# Closing the NLP Gap: Documentary Linguistics and NLP Need a Shared Software Infrastructure

**Luke Gessler**
Department of Linguistics
Georgetown University
lg876@georgetown.edu

## Abstract

For decades, researchers in natural language processing and computational linguistics have been developing models and algorithms that aim to serve the needs of language documentation projects. However, these models have seen little use in language documentation despite their great potential for making documentary linguistic artefacts better and easier to produce. In this work, we argue that a major reason for this NLP gap is the lack of a strong foundation of application software which can on the one hand serve the complex needs of language documentation and on the other hand provide effortless integration with NLP models. We further present and describe a work-in-progress system we have developed to serve this need, Glam.

## 1 Introduction

The labor that is required for documenting a language is complicated, repetitive, and time-consuming. As many have pointed out, methods from NLP and computational linguistics have great potential for expediting the documentary process (Bender et al., 2004; Gessler, 2019; Neubig et al., 2019, *inter alia*), and researchers in NLP/CL[1] have made great progress in advancing the ability of their models and algorithms to serve the needs of documentary linguistics. At the same time, interest in "low-resource languages" has surged in the past few years in the NLP research community, and there ought to be no better time than now for documentary projects to benefit from the contributions of researchers in NLP.

But somehow, most documentary linguistic work done even today in 2022 proceeds without any of the assistance that methods in NLP could be providing. This has been the state of affairs for quite some time—the introductory paragraph of the preface for the proceedings of the first ComputEL conference (Good et al., 2014) explains (emphasis ours):

> Contemporary efforts to document the world's endangered languages [...] are dependent on the widespread availability of [...] software to annotate [documentary data]. However, despite well over a decade of dedicated funding efforts aimed at the documentation of endangered languages, the **technological landscape** that supports the work of those involved in this research **remains fragmented**, and the **promises of new technology remain largely unfulfilled**. Moreover, **the efforts of computer scientists**, on the whole, are mostly **disconnected from the day-to-day work of documentary linguists**, making it difficult for the knowledge of each group to inform the other. On the one hand, **this deprives documentary linguists of tools making use of the latest research results to speed up the time-consuming task of describing an underdocumented language**. On the other hand, it severely limits the ability of computational linguists to test their methods on the full range of world's linguistic diversity.

Eight years later, at ComputEL-5, these words for the most part read as though they could have been written today.

Why is it that these "promises of new technology" remained unfulfilled for documentary linguists[2]? We argue here that the fundamental issue preventing vigorous exchange between documentary linguistics and NLP is a lack of application software which can adequately serve both communities: while it is true that apps exist and are commonly used in documentary linguistics, they are ill-suited for integration with NLP models. We therefore claim that **documentary linguistics will not benefit from advances in NLP until significant investments are made in developing application software which can compete with existing**

[1] We will simply write "NLP" in this work as a catch-all for any kind of computational work involving language, as a distinction between NLP and computational linguistics is fraught and not particularly important for the issues we discuss.

[2] For want of a better phrase, we will use "documentary linguist" as a flawed but useful shorthand for anyone involved in the documentary process who is not a computationalist, with the understanding that a linguist is only one kind of person who can be involved in a language documentation project.



*Proceedings of the Fifth Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 119 - 126

**apps in functionality and provide first-class support for NLP model integration**.

In this work, we present and discuss this thesis, outlining ideals for what application software ought to accomplish for the documentary linguistics community. In addition, we present a work-in-progress system we have developed which attempts to implement these ideals as practical, usable application software aimed at catalyzing research relationships between documentary linguists and computationalists by taking the needs of both seriously and in equal proportion.

## 2 Related Work

### 2.1 NLP for Language Documentation

NLP researchers have grown steadily more more interested in work on what in the NLP community are referred to as "low resource languages", with the watershed moment perhaps being the advent of deep learning in NLP in the early 2010's (LeCun et al., 2015).[3] A full review of this work is out of scope of the present work, but suffice to say that leading NLP researchers believe enough progress has been made that the average language documentation project could benefit greatly from NLP assistance, though they also observe that adoption of methods in NLP in language documentation has been slow (Neubig et al., 2019, 2020).

### 2.2 Language Documentation Apps

Since the 90's, application software has entered use in language documentation, with many of them focusing particularly on speech transcription and linguistic annotation of transcribed speech (glossing, POS tagging, etc.).[4] Many apps have been created, but a few have emerged as favorites. ELAN (Wittenburg et al., 2006) is favored for transcribing speech from audio or video recordings, and SIL products, FLEx (Moe, 2008) and SayMore[5] foremost among them, are popular for analysis such as interlinearization and lexicon construction. Development of these apps all began well before methods in NLP were mature enough to be practically useful for the average low-resource language, and as a consequence, these apps were not designed to accommodate integration with NLP models and have struggled to expand to support them.

For example, Moeller and Hulden (2018) present an algorithm for automatic glossing of transcribed documentary data, but as they describe, it was impossible to integrate the model into FLEx itself—instead, data needed to be exported from FLEx so that it could be presented to the algorithm. This is a common limitation: in the area where there has been the most activity on providing usable NLP for documentary linguists, automatic speech recognition (ASR), the leading solution, ELPIS (Foley et al., 2018), requires that users close their ELAN file, present it to the model, then download a new ELAN file to replace the old one with the ASR output. Thus while it is sometimes possible today to use NLP models in conjunction with the leading software solutions for language documentation, support is limited to the NLP packages which explicitly support this option, and there are very few examples of language documentation apps providing in-app integration with NLP models.

The earliest example we are aware of of an app which attempts to provide rich in-app integration with computational tools work is Bender et al. (2004), where a vision for high-tech language documentation is given, accompanied by a prototype implementation. The system, Montage, describes a documentary workflow where the documentary workflow is tightly integrated with contemporary NLP techniques (specifically, "precision formal grammars"): for example, grammatical description is brought into the software, which allows users to construct a grammar in the app instead of "offline", and the implemented grammar becomes available for partial parsing of new textual inputs.[6] Critically, what is enabling the use of these advanced methods from computational linguistics in Montage is a foundation of application software: for example, the "markup tool" which enables the construction of the precision formal grammars would need to be a complicated piece of UI which can present

[3] What exactly counts as "low resource" is extremely variable, but its meaning is essentially that a language does not have nearly as much readily usable linguistic data as a "high resource" language such as Mandarin Chinese or Arabic, with respect to either quality or quantity. Thus even a language with many speakers, such as Luganda with 20M speakers, might count as a low resource language depending on context. Virtually all languages being documented by linguists would count as "low resource" from an NLP perspective.

[4] There are many other parts of the language documentation "pipeline" beyond these, such as metadata management, but since these are the tasks that have received disproportionate attention, we will mainly focus on them in this work.

[5] https://software.sil.org/saymore/

[6] Tangentially, it is also worth noting their discussion of software providing first-class support for the hypertextual links that inhere in documentary artefacts, e.g. between example sentences in a grammar and the texts the examples were drawn from, along the lines of Musgrave and Thieberger (2021).

itself and the content of precision formal grammars in a way that is approachable to documentary linguists.[7]

Beyond the apps that have been mentioned so far, some others have been developed through the years, though none of them have made it a major goal to tackle the issue of NLP integration. For example LingSync (Dunham, 2014; Dunham et al., 2015) is a newer app along the lines of FLEx; Hall (2022) presents a toolkit for empowering documentary linguists to tailor apps to their needs; and SayMore[8] and Aikuma (Bird et al., 2014) are apps aimed at spoken text collection and transcription. But none of these projects make it a major goal to tackle the NLP model integration problem.

In sum, while there is every indication that NLP models are ready to provide documentary linguists with great productivity gains, existing apps have not been able to accommodate them in a way that is ergonomic and complete, and no new apps have yet emerged which are competitive with the most popular apps on features and offer first-class support for integrating with NLP models. We term this disconnect between the availability of NLP models and the inability of existing apps to make effective use of them the NLP gap.

## 3 The NLP Gap

Why does the NLP gap exist? That is, why is it that language documentation is still being carried out without the help of NLP models despite their great potential to help? We argue here that the single most important reason why the NLP gap exists is a rather simple one: there is not a foundational infrastructure of application software that can serve both NLP researchers and documentary linguists.[9]

When one first thinks of language documentation, and NLP models in language documentation, one might suppose that it is the development of NLP models and their application that is hard. Indeed, developing these models is hard, and low-resource NLP is by no means solved. But we have reached a point where some models can be applied to any language and work with a respectable amount of accuracy even without any additional training, one such example being the universal phone recognizer of Li et al. (2020). Some logistical difficulties might remain (e.g. preparing and maintaining computers for them to run on, and finding stakeholders in the project who have the know-how to run them), but for many larger documentation projects these issues are not serious, and we still do not see them using these models.

If models are good enough to deliver value, and documentary linguists want to reap the benefits of NLP and know where in their workflows they'd like models to assist, and computationalists are often available to assist in getting their models to process documentary data, then what else remains? The only possible answer seems to be that it is the lack of support in language documentation apps that is to blame. As noted in §2, documentary linguists cite difficulty in using models, as to the extent that they are available at all, they are usable only in awkward ways which grate against their workflows. NLP models, if they are to be unobtrusive, must have deep integration with documentary workflows, and since these workflows occur in software, NLP models must be deeply integrated into documentary software, the only substrate in which vigorous exchange between these two communities may occur.

This is not a small challenge, as this software, if it were to succeed in its goal of catalyzing cooperation between computationalists and documentary linguists, would need to serve well the needs of both parties. From the perspective of the documentary linguists, the whole point of using an NLP model is that it ought to reduce their labor, and as we have seen, existing ways of using NLP models with apps like FLEx and ELAN are unergonomic to the point of often being more work than the alternative. From the perspective of NLP researchers, we must make it easy for them to do something more than make their model publicly available, which is a necessary but unfortunately insufficient step in making them usable by all but the most technically

[7] To our knowledge, Montage was never implemented, and nothing has been published on it since 2005, though some of its conceptual threads have been continued in the AGGREGATION project (http://depts.washington.edu/uwcl/aggregation/).

[8] https://software.sil.org/saymore/

[9] We must hasten to add that this is not the *only* reason for the NLP gap: there are broader problems to be solved, such as how to succeed in designing language technology in a way that includes and serves the many stakeholders in the documentary process (Bird, 2018), and how to do so in a way that will not reproduce the colonial legacy of disenfranchisement and extraction (Bird, 2020). But the lack of software is at least as important as these other issues—addressing the lack of software may not be sufficient for closing the NLP gap, but it is necessary. As such, we will focus here on the narrow, software problem, recognizing that there are broader problems that need to be solved in fully equip every party in a documentary process with language technologies.

experienced and motivated documentary linguists.

Beyond these design challenges, there is also the challenge of how to find the labor necessary to develop this software, which has been noted as a severe issue (Thieberger, 2016). Despite the fact that a path forward for excellent research and positive outcomes for language communities requires significant investment in application software infrastructure, the cultural currents within both linguistics and NLP, for better or worse, dictate that software engineering (which also happens to be incredibly time-intensive) does not constitute research activity. The obvious outcome is that no researcher in either discipline would be well advised to make this kind of work more than a side-interest in their research interests, and it is telling that the two most popular apps, FLEx and ELAN, were developed by software engineering staff at language-related organizations rather than academics themselves.

That is a bleak outlook—is a shift in how our fields reward software development too much to hope for? It is worth digressing for a moment here to note that academic communities do have the power to change how the field views and rewards software artefacts as contributions, if they choose to prioritize bringing about such a cultural shift. For example, in the field of astronomy, academics have been publishing software packages providing implementations of commonly needed statistical and simulation algorithms for decades, though traditionally, such packages were only viewed as "contributions" worthy of the attention of, say, a hiring or tenure committee if there was an associated publication in a journal (Chase, 2022). Securing such a publication could be difficult if a package was very specialized or small, and as the need for new packages has risen sharply, the field of astronomy has responded by lowering the requirements for a "software publication" (see Kelley 2021 for an example). In the future, the field may be moving towards treating a package in itself as a "publication" (in the academic sense, i.e. something that can appear on a C.V. or be indexed by Google Scholar). In sum, the field was able to recognize that its traditional assessment and treatment of certain research activity was no longer appropriate, and needed to be changed so that activity that used to be thought of as marginal would be recognized and rewarded as a first-class scholarly activity.

Despite these challenges, we believe it is possible and vitally important for researchers in language documentation and NLP to try to find ways of building the backbone of application software which is needed for interchange between the two fields to progress, which as we hope is clear by now is crucially necessary for achieving widespread use of NLP models in language documentation. In the short term, we hope that individuals will be able to overcome career risks that come with working on something that is not "research" by cooperating with others, thereby amortizing the loss of time spent on more traditional research topics. In the long term, we challenge senior academics, and especially senior academics in NLP who have presented their models and algorithms as beneficial for language documentation, to consider whether it is not time to reassess whether the software work we have described is deserving of more recognition and support, and if it is, how the community's cultural values and institutions could be changed to reward such work.

We close our discussion of the NLP gap on this note. In the remainder of this work, we turn to describe what we believe would be key goals for an app aimed at closing the NLP gap, and further describe a prototype-grade system we have constructed which aims to achieve these goals.

## 4 System Description

Glam is an alpha-quality system we have developed which aims to serve the needs we have described. While for the rest of this section we speak mostly of design instead of the state of the implementation, we take a moment to note its progress.

In its present state, Glam is capable of surface-level interlinear annotation of texts, and there is work underway to add support for lexical inventories (as in FLEx). This is the bare minimum necessary to conduct a small-scale language documentation project, such as for a semester-long field methods course that might be offered at a university. Support for NLP models has not yet been implemented, which may seem strange. The reason is that, as we have noted, it is important for this app to fully satisfy the needs of both documentary linguists and NLP researchers, and we have viewed the former as the much harder problem and prioritized solving it first. We have however naturally been considering the problem of NLP integration from the very initial stages of design, and have made implementation decisions with care in order to facilitate its eventual implementation. The latest

state of the project can be tracked by visiting the repository.[10]

### 4.1 Core Goals

After considering the many and often conflicting needs that arise in language documentation and using models in language documentation, we arrived on these five goals, which we believe are some of the most important to achieve in order to make an app that documentary linguists will gladly use and will be easily integrable with models.

1. **Flexible Data**: all language documentation projects have different data needs, so you should be able to record however much data and whatever kind of data you desire. Annotating anything from good old-fashioned interlinear glossed text to more complicated formats like Universal Dependencies should be possible and easy.

2. **Seamless Collaboration**: working with others should be frictionless—you should be able to share data without even clicking a button, changes should be viewable by everyone in real time, and everyone should be able to pick the system up quickly.

3. **Durability**: data should never be lost—all past states of the database should be recorded and accessible.

4. **NLP Model Integration**: it should be easy to configure cutting-edge NLP models to provide best possible annotations to be corrected by humans, and have them train incrementally as new gold annotations become available.

5. **Pluggable UIs**: if you want to code new UIs for different kinds of annotation (e.g. entity recognition, syntax, and coreference), you should be able to do so just by writing JavaScript using the Glam API, with no backend changes required.

### 4.2 Implementation

We will review some key points of our implementation of Glam here. It would take space beyond what is available here to describe exactly how documentary workflows are performed in Glam—instead, we will discuss only the fundamentals here, and refer readers to a video demo for more detail.[11]

**Platform** Glam is implemented, in software engineering jargon, as a single-page web application. We chose to make Glam a web application because of the difficulty that comes with requiring local installation of apps: for example, some apps are not compatible with certain operating systems (FLEx, for instance, does not work on macOS), and others require some tricky installation steps (ELAN can require you to download supplementary software during installation). These difficulties are bypassed in a web application, where all that is required is a web browser and an internet connection (albeit at the cost of maintaining a publicly-accessible web server).

**Database** Data in Glam is stored in XTDB,[12] an immutable database which allows all past states of the database to be accessible. This means that data cannot be lost, and moreover that if there were demand for it, it would be relatively straightforward to allow users to see historical states of the database.[13]

**Data Model** The data model of Glam is designed to be extremely flexible: documents in the system are separated by project, and each project has a structure which is expressed just in terms of four basic constructs, which we call *layers*. A text layer holds a string representing the text that is to be analyzed. A token layer depends on a text layer and holds *tokens*, each of which is defined using the text layer with a begin and end index. A span layer depends on a token layer and holds *spans*, each of which refers to at least one token and has a value, such as a POS tag or an entity label. A relation layer depends on a span layer and consists of *relations*, each of which has a start and end span and has a value, such as a dependency relation or a coreference type. A vocabulary layer is a list of items which have at minimum a *form* and any number of additional fields, which may hold information such as part of speech or alternative spellings and may be open or closed depending on whether it is desirable for users to expand the vocabulary with more entries.

These layers are designed to be sufficient to express any kind of linguistic annotation, and we

[10] https://github.com/lgessler/glam

[11] https://youtu.be/VXWPw91nTGY

[12] https://xtdb.com/

[13] Sometimes it might be desirable to destroy data, e.g. if a language consultant decides a text is too sensitive to share. XTDB provides technical means for accomplishing this (the *evict* operation), and implementation of data eviction using this database facility is planned for Glam.

believe this is possible because other researchers in corpus linguistics (Zipser and Romary, 2010) and NLP (Jiang et al., 2020) have convincingly argued that very similar data models are capable of expressing almost any linguistic structure. In practice, we expect that most projects will have a very similar structure, but the intention behind approaching data modeling this way is to give users good support no matter what their data looks like. In addition, we plan to expand the data model with document-level metadata, which will be useful for tracking information such as when a text was collected and who produced the data.

**User System** A basic user system with password authentication is used for maintaining security over data. Privileged users called administrators can set up projects and manage users, and may grant users either read-only or read-and-write privileges over any project. By default, projects are invisible to users.

**NLP Integration** Recall that the data model of Glam is composed of five fundamental layers. NLP integration is made general for any layer with the following procedure:

1. An NLP model is prepared for integration by making it contactable via generic protocols, such as HTTP(S), e.g. by wrapping it in a small web server (such as Flask for Python) and implementing an API specification provided by Glam which describes what methods must be supported to e.g. tokenize a string of text.

2. The model is registered within the Glam instance by an administrator, which will tell the instance how to contact the model (e.g. by URL, like `http://127.0.0.1:5128`). At this point, the system will attempt to contact the model and, if successful, register the *hooks* that are supported by that model. A *hook* is an action the model can take whenever a certain operation happens: for a span layer, this might be token creation, token boundary modification, or token deletion.

3. Every layer that depends on output from that model will be configured to contact that model using the model registration, and the exact hooks which are to be executed may be modified.

This strategy produces a loose coupling between NLP models and Glam: their only point of contact is HTTP(S) with a specified structure, meaning that as long as the model provides this it can be implemented in any way desired.

### 4.3 Outlook

At present, Glam has been receiving feedback from documentary linguists and is a few months from a beta release. Multiple field linguists have expressed interest in some of the design goals and features in Glam. Time will tell if the design and implementation choices we have made are the right ones, but our more important intent in this discussion is to demonstrate the kind of problems we think an app will need to solve in order to close the NLP gap.

## 5 Conclusion

We have discussed the problem of why NLP models have not seen more use in documentary linguistics, and concluded that the single most important barrier to adoption of NLP models is the lack of a substrate of application software that can serve the needs of both documentary linguists and NLP models well. We have moreover presented design goals and implementations of a system which we think shows potential to meet this need.

Regardless of the ultimate fortunes of our system, Glam, we reprise our invitation to readers to consider whether our assessment of the NLP gap is correct (i.e., that it cannot be closed without serious investment in application software, which in turn might require a cultural shift in some academic communities), and if it is, what there is to be done about it. NLP researchers have gained much from endangered languages, not least by sourcing unique data from them for publications—if they are in dire need of assistance that the NLP community is singularly able to provide, and which is not forthcoming from any other community or organization in the world, should the NLP community not act? Moreover, beyond this matter of deserts, there is also the exciting prospect of opportunity for new methods and models that could come from a deeper relationship between these two fields, mediated by a substrate of application software.

For junior researchers without a faculty position or tenure, a helpful action might be to find collaborators to work on this software problem with. For researchers in NLP working on low-resource NLP models aimed at application in documentation of

endangered languages, it might be right to consider whether they ought to have more involvement in making this application actual instead of potential. For senior researchers with tenure, who wield the most influence, it may be appropriate to reexamine the reasons why the current norms around what constitutes "research activity" are what they are, and whether it might be right to reform them given the unmet needs of endangered languages.

## Acknowledgements

We thank Andrew Harvey, Joey Lovestrand, and Patrick Hall for helpful comments on a draft of this work. We thank Patrick Hall and for feedback on both this work and a prelimary version of Glam. We thank Nathan Schneider, Amir Zeldes, Hannah Sande, William Croft, Andrew Cowell, Jin Zhao, Rosa Vallejos, Meagan Vigus, Jens Van Gysel, Lukas Denk, Joshua Birchall, and Alexis Palmer for critical feedback on preliminary versions of Glam. We thank Eve Chase for a helpful discussion on the nature of software contributions in linguistics/NLP and astrophysics.

## References

Emily Bender, Dan Flickinger, Jeff Good, and Ivan Sag. 2004. Montage: Leveraging advances in grammar engineering, linguistic ontologies, and mark-up for the documentation of underdescribed languages. *Proc. of LREC*.

Steven Bird. 2018. *Designing mobile applications for endangered languages*, 1 edition, pages 842–861. Oxford University Press, United Kingdom.

Steven Bird. 2020. Decolonising speech and language technology. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3504–3519, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Steven Bird, Florian R. Hanke, Oliver Adams, and Haejoong Lee. 2014. Aikuma: A mobile app for collaborative language documentation. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 1–5, Baltimore, Maryland, USA. Association for Computational Linguistics.

Eve Adde Chase. 2022. personal communication.

Joel Dunham, Jessica Coon, and Alan Bale. 2015. Lingsync: web-based software for language documentation. In *4th International Conference on Language Documentation and Conservation (ICLDC)*, Honolulu, Hawaii.

Joel Robert William Dunham. 2014. *The online linguistic database : software for linguistic fieldwork*. Ph.D. thesis, University of British Columbia.

Ben Foley, Josh Arnold, Rolando Coto-Solano, Gautier Durantin, T. Mark Ellison, Daan van Esch, Scott Heath, František Kratochvíl, Zara Maxwell-Smith, David Nash, Ola Olsson, Mark Richards, Nay San, Hywel Stoakes, Nick Thieberger, and Janet Wiles. 2018. Building Speech Recognition Systems for Language Documentation: The CoEDL Endangered Language Pipeline and Inference System (ELPIS). In *6th Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU 2018)*, pages 205–209. ISCA.

Luke Gessler. 2019. Developing without developers: choosing labor-saving tools for language documentation apps. In *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers)*, pages 6–13, Honolulu. Association for Computational Linguistics.

Jeff Good, Julia Hirschberg, and Owen Rambow, editors. 2014. *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*. Association for Computational Linguistics, Baltimore, Maryland, USA.

Patrick James Hall. 2022. *Participatory Design in Digital Language Documentation: A Web Platform Approach*. Ph.D. thesis, University of California, Santa Barbara.

Zhengbao Jiang, Wei Xu, Jun Araki, and Graham Neubig. 2020. Generalizing Natural Language Analysis through Span-relation Representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2120–2133, Online. Association for Computational Linguistics.

Luke Zoltan Kelley. 2021. kalepy: a Python package for kernel density estimation, sampling and plotting. *Journal of Open Source Software*, 6(57):2784.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

Xinjian Li, Siddharth Dalmia, Juncheng Li, Patrick Littell, Matthew Lee, Jiali Yao, Antonios Anastasopoulos, David Mortensen, Graham Neubig, Alan Black, and Florian Metze. 2020. Universal Phone Recognition with a Multilingual Allophone System. In *2020 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Barcelona.

Ronald Moe. 2008. FieldWorks Language Explorer 1.0.

Sarah Moeller and Mans Hulden. 2018. Automatic Glossing in a Low-Resource Setting for Language Documentation. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 84–93, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Simon Musgrave and Nick Thieberger. 2021. The language documentation quartet. In *Proceedings of the 4th Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers)*, pages 6–12, Online. Association for Computational Linguistics.

Graham Neubig, Patrick Littell, Chian-Yu Chen, Jean Lee, Zirui Li, Yu-Hsiang Lin, and Yuyan Zhang. 2019. Towards a General-Purpose Linguistic Annotation Backend. In *Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers)*, Honolulu, Hawaii. Association for Computational Linguistics. ArXiv: 1812.05272.

Graham Neubig, Shruti Rijhwani, Alexis Palmer, Jordan MacKenzie, Hilaria Cruz, Xinjian Li, Matthew Lee, Aditi Chaudhary, Luke Gessler, Steven Abney, Shirley Anugrah Hayati, Antonios Anastasopoulos, Olga Zamaraeva, Emily Prud'hommeaux, Jennette Child, Sara Child, Rebecca Knowles, Sarah Moeller, Jeffrey Micher, Yiyuan Li, Sydney Zink, Mengzhou Xia, Roshan S Sharma, and Patrick Littell. 2020. A Summary of the First Workshop on Language Technology for Language Documentation and Revitalization. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 342–351, Marseille, France. European Language Resources association.

Nick Thieberger. 2016. Language Documentation Tools and Methods Summit Report.

Peter Wittenburg, Hennie Brugman, Albert Russel, Alex Klassmann, and Han Sloetjes. 2006. ELAN: a Professional Framework for Multimodality Research. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Florian Zipser and Laurent Romary. 2010. A model oriented approach to the mapping of annotation formats using standards.