BELT: Building Endangered Language Technology

Michael Ginn[1] **David R. Saavedra-Beltrán**[2]
Camilo Robayo[2] **Alexis Palmer**[1]
[1]University of Colorado [2]Universidad Nacional de Colombia
michael.ginn@colorado.edu

Abstract

The development of language technology (LT) for an endangered language is often identified as a goal in language revitalization efforts, but developing such technologies is typically subject to additional methodological challenges as well as social and ethical concerns. In particular, LT development has too often taken on colonialist qualities, extracting language data, relying on outside experts, and denying the speakers of a language sovereignty over the technologies produced.

We seek to avoid such an approach through the development of the *Building Endangered Language Technology* (BELT) website, an educational resource designed for speakers and community members with limited technological experience to develop LTs for their own language. Specifically, BELT provides interactive lessons on basic Python programming, coupled with projects to develop specific language technologies, such as spellcheckers or word games. In this paper, we describe BELT's design, the motivation underlying many key decisions, and preliminary responses from learners.

1 Introduction

The development of language technologies (such as spellcheckers, automated transcription, and machine translation) has been commonly suggested as a goal in language revitalization projects (Kornai, 2013; Zhang et al., 2022). However, research and development of these LTs historically has been fraught with social and ethical concerns. NLP research generally underrepresents such languages (Joshi et al., 2020; Blasi et al., 2022), and research into so-called "low-resource" and endangered languages often treats them as a homogenous group of languages, identical to high-resource and politically dominant languages in every aspect except data availability (Doğruöz and Sitaram, 2022).

Worse, the development of LTs for endangered languages has often taken on colonialist qualities (Bird, 2020): over-promising the benefits of documentation and technology development (Speas, 2009; Brinklow et al., 2019); consuming language data with little regard to speaker privacy (Macri and Sarmento, 2010); denying the language community sovereignty over their data (Pool, 2016); prioritizing the development of tools of interest to the outside expert, rather than those desired by actual speakers (Liu et al., 2022); and relying on experts with little relationship to the community for continued development and maintenance (Bird, 2020; Flavelle and Lachler, 2023).

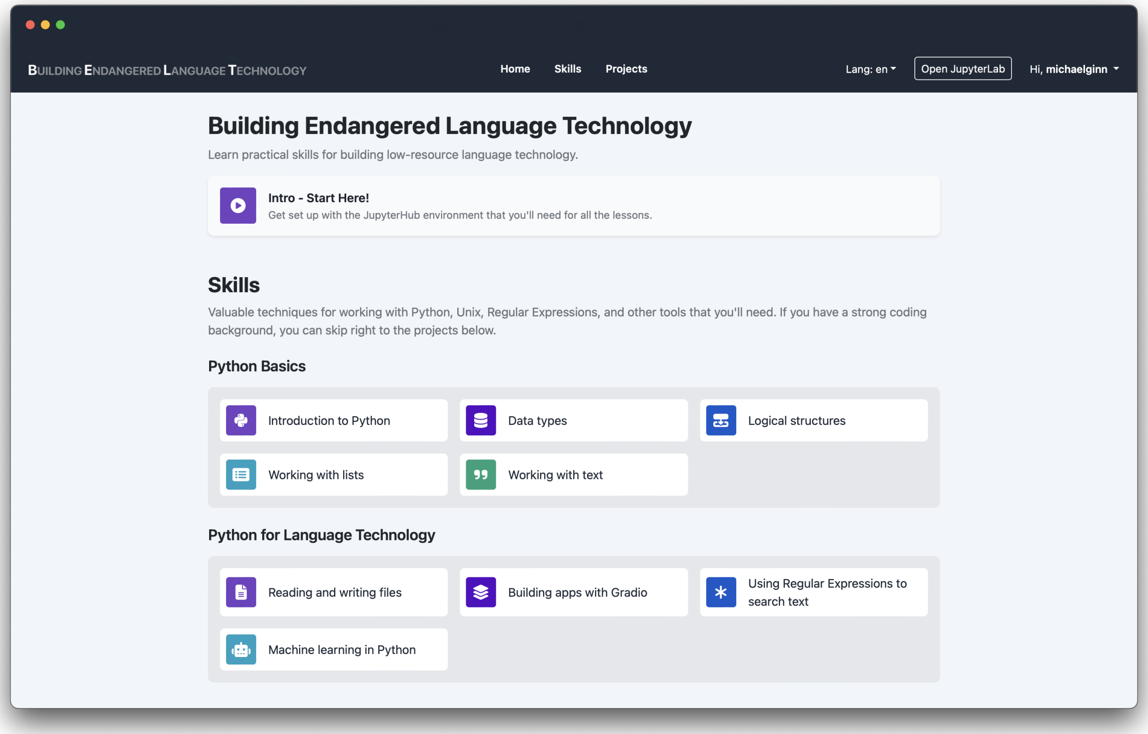
While some recent results suggest that large language models can be usefully deployed in language documentation and revitalization contexts (Tanzer et al., 2024; Zhang et al., 2024b,a, among others), most such models remain closed, and running them for a new language requires exposing data that language communities may prefer to keep private.

We strive to provide a resource for speakers interested in developing and maintaining language technology for their own language with the BELT (*Building Endangered Language Technology*) website. BELT is an educational tool designed for learners without any coding experience to gain basic programming skills, develop language technology using data from their language, and deploy simple applications for real-world usage.

In particular, we developed BELT with the following goals:

- A free, open-source resource that can be used for guided workshops as well as independent, asynchronous learning.
- Interactive Python lessons that encourage repeated practice and experimentation.
- Lesson materials that are approachable to a beginning programmer, while enabling creation of realistic and useful applications.

Proceedings of the Sixth Workshop on Teaching NLP, pages 94–104
August 15, 2024 ©2024 Association for Computational Linguistics



Building Endangered Language Technology

Learn practical skills for building low-resource language technology.

Intro - Start Here!
Get set up with the JupyterHub environment that you'll need for all the lessons.

Skills

Valuable techniques for working with Python, Unix, Regular Expressions, and other tools that you'll need. If you have a strong coding background, you can skip right to the projects below.

Python Basics

Introduction to Python
Data types
Logical structures
Working with lists
Working with text

Python for Language Technology

Reading and writing files
Building apps with Gradio
Using Regular Expressions to search text
Machine learning in Python

Figure 1: The main BELT course page. Lessons are divided into short skills and longer, in-depth projects. Each lesson links to a Jupyter Notebook in the user's environment.

- Language-agnostic project tutorials that require only a text file of unlabeled language data as input.
- A strict focus on NLP-related topics, avoiding an overly-general programming curriculum.
- Full localization into languages other than English.

We describe our design decisions for the website and lesson materials. Furthermore, we report feedback and results from two in-person workshops based around the BELT site, in which community members for a variety of endangered languages were able to build and share language technologies for their respective languages. Finally, we reflect on future improvements to continue to make BELT a valuable resource for NLP education.

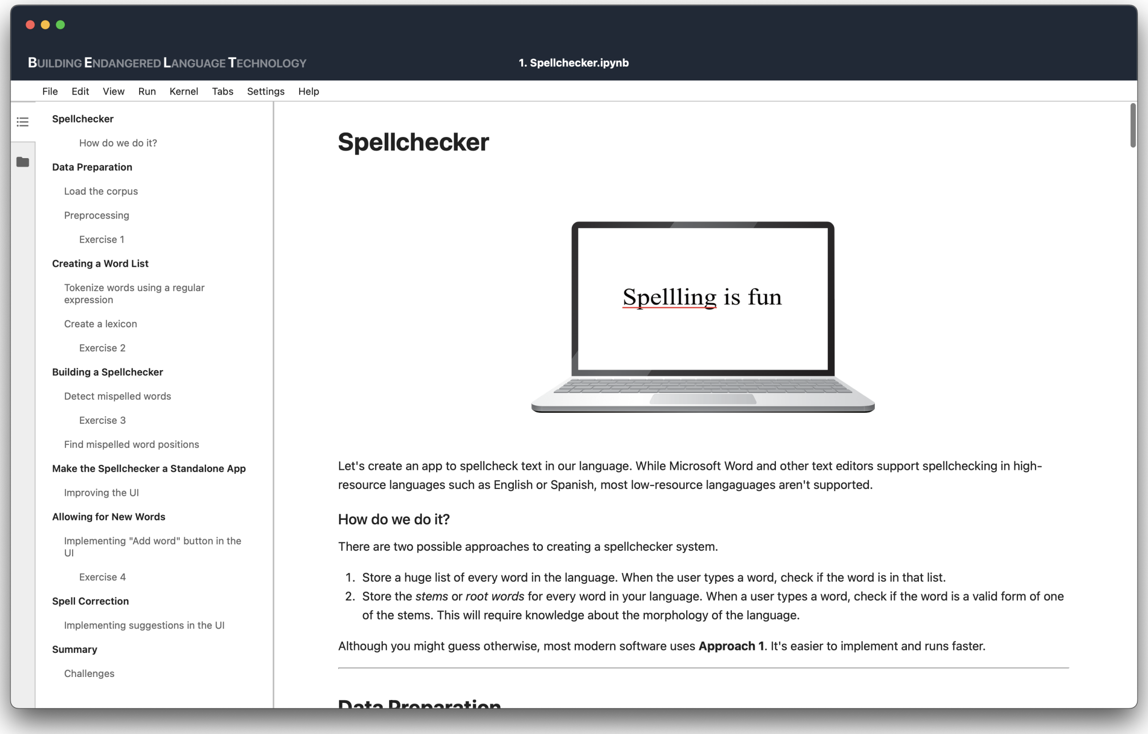
2 Overview

BELT is an interactive web course, which anyone can access for free.[1] After logging in, a user is presented with the main course page (Figure 1). The course contains fifteen interactive lessons covering language technologies and the coding skills needed to build them, which are provided here in a recommended order.

Curriculum Lessons are divided into *Skills*, shorter notebooks each covering a single focused topic (such as regular expressions), and *Projects*, longer tutorials which cover a realistic language technology (such as a spellchecker). We recommend completely new learners to work through the skills section first before attempting the projects; more experienced users may simply refer back to the skills section as needed.

Lesson Format Clicking on a lesson launches a user-specific Jupyter Lab environment, automatically loading a Jupyter notebook for that lesson (Figure 2). Notebooks are organized for structured learning and contain a mix of instructional text, prewritten code blocks, and interactive exercises and challenges. For example, the skill lesson covering strings in Python includes the exercise in Figure 3. Notebooks are stored on a per-user basis, and they persist across sessions. Lessons can access shared files from the server, such as corpora that we provide. In addition, users can upload their own files, allowing them to use their own language data for many of the projects.

[1] `https://lecs-lab.github.io/belt`



Spellchecker

Let's create an app to spellcheck text in our language. While Microsoft Word and other text editors support spellchecking in high-resource languages such as English or Spanish, most low-resource langaguages aren't supported.

How do we do it?

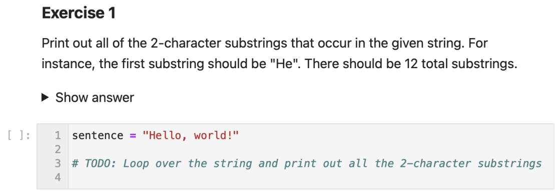
There are two possible approaches to creating a spellchecker system.

1. Store a huge list of every word in the language. When the user types a word, check if the word is in that list.
2. Store the *stems* or *root words* for every word in your language. When a user types a word, check if the word is a valid form of one of the stems. This will require knowledge about the morphology of the language.

Although you might guess otherwise, most modern software uses **Approach 1**. It's easier to implement and runs faster.

Data Preparation

Figure 2: Lessons load notebooks in the Jupyter Lab environment. Lessons are fully interactive, can load files from the shared filesystem as well as from the user's storage, and can even launch user-created apps with the Gradio framework.



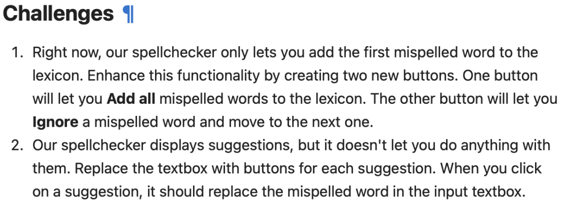
Exercise 1

Print out all of the 2-character substrings that occur in the given string. For instance, the first substring should be "He". There should be 12 total substrings.

```
sentence = "Hello, world!"

# TODO: Loop over the string and print out all the 2-character substrings
```

Figure 3: A sample exercise from the lesson on strings.



Challenges ¶

1. Right now, our spellchecker only lets you add the first mispelled word to the lexicon. Enhance this functionality by creating two new buttons. One button will let you **Add all** mispelled words to the lexicon. The other button will let you **Ignore** a mispelled word and move to the next one.
2. Our spellchecker displays suggestions, but it doesn't let you do anything with them. Replace the textbox with buttons for each suggestion. When you click on a suggestion, it should replace the mispelled word in the input textbox.

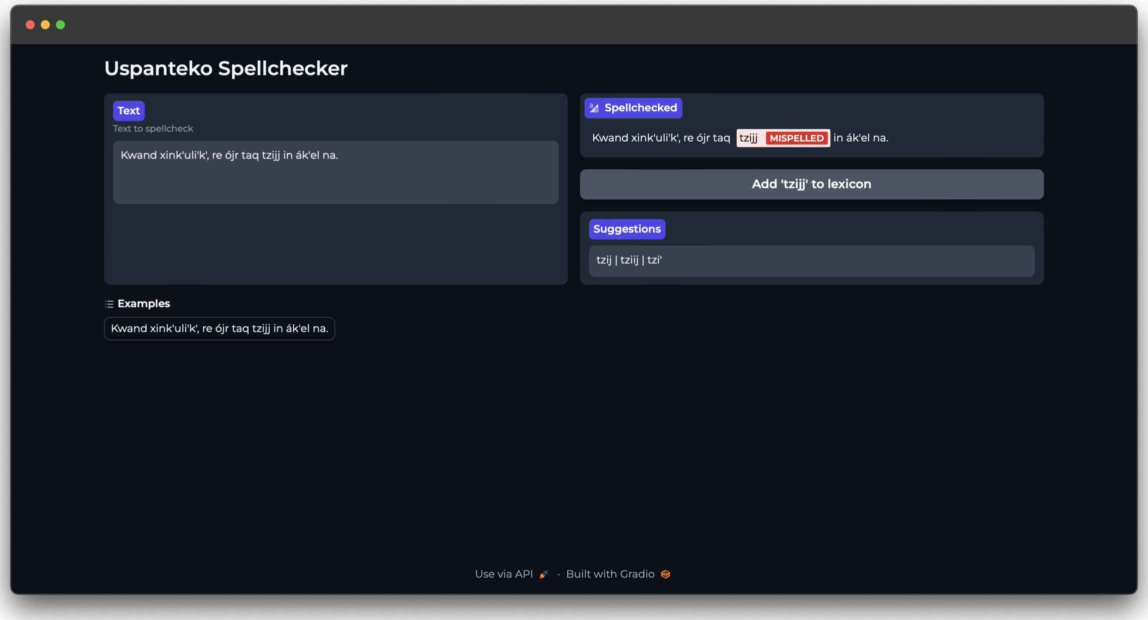
Figure 4: Challenge exercises at the end of the spellchecker lesson.

Projects While skill lessons tend to be short, focused overviews of relevant topics, projects are more involved and tailored to endangered language technology development. For example, the spellchecker lesson (Figure 2) requires only a wordlist as input. The spellchecker itself uses low-resource techniques such as edit distance to identify suggested spellings, rather than data-hungry, state-of-the-art methods (generally neural networks).

Projects are highly interactive, with many exercises drawing on knowledge learned in the skill lessons. Projects also include several challenge exercises (Figure 4) for motivated learners, ranging in difficulty from simple UI modifications to development of additional system features.

Gradio Perhaps the most overlooked challenge in community-led language application development is the deployment of usable, scaleable software. In BELT lessons, the Gradio[2] framework is used extensively in order to build simple user interfaces for the technologies developed. Gradio runs as a local, interactive app inside the Jupyter notebook, and it can be accessed temporarily via a live URL (see Figure 5). Furthermore, through Hugging Face Spaces,[3] Gradio apps can be hosted permanently and for free.

[2] https://www.gradio.app
[3] https://huggingface.co/spaces



Kwand xink'uli'k', re ójr taq tzijj in ák'el na.

Kwand xink'uli'k', re ójr taq tzijj MISPELLED in ák'el na.

tzij | tziij | tzi'

Kwand xink'uli'k', re ójr taq tzijj in ák'el na.

Figure 5: A running Gradio app. Learners create small applications such as spellcheckers, predictive text, and word games using the Gradio framework. Apps can be accessed via a live URL while they are running, or permanently if the app is hosted on Hugging Face.

3 Platform

We selected JupyterHub[4] as the primary framework powering BELT. JupyterHub is an open-source platform that allows many users to access Jupyter computing environments on a remote server. JupyterHub works by spawning a separate Jupyter environment for each user and serving the Jupyter Notebook or Lab application as a web app.

3.1 Alternatives

Before deciding on JupyterHub, we considered several alternatives, which we describe below in addition to our reasons for selecting as we did.

Static materials Digital textbooks or static webpages would require the least effort to distribute, and many high-quality NLP resources of this sort already exist. They require minimal computational resources, can often be used offline, and are straightforward to translate into other languages. However, we believe that interactivity is critical to a fluid learning experience, and a static approach would add significant friction, as learners would need to configure their own development environments and solve any issues that arise with installation.

Jupyter Notebooks Interactive notebooks have become a popular tool for pedagogy in computational fields (Cardoso et al., 2019; Johnson, 2020), allowing for alternating plain-text and code blocks that can be edited and run. Distributing downloadable Jupyter notebooks can often be an ideal choice for learners with some prior programming experience (such as computer science undergraduates), who are expected to set up Python environments, configure coding software, and install packages. While these are valuable skills to learn, we believed that this additional friction could discourage potential learners with minimal technical experience.

Jupyter Books Jupyter Book[5] is an alternative platform, also based on the Jupyter environment, used to create digital textbooks that incorporate interactive code contents. While Jupyter Book provides excellent features for creating well-formatted content, there are significant limitations for interactive content that would not allow learners to easily deploy their applications.

3.2 Deployment

For initial development of BELT, we use The Littlest JupyterHub (TLJH)[6] distribution, which runs

[4] https://jupyter.org/hub

[5] https://jupyterbook.org/en/stable/intro.html

[6] https://github.com/jupyterhub/the-littlest-jupyterhub

on a single server and supports up to one hundred simultaneous users. Jupyter Hub can also be run on Kubernetes, scaling to tens of thousands of users, and in the future we plan to deploy BELT using this method.

We hosted the initial BELT site on a server instance running Ubuntu 18.04 with 4 GB of RAM and 100 GB of disk space.

3.3 Serving Notebooks

When a user clicks a lesson for the first time, we use `nbgitpuller`[7] to fetch the lesson notebook from our Git repository. The latest version of the notebook is cloned into the user's personal storage on our JupyterHub. After this point, they can modify the notebook and changes will persist.

4 Design

We made several design decisions involving features and enhancements to the JupyterHub platform, aiming to make the BELT site as approachable and usable as possible.

4.1 Web Pages

By default, JupyterHub serves several web pages for login, registration, launching servers, and other basic functions. These pages have minimal, simple design elements, but we find that they can be confusing for users not familiar with that style of interface. To address this interface issue, we replace the pages completely, striving for a look that resembles a modern web or mobile app rather than a scientific tool (see Figure 1).

We add up-to-date styling libraries, Bootstrap v5[8] and FontAwesome v6,[9] both commonly-used frameworks in web design. We also create entirely new page templates (built with the Jinja[10] engine), custom CSS styles, and custom assets.

We also modify the configuration and style code for the Jupyter Lab environment, removing unnecessary elements to create a clean, intuitive interface (Figure 2).

4.2 Localization

Reaching a global audience is an important goal of the project. We prepared the application to be fully localized, and completed preliminary localization into Spanish and partial localization into Portuguese.

JupyterHub itself does not provide any framework for page internationalization, and standard localization libraries require server-side modifications. Instead, we developed a small JavaScript helper that does the following:

1. Finds any components marked with a custom HTML attribute `data-i18n-id`;
2. Dynamically replaces the string contents of the node with a localized string from the appropriate lookup table; and
3. Dynamically modifies any URLs to point to the appropriate localized content.

While this method may not scale well to large numbers of components, it is very lightweight and very performant in our usecase. We perform localizations using Crowdin,[11] a web platform for creating and storing localization data. The user can switch their language with a simple picker in the menu bar, and their choice is persisted with cookies.

Because Jupyter notebooks cannot be dynamically updated as easily, we employ a different strategy for localizing the notebooks. We create localized versions of every notebook using Crowdin, and store each version at a path containing the appropriate language code. Then, we dynamically switch the links using our localization script to point to the correct notebook.

Within lesson notebooks we primarily translate the informational material and part of the code comments (those corresponding to Portuguese), leaving variable and function names untouched for both languages. We debated whether to localize these as well, but elected against it as doing so may make running bilingual workshops more difficult. We also install language packs for the Jupyter Lab user interface, allowing the user to switch the language for text elements such as menus and tooltips.

5 Lesson Materials

Skills Currently, our curriculum includes lessons for the following skills:

1. Introduction to Python
2. Data types

[7] https://nbgitpuller.readthedocs.io/en/latest/index.html

[8] https://getbootstrap.com/docs/5.0/getting-started/introduction/

[9] https://fontawesome.com

[10] https://jinja.palletsprojects.com/en/3.1.x/

[11] https://crowdin.com

3. Logical structures
4. Lists
5. Strings and text
6. Files
7. Gradio
8. Regular expressions
9. Machine learning

Within each lesson, we strive to discuss only the information that is absolutely required for the projects later to come. Lessons tend to be short and focused, typically around fifteen to thirty minutes worth of content with a few exercises. We recognize that there are a multitude of existing, in-depth Python educational resources, and we refer learners to outside sources when appropriate.

The material is presented with English examples and written with a succinct prose style that strives to be accurate without being overly technical. We primarily frame each skill in relation to language technology; e.g., while regular expressions can be useful for a wide variety of tasks, we primarily discuss their use for searching and preprocessing natural language text.

Exercises strive to reinforce critical concepts. They often foreshadow tasks that will be necessary for projects; for example, the lesson on lists and sets asks the user to turn a list of words into a unique set. This skill is later used in the spellchecker project for creating a wordlist from a lexicon.

Projects Our curriculum currently includes four projects, described below. As development of BELT continues, we plan to add new projects, either for new language technologies, or for more advanced versions of the technologies already included. We additionally welcome new projects from the broader community.[12]

1. **Spellcheckers:** In this project, students create a simple, lookup-based spellchecker, using a word list built from a text file.
2. **Predictive text:** The same text file is used to build a simple n-gram language model, which is then used to predict the most likely next word in a sequence.
3. **Word games:** In this project, users learn to create two word games: a word scramble, and a hangman-style game. These can draw from the provided text, or users can upload custom word lists.
4. **Automatic morphological inflection:** In this project, users learn to use finite-state machines to build rule-based morphological inflection tools, which can be useful in spell-checking, dictionaries, language learning games, and other downstream applications.

These projects cover a number of common technologies that can be built with minimal resources and are appropriate for most languages. The projects use data files from several different languages to teach users how to build the technologies: English, Spanish, Latin, and the Mayan language Uspanteko. However, we strongly encourage learners to use their own language data where available, and we provide guidance for how to import, load, and manage those data files.

Within each project, exercises require application of information from the skill lessons (encouraging learners to use prior lessons as a reference), and they often combine multiple topics. To avoid too much frustration, learners can double-check their solutions by revealing answers to the exercises. For more difficult exercises, we offer for learners to receive one or more hints. We also provide stretch exercises, in the form of open-ended challenge problems, for more experienced learners.

Each project tutorial culminates with users having built a simple version of a tool. Depending on the data file used as input, these tools may or may not be suitable for real-world use. They are, however, suitable foundations for more sophisticated versions of the same tools. It is our hope that some learners will be inspired to build better and better versions of the tools produced by our lessons, thus working toward truly viable technologies for the languages those learners care about.

We would also note that our goal very specifically is not to teach learners how to build state-of-the-art tools using the latest developments in NLP. Rather, we aim for tools that have a low computational expense, need minimal input data, do not require annotation, and can be deployed in a straightforward way.

[12] Please contact the first author if you are interested in contributing to BELT.

6 Live Workshops

Over the last year, we have held two live workshops to teach all or part of the BELT lessons to different audiences. As it happens, both workshops took place in South America. The initial workshop, held in the summer of 2023, helped us to identify some improvements around both the hosting method and the lesson contents that we implemented for the second workshop at the start of 2024.

6.1 Bogotá, Colombia, June 2023

The first live workshop was held in the context of the Amazonicas IX conference in Bogotá, Colombia in June 2023.[13]

The event was attended by around fifteen participants, with hybrid participation (some in situ and others via Zoom). The participants were mostly linguistics/anthropology undergraduates with no or some basic programming experience.

The workshop was taught across four consecutive days, with each day's session lasting three and a half hours. Each session was divided into two working blocks with a 30-min break between. The first session covered the Skills lessons, the remaining three sessions were dedicated to the first three applied projects: Spellchecker, Predictive text and Word Games. The sessions were delivered in English, and supported in Spanish by a collaborator who had previously translated the interactive web course materials.

In terms of results, the materials were positively received. However, time constraints, along with the background knowledge of programming by most attendees, led to some difficulties which limited the exploration of more specialised and related lessons such as Regular Expressions (RegEx) and Machine Learning (ML).

Participants were able to create applications for several languages which they worked closely with, including word games in the endangered Indigenous languages Karijona and Umbra.

6.2 Santiago, Chile, January 2024

The second BELT workshop took place in Santiago, Chile, in early January 2024, as part of a 3-day workshop on Tecnologías Digitales y Lenguas Indígenas (Language Technologies and Indigenous Languages).[14] This was an abbreviated version of the workshop, offered during one three-hour morning session. Code from the word games project was then used on the third day to build word games for four different Indigenous languages.

The workshop had more than 40 participants, with 56% linguists, 20% speakers of Indigenous languages, 15% computer scientists, and 9% representatives of public policy organizations. Most presentations were delivered in Spanish, some in English, and a small number in local Indigenous languages. The BELT session was delivered in English, with real-time translation into Spanish.

Because of time constraints and the mixed backgrounds of participants, we grouped participants into teams of 2-5 people, ensuring that each team included at least one participant with prior experience writing code in Python. In this case, the main goals were: a) to introduce all participants to Python programming and to the range of topics available through BELT; b) to provide a foundation for a shared project on the third day of the workshop; and c) perhaps most importantly, to demonstrate the attainability of producing simple language technologies with a bit of data and a bit of code.

Some participants noted that, whether or not they intended to continue the BELT lessons, seeing first-hand how a programming language works, and how pieces can be put together to build a functioning system, opened their eyes to new possibilities for their languages. Other participants asked about using the lessons for teaching kids in school contexts. Localization of the course materials into Spanish was essential to the success of the workshop.

The third day of the workshop was set up as a hackathon, with several different interdisciplinary teams working on practical projects. We used code from the BELT word games project to produce hangman and word scramble games for Mapudungun, Aymara, Quechua, and Ckunsa. To make the games playable by new language learners, we compiled a list in Spanish of colors, numbers, animals, body part terms, and family relationship terms. Speaker and linguist participants then filled in the equivalents in the four Indigenous languages. The entire process took about three hours, and the eight games are hosted and playable through Hugging Face Spaces.[15]

[13] https://cienciassociales.uniandes.edu.co/congreso-de-las-lenguas-amazonicas/

[14] https://ws.dcc.uchile.cl/en/

[15] https://huggingface.co/TDLI2024

7 Participant Perspectives

(Note: This section is written from the perspective of our third author, who was a participant at the first workshop.)

As a descriptive linguist, the linguistic software I have used in the past—Phonology Assistant[16] and Fieldworks Language Explorer (FLEx)[17]—do not require any programming knowledge. My only experience with programming has been modifying small chunks of PRAAT scripts. The BELT workshop was an opportunity to get hands-on experience to build language tools for small resources languages and later to share my results with native speakers.

The website for the workshop provided a comprehensive and self-contained approach to the subject, with sessions clearly divided into lessons. The first lessons started with fundamentals, covering syntax and basic operators, progressing through numerical and set operations. Then, lessons became more complex, combining earlier lessons for tasks such as data preprocessing and manipulation. For several parts of the workshop, I and other participants were able to work with my own corpus.

I primarily work with Karijona, a Cariban Amazonian language with very little existing resources. The Karijona orthography and basic rules were discussed with the Puerto Nare Community in 2015 and since then standardized for educational use (Guerra et al., 2024), and the first language learning book was published just a few months before the workshop (Resguardo Indígena de Puerto Nare, 2023).

The book consisted of nine brief texts and many examples, all cross-checked with native speakers for transcription consistency. For the workshop, I uploaded these materials to my account on BELT in order to create apps in the Karijona language.

Following the lessons, we built a word unscrambling game and hangman game, which I was able to deploy through HuggingFace Spaces and even use on a mobile device. Then, I shared the apps with a group of Karijona speakers from Puerto Nare, who were able to try them during a trip to Bogotá. However, as there is very limited internet in Puerto Nare, making it difficult for speakers to use these apps regularly. Thus, it is an urgent priority to expand the BELT lessons to enable offline use apps that can be used on mobile devices.

8 Related Work

NLP and computational linguistics pedagogy has a rich history of research and applications, with much work in developing online learning resources (Artemova et al., 2021; Baglini and Hjorth, 2021) and live instruction techniques (Bender et al., 2008; Agarwal, 2013; Gaddy et al., 2021; Durrett et al., 2021; Kennington, 2021). Research has explored how best to teach language technology concepts to learners without a computer science background (Fosler-Lussier, 2008; Poliak and Jenifer, 2021; Vajjala, 2021) and in non-English instruction settings (Messina et al., 2021; Pannitto et al., 2021). Camacho and Zevallos (2020) recommends integration of (computational) linguistics into high school curricula as a method to fight language decline. However, to our knowledge, there are no existing educational resources for the development of endangered language technologies.

In general, the development of endangered language technology faces well-studied challenges (Doğruöz and Sitaram, 2022). Penttonen (2011) describes methods used to create online technologies such as dictionaries and language learning games for Karelian. Bird (2018) explores challenges in mobile applications specifically.

9 Conclusion

The BELT website offers a new resource for teaching beginning Python programming, with the direct goal of supporting development of language technologies for endangered languages. All course materials are freely available online and can be used to teach instructional workshops or for self-guided, asynchronous learning. Users need nothing more than a standard laptop and an Internet connection. It is our hope that these resources might spur new activity in community-based development of language technologies, thereby supporting data privacy and sovereignty for language communities.

Looking ahead, we are continuing development of BELT along several pathways. First, we would like to use the localization framework we built to translate the course materials into a wide range of languages of wider communication, such that the materials will be accessible to multilingual speakers around the world. Second, we have several planned additional skills and projects to add to the platform. In both of these directions, we welcome contributions from the broader NLP community.

We hope to develop a mobile version of BELT,

[16] https://software.sil.org/phonologyassistant/

[17] https://software.sil.org/fieldworks/

as mobile devices may be more common among younger members in many communities. Through this app, we hope to allow offline use of the applications created in BELT, for situations where users have intermittent internet access. Finally, we would like to explore the possibility of branching off BELT access such that versions of the website could be self-hosted, for example by tribal governments, immersion schools, summer schools, and the like.

Ethics Statement

First, it is important to note that we ourselves are not members of Indigenous communities, nor speakers of endangered languages. We offer these resources in the hopes that members of those communities can use them as a launching point to develop their own technologies in a manner that is entirely self-determined (Schwartz, 2022).

Second, we have been careful to develop BELT in a way that allows users to retain control over their own data. User-uploaded files are hosted on our server, but they are not copied or re-distributed in any way, nor are they available to other users of BELT. Users can delete these files at any time.

Finally, BELT is a freely-offered resource, currently supported by grant funding. We are committed to providing for sustained hosting for the website, so that these resources will remain available as long as they are relevant and useful. We will never charge users to learn using BELT.

Acknowledgments

BELT runs on the CUmulus platform provided by the University of Colorado Research Computing. Portions of this work were supported by the National Science Foundation under Grant No. 2149404, “CAREER: From One Language to Another”. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Apoorv Agarwal. 2013. Teaching the basics of NLP and ML in an introductory course to information science. In *Proceedings of the Fourth Workshop on Teaching NLP and CL*, pages 77–84, Sofia, Bulgaria. Association for Computational Linguistics.

Ekaterina Artemova, Murat Apishev, Denis Kirianov, Veronica Sarkisyan, Sergey Aksenov, and Oleg Serikov. 2021. Teaching a massive open online course on natural language processing. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 13–27, Online. Association for Computational Linguistics.

Rebekah Baglini and Hermes Hjorth. 2021. Natural language processing 4 all (NLP4All): A new online platform for teaching and learning NLP concepts. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 28–33, Online. Association for Computational Linguistics.

Emily M. Bender, Fei Xia, and Erik Bansleben. 2008. Building a flexible, collaborative, intensive master’s program in computational linguistics. In *Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics*, pages 10–18, Columbus, Ohio. Association for Computational Linguistics.

Steven Bird. 2018. Designing Mobile Applications for Endangered Languages. In *The Oxford Handbook of Endangered Languages*. Oxford University Press.

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.

Damian Blasi, Antonios Anastasopoulos, and Graham Neubig. 2022. Systematic inequalities in language technology performance across the world’s languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5486–5505, Dublin, Ireland. Association for Computational Linguistics.

Nathan Thanyehténhas Brinklow, Patrick Littell, Delaney Lothian, Aidan Pine, and Heather Souter. 2019. Indigenous language technologies and language reclamation in canada. In *Collection of Research Papers of the 1st International Conference on Language Technologies for All*, pages 402–406.

Luis Camacho and Rodolfo Zevallos. 2020. Language technology into high schools for revitalization of endangered languages. In *2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, pages 1–4.

Alberto Cardoso, Joaquim Leitão, and César Teixeira. 2019. Using the jupyter notebook as a tool to support the teaching and learning processes in engineering courses. In *The Challenges of the Digital Transformation in Education: Proceedings of the 21st International Conference on Interactive Collaborative Learning (ICL2018)-Volume 2*, pages 227–236. Springer.

A. Seza Doğruöz and Sunayana Sitaram. 2022. Language technologies for low resource languages: Sociolinguistic and multilingual insights. In *Proceedings of the 1st Annual Meeting of the ELRA/ISCA Special Interest Group on Under-Resourced Languages*, pages 92–97, Marseille, France. European Language Resources Association.

Greg Durrett, Jifan Chen, Shrey Desai, Tanya Goyal, Lucas Kabela, Yasumasa Onoe, and Jiacheng Xu. 2021. Contemporary NLP modeling in six comprehensive programming assignments. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 99–103, Online. Association for Computational Linguistics.

Darren Flavelle and Jordan Lachler. 2023. Strengthening relationships between indigenous communities, documentary linguists, and computational linguists in the era of NLP-assisted language revitalization. In *Proceedings of the First Workshop on Cross-Cultural Considerations in NLP (C3NLP)*, pages 25–34, Dubrovnik, Croatia. Association for Computational Linguistics.

Eric Fosler-Lussier. 2008. Strategies for teaching "mixed" computational linguistics classes. In *Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics*, pages 36–44, Columbus, Ohio. Association for Computational Linguistics.

David Gaddy, Daniel Fried, Nikita Kitaev, Mitchell Stern, Rodolfo Corona, John DeNero, and Dan Klein. 2021. Interactive assignments for teaching structured neural NLP. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 104–107, Online. Association for Computational Linguistics.

Eleonara Guerra, Víctor Narváez, and Camilo Robayo. 2024. Documentación y revitalización de la lengua karijona. Conferencia Día de las lenguas Nativas.

Jeremiah W Johnson. 2020. Benefits and pitfalls of jupyter notebooks in the classroom. In *Proceedings of the 21st annual conference on information technology education*, pages 32–37.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Casey Kennington. 2021. Natural language processing for computer scientists and data scientists at a large state university. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 115–124, Online. Association for Computational Linguistics.

András Kornai. 2013. Digital language death. *PLOS ONE*, 8(10):1–11.

Zoey Liu, Crystal Richardson, Richard Hatcher, and Emily Prud'hommeaux. 2022. Not always about you: Prioritizing community needs when developing endangered language technology. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3933–3944, Dublin, Ireland. Association for Computational Linguistics.

Martha Macri and James Sarmento. 2010. Respecting privacy: Ethical and pragmatic considerations. *Language & Communication*, 30(3):192–197.

Lucio Messina, Lucia Busso, Claudia Roberta Combei, Alessio Miaschi, Ludovica Pannitto, Gabriele Sarti, and Malvina Nissim. 2021. A dissemination workshop for introducing young Italian students to NLP. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 52–54, Online. Association for Computational Linguistics.

Ludovica Pannitto, Lucia Busso, Claudia Roberta Combei, Lucio Messina, Alessio Miaschi, Gabriele Sarti, and Malvina Nissim. 2021. Teaching NLP with bracelets and restaurant menus: An interactive workshop for Italian students. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 160–170, Online. Association for Computational Linguistics.

Martti Penttonen. 2011. Ict at service of endangered languages. In *Proceedings of the 11th Koli Calling International Conference on Computing Education Research*, pages 95–101.

Adam Poliak and Jalisha Jenifer. 2021. An immersive computational text analysis course for non-computer science students at barnard college. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 92–95, Online. Association for Computational Linguistics.

Ian Pool. 2016. Colonialism's and postcolonialism's fellow traveller: the collection, use and misuse of data on indigenous people. In Tahu Kukutai and John Taylor, editors, *Indigenous Data Sovereignty*, 1st edition. ANU Press.

Resguardo Indígena de Puerto Nare. 2023. *Karijona Womirï ehorï. ¡Aprendamos Karijona!* Fundación Tropenbos.

Lane Schwartz. 2022. Primum Non Nocere: Before working with Indigenous data, the ACL must confront ongoing colonialism. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 724–731, Dublin, Ireland. Association for Computational Linguistics.

Margaret Speas. 2009. Someone else's language on the role of linguists in language revitalization. *Indigenous Language Revitalization*, page 23.

Garrett Tanzer, Mirac Suzgun, Eline Visser, Dan Jurafsky, and Luke Melas-Kyriazi. 2024. A benchmark for learning to translate a new language from one grammar book. *Preprint*, arXiv:2309.16575.

Sowmya Vajjala. 2021. Teaching NLP outside linguistics and computer science classrooms: Some challenges and some opportunities. In *Proceedings of the Fifth Workshop on Teaching NLP*, pages 149–159, Online. Association for Computational Linguistics.

Chen Zhang, Xiao Liu, Jiuheng Lin, and Yansong Feng. 2024a. Teaching large language models an unseen language on the fly. *Preprint*, arXiv:2402.19167.

Kexun Zhang, Yee Man Choi, Zhenqiao Song, Taiqi He, William Yang Wang, and Lei Li. 2024b. Hire a linguist!: Learning endangered languages with in-context linguistic descriptions. *Preprint*, arXiv:2402.18025.

Shiyue Zhang, Ben Frey, and Mohit Bansal. 2022. How can NLP help revitalize endangered languages? a case study and roadmap for the Cherokee language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1529–1541, Dublin, Ireland. Association for Computational Linguistics.