (2016b) uses techniques for automatic sequence comparison to create a network of similar morphemes for each meaning slot in a given dataset. It then filters those concepts in consecutive stages, with the goal of avoiding that two or

Language	Form	Strict	Loose	Exact
Bola	a^{31} ŋji^{35} nɛʔ^{31}	1	1	1 2 3
Lashi	a^{31} ŋjei^{55} nap^{31}	1	1	1 2 3
Rangoon	mɑ^{53} ne^{53} kɑ^{53}	2	1	0 3 0
Xiandao	n̥^{31} m̥an^{35}	3	1	3 4
Achang	man^{35}	4	1	4

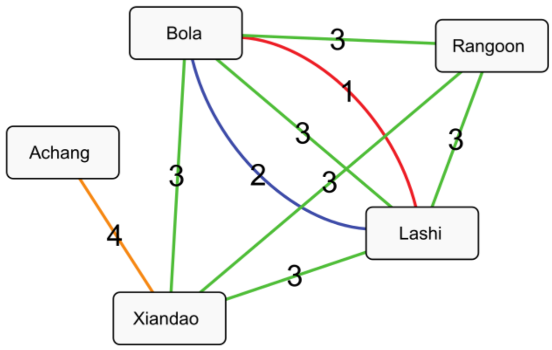


Figure 6: Partial cognacy in Burmish language varieties and different ways of coding (see Hill and List 2017 and further explanations in the main text). coding.

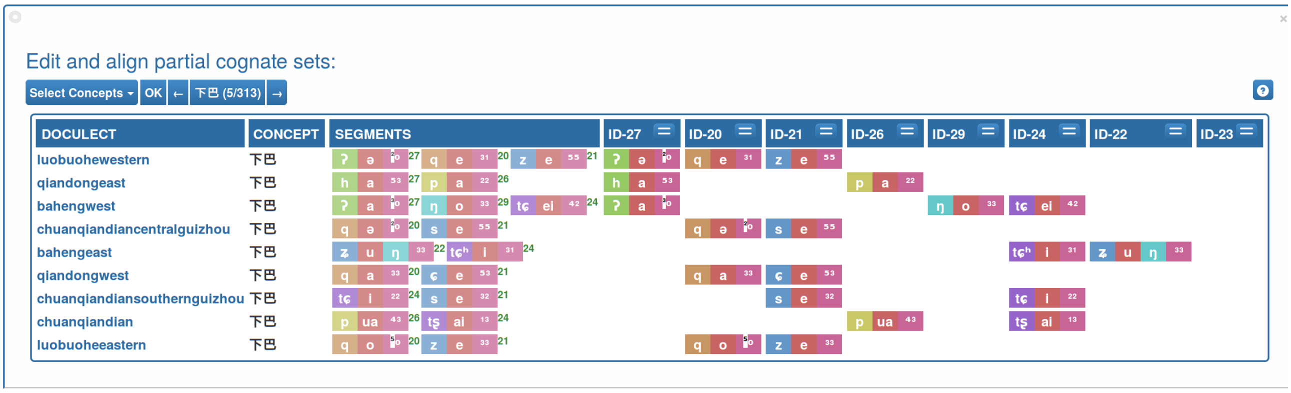


Figure 7: Partial cognate annotation within the EDICTOR tool for the word for “chin” in 10 selected Hmong-Mien varieties.

more morphemes in the same word for the same language are assigned to the same cluster. In the end, the algorithm outputs the cognate judgments in the same format as indicated above in Figure 6, namely, but assigning each morpheme to a given number, with the number representing that cognate set.

Note that this algorithm works quite well, although it is, of course, not infallible. It reaches between 88 and 90 percent on a test datasets consisting of Bai dialects, Chinese dialects, and dialects of Tujia. With more challenging datasets, the scores will surely drop, but we can expect that the automatic cognate detection is in any case *helpful*, as is easier to correct cognates than to assign them from scratch.

In addition to the cognate detection algorithm, the EDICTOR web-based tool for computer-assisted language comparison (List 2017), freely available at `http://edictor.digling.org`, can be used to quickly inspect and correct computer-generated cognate sets, by providing a very convenient interface that allows users to quickly assign morphemes to cognate sets. The interface is illustrated in Figure 7.

SUMMARY

- For a realistic annotation of cognate sets, the annotation of partial cognates, by which morphemes are assigned to cognate sets, is the only realistic choice.
- Partial cognates can be automatically identified with help of software, openly available as part of the LingPy software library (`lingpy.org`, List et al. 2018b) and the algorithm by List et al. (2016b).
- Partial cognates can be annotated consistently with help of the EDICTOR tool (List 2017), online available at `http://edictor.digling.org`.
- Partial cognates in these frameworks are assigned to morphemes occurring in words with the same meaning, both for algorithmic and for practical reasons.

3.3 From Cognate Sets to Alignments

3.4 From Alignments to Cross-Semantic Cognates

As mentioned above in Section 3.2, the partial cognates are only identified for words with the same meaning. This is being done for algorithmic reasons (it would become quite complex to compare all morphemes against each other algorithmically), and for practical reasons, since we believe that it is always better to start from the obvious and save etymologies in historical linguistics, rather than to start from complex ones. Given that semantic shift is a phenomenon for which we dispose of little knowledge with respect to its patterns, we agree explicitly with scholars like Dybo and Starostin (2008) in emphasizing that we should always expect to find clear-cut etymologies within words of the same meaning, even if we know that more etymologies could be find when searching *cross-semantically*, i.e., among words which differ with respect to their meanings.

There are only a few approaches that try to identify cognates across different concepts, and one could say that the task of *cross-semantic cognate detection* is still one of the open problems in computational historical linguistics. Approaches proposed so far include a rather complex workflow by Wahle (2016), who uses *hidden Markov models* for sequence comparison, and proxies on colexifications, drawn from the database by Dellert and Jäger (2017), to infer cognates across different meaning slots. As this task is not completely evaluated, and only described in a short paper, it is difficult to access its usefulness for our purposes. Another approach is presented by Arnaud et al. (2017), who apply Support Vector Machines trained on form and semantic similarities of word pairs along with a flat clustering algorithm to partition words into cognate sets. While this approach is publicly available and seems to yield promising results, we are not sure to which degree it would help us with our very specific goals of lifting an initially “raw” dataset to a level where we can assess sound correspondence patterns across multiple languages, especially since the algorithms the authors use for cognate detection do *not* take regular sound correspondences into account, and they are also *not* sensitive to partial cognates.

Thus, instead of these previously proposed solutions, we propose our own, rather simple approach to search for cross-semantic partial cognate sets in our data. This approach is based on the well-observed fact that the majority of morphemes in South-East Asian languages with a certain preference for compounding and a high degree of word formation, is highly *promiscuous* (List et al. 2016a: 8f), given that they recur within different words, surfacing in the form of *partial colexifications* (Hill and List 2017: 62). The term *partial colexification* hereby serves as a cover term for morphemes recurring across the lexicon of a language, with no specific distinction being made if they are polysemous or homophonous.

Our search for partial colexifications would not allow us directly to identify cross-semantic cognates consistently, given that sound change may yield different morpheme mergers across different languages. As a result, we cannot take the information from one language alone, but have to smartly summarize all the information on recurring morphemes we can find in our data. The solution for this problem is nevertheless straightforward, and it builds on the idea to not only compare single words, as originally proposed in Hill and List (ibid.), but to compare complete *alignments* instead. As our data is already aligned, and we have identified cognates in a first run, potentially even refined by experts, we can compare whole cognate sets that contain *identical words in the same language*.

If two alignments are completely identical with respect to the words they contain, there is no reason to assign them to different cognate sets, and we can directly assign them to the same cognate class. Even if they are simply homophonous, the assumption of regular sound change will allow us to treat them similarly if we reconstruct the words back to the ancestral language.

The problematic cases are those cases, where we have *incomplete data*. And this is usually rather the rule than the exception. We often will encounter cases where we have two alignments which are only filled in parts with data from the different languages, and we will usually have *missing data* for one or more of the languages in our sample in a given alignment. Thus, when comparing two alignments with each other, we need to make sure that we have at least one word in one language in common.

As an example, consider the data on "son" and "daughter" in five language varieties of our illustration data. As can be seen immediately, two languages show striking *partial colexifications* for the two concepts, Chuanqiandian and East Qiandong. In both cases, one morpheme recurs in the words for the two concepts. In the other cases, we find different words, but if we compare the overall cognacy, we can also see that all five languages share one cognate morpheme for "son" (corresponding to the Proto-Hmong-Mien *tu̯ɛn in Ratliff's reconstruction), and three varieties share one cognate morpheme for "daughter" (corresponding to *mphje^{D} in Ratliff's 2010 reconstruction), with the morpheme for "son" occurring also in the words for "daughter" in East Qiandong and Chuanqiandian, as mentioned before.

Language	Concept	Form	Cognacy	Cross-Semantic
East Baheng	SON	taŋ^{35}	1	1
East Baheng	DAUGHTER	p^{h}je^{53}	2	2
West Baheng	SON	ʔa^{3}/^{0} + taŋ^{35}	3 1	3 1
West Baheng	DAUGHTER	ta^{55} + qa^{3}/^{0} + t^{h}jei^{53}	4 5 6	4 5 6
Chuanqiandian	SON	to^{43}	1	1
Chuanqiandian	DAUGHTER	^{n}ts^{h}ai^{33}	7	7
Chuanqiandian (Central Guizhou)	SON	tə^{2}/^{0} + tə̃^{24}	8 1	8 1
Chuanqiandian (Central Guizhou)	DAUGHTER	tə̃^{24} + ^{n}p^{h}e^{42}	9 2	1 2
East Qiandong	SON	tei^{24}	1	1
East Qiandong	DAUGHTER	tei^{24} + p^{h}a^{35}	9 2	1 2

Table 2: Terms for "son" and "daughter" across five Hmong-Mien varieties.

Our workflow for automatically identifying these cases of cognacy is a new algorithm for cross-semantic cognate detection, developed first for the work in the Burmish Etymological Dictionary project lead by Nathan W. Hill. In this workflow, we start from all aligned cognate sets in our data, and then systematically compare all alignments with each other. Whenever two alignments are *compatible*, i.e., they have (1) at least one morpheme in one language occurring in both aligned cognate sets, which is (2) identical, and (3) no shared morphemes in two alignments which are *not* identical, we treat them as belonging to one and the same cognate set. We iterate over all alignments in the data algorithmically, merging the alignments into larger sets in a greedy fashion, and re-assign cognate sets in the data.

The results can be easily inspected with help of the EDICTOR tool, for example, by inspecting cognate set distributions in the data. When inspecting the cross-semantic cognates, which we label

`CROSSIDS` in our data, the tool will always show, which cognate sets span more than one concept, and users can directly filter the data and look at the relevant instances. Among the 64 cognate sets reflected in all languages in our sample, we find quite a few cross-semantically recurring morphemes, seven in total (with many more for the whole data). The results are shown in Table 3.

Language	Concept	Form	Morphemes
East Baheng	NOSE	^{n}pjau^{31}	`NOSE`
East Baheng	NASAL MUCUS	qa$^{3/0}$ + ^{n}pjau^{31}	`qa NOSE`
West Luobuohe	TWO	ʔu^{31}	`TWO`
West Luobuohe	TWENTY	ʔu^{31} + ʑo^{31}	`TWO zo`
West Baheng	SON	ʔa$^{3/0}$ + taŋ^{35}	`SON`
West Baheng	SON-IN-LAW	taŋ^{35} + wei^{31}	`SON wei`
West Baheng	GRANDSON	taŋ^{35} + seŋ^{31}	`SON seng`
East Qiandong	SUN	q^{h}aŋ^{33} + nei^{24}	`po SUN`
East Qiandong	DAY (NOT NIGHT)	nei^{24}	`SUN`
West Baheng	FAECES (EXCREMENT)	qa^{31}	`SHIT`
West Baheng	STOMACH	ʔa$^{3/0}$ + tɕ^{h}i^{35} + qa^{31}	`a tci SHIT`
West Qiandong	ANT	kæ̃^{44} + mjɔ^{22}	`INSECT mjo`
West Qiandong	EARTHWORM	kæ̃^{44}+tɕuŋ^{44}	`INSECT tsung`
East Baheng	BIRD	taŋ^{35} + nuŋ^{31}	`BIRD-A BIRD-B`
East Baheng	NEST	ʑo^{11} + taŋ^{35} + nuŋ^{31}	`zo BIRD-A BIRD-B`

Table 3: Partial cognates among stable concepts with reflexes in all languages in our test datasets. We highlight shared cognates by giving a tentative gloss for them in capital letters in the column *Morphemes*.

SUMMARY
• For a realistic analysis, we need to identify cognates not only within the same meaning slot, but across different concepts, specifically when dealing with languages in which compounding and word formation are very productive. • We employ a new method that makes use of a comparison of the alignments in readily identified and aligned partial cognate sets to identify those morphemes which recur across different concepts in our data. • The results can be inspected with help of the EDICTOR, but not directly, by now, only indirectly with help of the browser for cognate sets. • The interpretation of the results cannot be done automatically, but requires expert assessment with respect to the morphology of the data under consideration.

3.5 From cross-semantic cognates to correspondence patterns

4 Discussion

4.1 Possible improvements

a: semi-automatic reconstruction b: clearer integraation of automatic and semi-automatic methods in teh workflow c: better handling of output of the automatic ttasks (wvisualziation, etc.)

4.2 General challenges

a: lexical reconstruction: how to reconstruct whole words? b: sound change representation of all changes along some phylogeny with sound laws

5 Outlook

References

Anthony, D. W. and D. Ringe (2015). “The Indo-European homeland from linguistic and Archaeological perspectives”. *Annual Review of Linguistics* 1, 199–219.

Arnaud, A. S., D. Beck, and G. Kondrak (2017). “Identifying cognate sets across dictionaries of related languages”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. (Copenhagen, 09/07–09/11/2017). Association for Computational Linguistics, 2509–2518.

Barrachina, S. et al. (2008). “Statistical approaches to computer-assisted translation”. *Computational Linguistics* 35.1, 3–28.

Dellert, J. and G. Jäger (2017). *NorthEuraLex (Version 0.9)*. Tübingen: Eberhard-Karls University Tübingen.

Dybo, A. and G. S. Starostin (2008). “In defense of the comparative method, or the end of the Vovin controversy”. In: *Aspekty komparativistiki* [Aspects of comparative linguistics]. Vol. 3: *Aspekty komparativistiki*. Ed. by I. S. Smirnov. Moscow: RGGU, 119–258.

Forkel, R., J.-M. List, S. J. Greenhill, C. Rzymski, S. Bank, M. Cysouw, H. Hammarström, M. Haspelmath, G. A. Kaiping, and R. D. Gray (2018). “Cross-Linguistic Data Formats, advancing data sharing and re-use in comparative linguistics”. *Scientific Data* 5.180205, 1–10.

Hill, N. W. and J.-M. List (2017). “Challenges of annotation and analysis in computer-assisted language comparison: A case study on Burmish languages”. *Yearbook of the Poznań Linguistic Meeting* 3.1, 47–76.

Holm, H. J. (2007). “The new arboretum of Indo-European “trees⍰. Can new algorithms reveal the phylogeny and even prehistory of Indo-European?” *Journal of Quantitative Linguistics* 14.2-3, 167–214.

List, J.-M. (2014). *Sequence comparison in historical linguistics*. Düsseldorf: Düsseldorf University Press.

– (2016). “Beyond cognacy: Historical relations between words and their implication for phylogenetic reconstruction”. *Journal of Language Evolution* 1.2, 119–136.

– (2017). “A web-based interactive tool for creating, inspecting, editing, and publishing etymological datasets”. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics. System Demonstrations*. Valencia: Association for Computational Linguistics, 9–12.

List, J.-M. (2018). *Regular cognates: A new term for homology relations in linguistics*. Vol. 5. 8.

– (2019). “Automatic inference of sound correspondence patterns across multiple languages”. *Computational Linguistics* 1.45, 137–161.

List, J.-M., J. S. Pathmanathan, P. Lopez, and E. Bapteste (2016a). “Unity and disunity in evolutionary sciences: process-based analogies open common research avenues for biology and linguistics”. *Biology Direct* 11.39, 1–17.

List, J.-M., P. Lopez, and E. Bapteste (2016b). “Using sequence similarity networks to identify partial cognates in multilingual wordlists”. In: *Proceedings of the Association of Computational Linguistics 2016 (Volume 2: Short Papers)*. Association of Computational Linguistics. Berlin, 599–605.

List, J.-M., S. Greenhill, C. Anderson, T. Mayer, T. Tresoldi, and R. Forkel, eds. (2018a). *CLICS: Database of Cross-Linguistic Colexifications*. URL: `http://clics.clld.org/`.

List, J.-M., S. Greenhill, T. Tresoldi, and R. Forkel (2018b). *LingPy. A Python library for quantitative tasks in historical linguistics*. URL: `http://lingpy.org`.

Moran, S. and M. Cysouw (2018). *The Unicode Cookbook for Linguists: Managing writing systems using orthography profiles*. Berlin: Language Science Press.

Ratliff, M. (2010). *Hmong-Mien language history*. Canberra: Pacific Linguistics.

Wahle, J. (2016). “An approach to cross-concept cognacy identification”. In: *Proceedings of the Leiden Workshop on Capturing Phylogenetic Algorithms for Linguistics*. “Capturing Phylogenetic Algorithms for Linguistics” (Leiden, 10/26–10/30/2015). Ed. by C. Bentz, G. Jäger, and I. Yanovich. Tübingen.

陳其光, C. Q. (2012). *Miàoyáo yǔwén* 妙药语文 [Miao and Yao language]. Ed. by Anonymous. Běijīng: Zhōngyāng Mínzú Dàxué 中央民族大学[Central Institute of Minorities].