Cerberus: Natural Language Processing Tools for Linguists

Geoff Bacon
Department of Linguistics
University of California, Berkeley
bacon@berkeley.edu

Abstract

Modern linguistics faces a "transcription bottleneck" in which more data is collected than can be expertly annotated. Modern natural language processing (NLP) provides a wealth of tools that can reduce the annotation burden of linguists, but using these tools requires technical proficiency in programming and NLP that most linguists do not have. To address this gap, we introduce Cerberus, an easy-to-use and open-source app that allows linguists to train and deploy their own models for annotation tasks like part-of-speech tagging, machine translation and classification. We discuss Cerberus' design and illustrate how it fits into a linguist's existing workflow with data from the low resourced language Wolof. We provide a quantitative evaluation of the trained models on 21 low resourced languages and a qualitative evaluation of the user experience. We end by describing our immediate and long-term plans to continually improve Cerberus with the help of the community.

Keywords: Software, low resource languages, POS tagging, machine translation, deep learning, linguistics

1. Introduction

With the resurgence of neural networks, modern natural language processing (NLP) has seen dramatic advances in our ability to process linguistic data with computers. Core tasks in NLP such as part-of-speech tagging (Bohnet et al., 2018; Plank et al., 2016; Wang et al., 2015), language modeling (Merity et al., 2017; Bengio et al., 2003), machine translation (Vaswani et al., 2017; Chen et al., 2018) and speech recognition (Chiu et al., 2018) all benefit from deep learning. Much of this improvement is attributed to their ability to learn representations in service of a particular task, in contrast to the use of hand-engineered features in earlier statistical models (Manning, 2015). Without the need for human feature-engineering, deep learning models apply directly to labelled data and are therefore language-agnostic in design.

At the same time that deep learning is revolutionizing NLP, linguistics, in particular the subfield of language documentation, is experiencing a "transcription bottleneck" (Arkhipov and Thieberger, 2018; Seifart et al., 2018). With the digital revolution, it is now relatively easy not only to compile a large corpus but also to archive and make it publicly available (Himmelmann, 2018). This is of great scientific and cultural importance, with over half the world's estimated 6,000 languages predicted to be extinct by the end of this century (Evans, 2011). Although compiling such corpora is comparatively straightforward, the next step in language documentation of annotating them with part-of-speech and other grammatical metadata is a time-consuming and expensive task (Foley et al., 2018). This asymmetry between the ease of collecting language resources and annotating them is often described as the transcription bottleneck in linguistics.

Recently, there has been renewed interest in the two communities of NLP and linguistics collaborating to address this transcription bottleneck. Bird (2009) called for the NLP community to focus on research that "accelerates the documentation and description of the world's linguistic heritage". Foley et al. (2019) built Elpis, a user-friendly interface to state-of-the-art speech recognition tools designed for linguists documenting endangered languages. Adams et al. (2018) introduced Persephone, a phonemic and tonal transcription tool for low resource settings. Bird and Chiang (2012) discuss the ways in which machine translation can support the urgent task of documenting the world's endangered languages. Van Esch et al. (2019) suggested a series of tools that would serve the language documentation community. While they did not implement any of the suggestions, van Esch et al. (2019) did identify which NLP tasks would be most beneficial to the field and motivate a "simple, pre-configured graphical user interface" for users "with limited technical knowledge" and an existing workflow and tools.

In this paper, we introduce Cerberus, an implementation of some tools suggested in van Esch et al. (2019). Cerberus is an easy-to-use and open-source app that reduces the annotation burden of linguists. Cerberus has an extensible architecture that currently supports part-of-speech tagging, machine translation and classification tasks. It allows linguists with no technical knowledge to train and use state-of-the-art NLP models on their language resources. Importantly, Cerberus is optimized for the comparatively low resourced settings common in language documentation.

The remainder of the paper is structured as follows. In the next section, we discuss the design considerations and implementation of Cerberus. We then use the low resourced language Wolof to illustrate its use in section 3. Section 4 presents a quantitative and qualitative evaluation of the software. Finally, section 5 outlines our plans for improving Cerberus and call for contributions.

2. Design

Cerberus is designed and implemented on five core principles: usability, compatibility, extensibility, optimization for low resourced languages and openness. In order to meet these requirements, we identified two principles that we do *not* need Cerberus to meet: customizability and scalability. We elaborate on these design requirements and non-requirements below. Following this, we provide high-level details on our implementation that we claim satisfies the design requirements.

2.1. Requirements

Usability Cerberus exists precisely to make NLP tools more usable to linguists, who often don't have strong programming skills or experience at the command line. To serve its users, Cerberus needs to be intuitive, have a zero barrier to entry and present only the most important information to users. This motivates a minimalist and consistent graphical user interface.

Compatibility We intend Cerberus to augment, not replace, linguists' workflows. These workflows currently rely on existing software such as Praat (Boersma and Weenink, 2019), ELAN (Sloetjes and Wittenburg, 2008) and FLEx[1] and their particular file formats. Cerberus is compatible with these formats, allowing users to switch between tools as needed.

Extensibility Many NLP methods are relevant to linguistic annotation tasks, only a small number of which have been implemented in Cerberus so far. Cerberus must allow for new tasks to be added in a thoughtful and consistent manner. Extending Cerberus for a new annotation task is as straightforward as writing less than 100 lines of Python code.

Optimization for low resourced languages Most NLP methods are designed and evaluated on high resource languages like English, where billions of tokens of text are readily available. In contrast, linguists typically work with corpora under 50,000 tokens and only hundreds of annotated sentences. Cerberus must be tailored to perform best with the comparatively low resources typical in language documentation settings.

Openness Open source collaboration produces high quality software and encourages reproducible research. Cerberus is an open source project under the MIT license in which contributions are welcomed.

2.2. Non-requirements

Customizability Users are not expected to have experience in NLP. This motivates a one-size-fits-all design over one that can be tweaked by power users. We have pre-selected models and hyperparameters so that users do not have to.

Scalability Cerberus is intended to be used by a single linguist working at their local machine on low resourced languages. In this goal, we do not receive any benefit from the ability to scale to multiple machines or larger datasets.

2.3. High level implementation details

To meet our requirement of extensibility, we implemented Cerberus entirely in Python, the language of choice for the vast majority of modern NLP research. Cerberus is implemented in two halves: a back end that handles the training, evaluation and predictions of models; and a front end that provides a user-friendly interface. The back end is built on AllenNLP (Gardner et al., 2017) and the front end is built with Streamlit.[2]

AllenNLP is a popular platform for deep learning based natural language processing. Input pipelines, models and training hyperparameters are defined in configuration files, making it easy for Cerberus developers to experiment with different settings. Cerberus defines a single model for each supported task as we do not need customizability. For part-of-speech tagging, we use a bidirectional long short-term memory network (Hochreiter and Schmidhuber, 1997, LSTM) predicting a tag for each token in the sentence. For classification tasks, we encode the text with a bidirectional LSTM which is then passed into a linear classifier. For machine translation, we use a unidirectional LSTM-based seq2seq architecture (Sutskever et al., 2014). Apart from these architectural choices, all other hyperparameters and training details are identical across all tasks. Words are represented with 50 dimensional embeddings concatenated with the 50 dimensional output of a unidirectional LSTM-based character encoder with 10% dropout. All other LSTMs use 100 dimensional hidden states and are trained with Adam (Kingma and Ba, 2014) in batches of 32 for 5 epochs.

Streamlit is an open-source front end framework in Python designed specifically for machine learning and data science apps. Streamlit apps run in the browser making them cross-platform. Using the same language in both the front and back end greatly simplifies our implementation. Cerberus also makes use of the visualization tool displaCy[3] to visualize the output of part-of-speech tagging and classification models.

Cerberus is tested on Python 3.6+ and runs on all platforms supporting those versions, including Windows, macOS and Linux. Regardless, we have found that the installation of software can be a difficult process and an impediment to its use. To achieve our usability requirements, we provide a Docker image[4] with Cerberus pre-installed.[5] This has become a standard practice among similar projects such as Elpis (citation) and Persephone (citation). For more details on our implementation, we refer the reader to our project homepage.[6]

3. Use

When users start Cerberus, they are greeted with the initial home page depicted in Figure 1. It gives a short description of what Cerberus is as well as instructions on its use. Users can return to these instructions from anywhere in the app. All functionality is accessed through the drop-down menu on the left-hand side. By choosing a task in this menu, such as part-of-speech tagging, Cerberus displays a single training button. Training a part-of-speech tagger is as simple as clicking this button. While training, Cerberus displays the estimated time remaining, as shown in Figure 2. Once training is complete, Cerberus is ready to accept an unlabelled sentence as input and produce the annotated output. This is illustrated in Figure 3, which shows Cerberus after having trained a part-of-speech tagger for the low resourced language Wolof. Given the unseen sentence *Mu mujj nag di nguur gu dëgër*, Cerberus produces the part-of-speech tags as output. In this example, the auxiliary verb *di* was

[1] https://software.sil.org/fieldworks/
[2] https://streamlit.io/
[3] https://spacy.io/universe/project/displacy
[4] https://opensource.com/resources/what-docker
[5] https://hub.docker.com/repository/docker/gbacon/cerberus
[6] https://github.com/geoffbacon/cerberus

incorrectly labelled as a pronoun but the annotation is otherwise correct. This short series of steps illustrates the full functionality of Cerberus for part-of-speech tagging, with the other tasks of classification and translation having an identical user experience. The user interface of Cerberus is intentionally kept minimalist to meet our usability requirement.

4. Evaluation

This section describes our own evaluation of Cerberus. In particular, we are interested in measuring two properties of the software. First, we measure how well the trained models perform in low resource settings. This helps answer the question, is Cerberus *useful* to linguists? Second, we study how well the interface functions. That is, is Cerberus *usable* by linguists?

4.1. Quantitative evaluation of trained models

Linguists are used to having high-quality data annotations (albeit in relatively low quantities), because the vast majority is annotated by hand. For Cerberus to be useful to linguists, it must provide high-quality annotations. In addition, it must learn to do so in low resource settings. Here, we evaluate this goal across the three currently implemented tasks with data from 21 low resourced languages.

We want to evaluate Cerberus in as many low resource languages as possible while also keeping the evaluations as comparable across languages as possible. These desires motivate the use of the Universal Dependencies treebanks (Nivre et al., 2016, UD) in our evaluation. The UD treebanks use a consistent schema across all languages to annotate naturally occurring sentences at the word level with rich grammatical information. We used the most recent release as of the time of writing, version 2.5, which contains data in 90 languages. For our evaluations, we selected all languages whose UD corpora total under 50,000 tokens, resulting in the 21 languages in Table 1 For part-of-speech tagging we used the universal part-of-speech categories (as opposed to the language-specific categories). To demonstrate the classification abilities in Cerberus, we wanted a task that i) would be the kind of task linguists might like to perform, and ii) is relevant across all languages. For this, we chose the task of predicting the grammatical number of the subject. For all languages evaluated, grammatical number is a binary distinction between singular and plural, except in North Sami and Scottish Gaelic which include a category for dual number. For translation, we extracted English translations when available. This process resulted in 20 languages for part-of-speech tagging, 11 for subject number classification and 2 for translation. We used Cerberus to train models for each task and language combination in which we had data. We reserved a randomly chosen 20% of the data to evaluate on and trained on the remaining 80%. The results along with the number of training examples for each model are given in Table 1.

Table 1 shows that while there is some variation between languages, on the whole the part-of-speech tagging results are positive. For most of the low resourced languages evaluated, Cerberus is able to train a part-of-speech tagger performing above 80% accuracy from as little as 1,500 labelled sentences. Visualizing the part-of-speech tagging results in Figure 4 illustrates that accuracy increases with the number of training examples available for that language. This suggests that users who require higher accuracy in Cerberus can often achieve this by manually annotating more data for their task.

The classification results in Table 1 are stronger and more consistent than the part-of-speech tagging results. Cerberus can accurately predict the grammatical number of the subject from an unlabelled sentence in a range of languages given only a few hundred examples. In our results, 900 labelled examples suffice to achieve over 80% accuracy. It is worth noting that the label space for part-of-speech tagging is much larger than that of subject number classification, making the part-of-speech tagging task more difficult than the classification task.

Unfortunately, the translation model performed poorly in this evaluation. For neither Naija nor Thai was Cerberus able to train a useful translation model. While this is a significantly harder task than part-of-speech tagging and classification, we believe improvements can be made to our current approach. These improvements are highlighted in section 5. Overall, these results show that Cerberus currently performs well in low resource settings for part-of-speech tagging and classification, while poorly for translation. This makes Cerberus a useful tool for linguists with room for improvement.

4.2. Qualitative evaluation of user experience

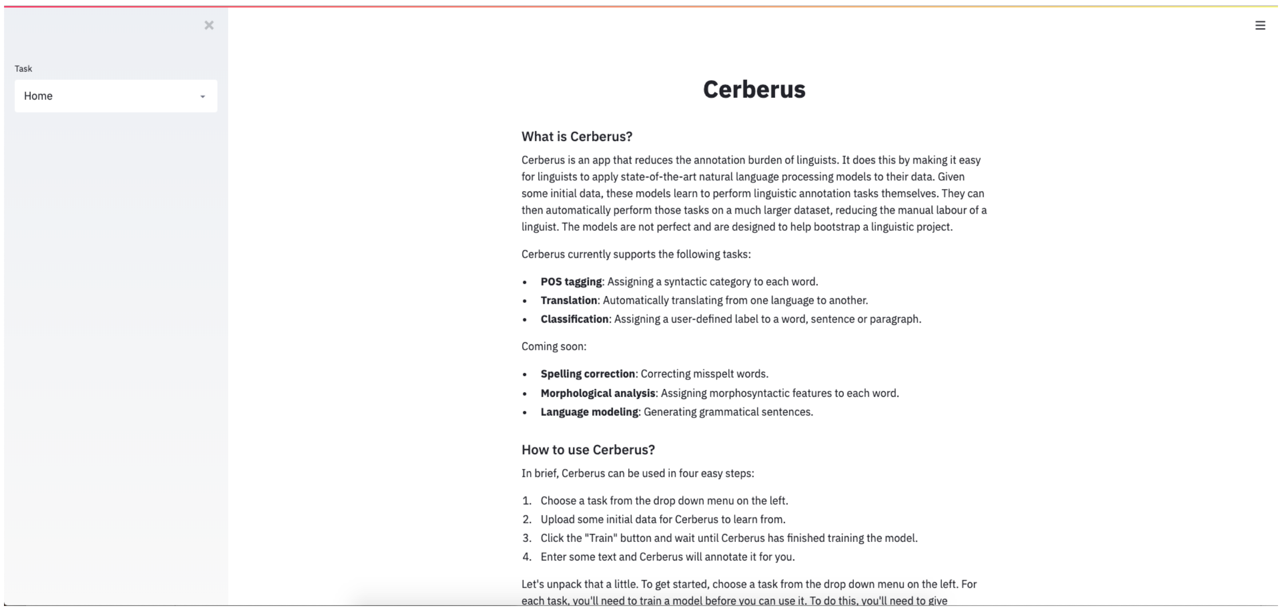
One of our core design requirements is usability. We expect users not to have technical experience in programming or NLP. Our user interface needs to allow such users to easily train and use models to annotate data in their existing workflow. We evaluated this goal by observing two users while using Cerberus for the first time.

We instructed both users to install Cerberus, train and use a model and otherwise explore the app. Users were only given access to information and instructions on the project's homepage and inside the app itself.

Installation Both users needed assistance installing the app. Although we intended for Cerberus to be usable without any programming experience, currently the installation requires running a single script from the command line, with or without the Docker image. While not an insurmountable impediment, we aspire to become completely command line free. Indeed, one option is a permanently hosted cloud service, although this is costly and requires users to upload their potentially sensitive data.

Navigation Users commented that navigating around the app was as intuitive and easy as they could imagine. The ability to always return to the home page instructions was critical in one user working with the software.

Training a model Although neither user had NLP or machine learning experience, both users understood the concept of training a model from the home page instructions. The clear link between the steps in the instructions and the layout of the individual task pages was appreciated. Until the first epoch of training is completed, the user interface currently displays that the estimated remaining training time is unknown. One user highlighted that this left them unsure



Cerberus

What is Cerberus?

Cerberus is an app that reduces the annotation burden of linguists. It does this by making it easy for linguists to apply state-of-the-art natural language processing models to their data. Given some initial data, these models learn to perform linguistic annotation tasks themselves. They can then automatically perform those tasks on a much larger dataset, reducing the manual labour of a linguist. The models are not perfect and are designed to help bootstrap a linguistic project.

Cerberus currently supports the following tasks:

- **POS tagging**: Assigning a syntactic category to each word.
- **Translation**: Automatically translating from one language to another.
- **Classification**: Assigning a user-defined label to a word, sentence or paragraph.

Coming soon:

- **Spelling correction**: Correcting misspelt words.
- **Morphological analysis**: Assigning morphosyntactic features to each word.
- **Language modeling**: Generating grammatical sentences.

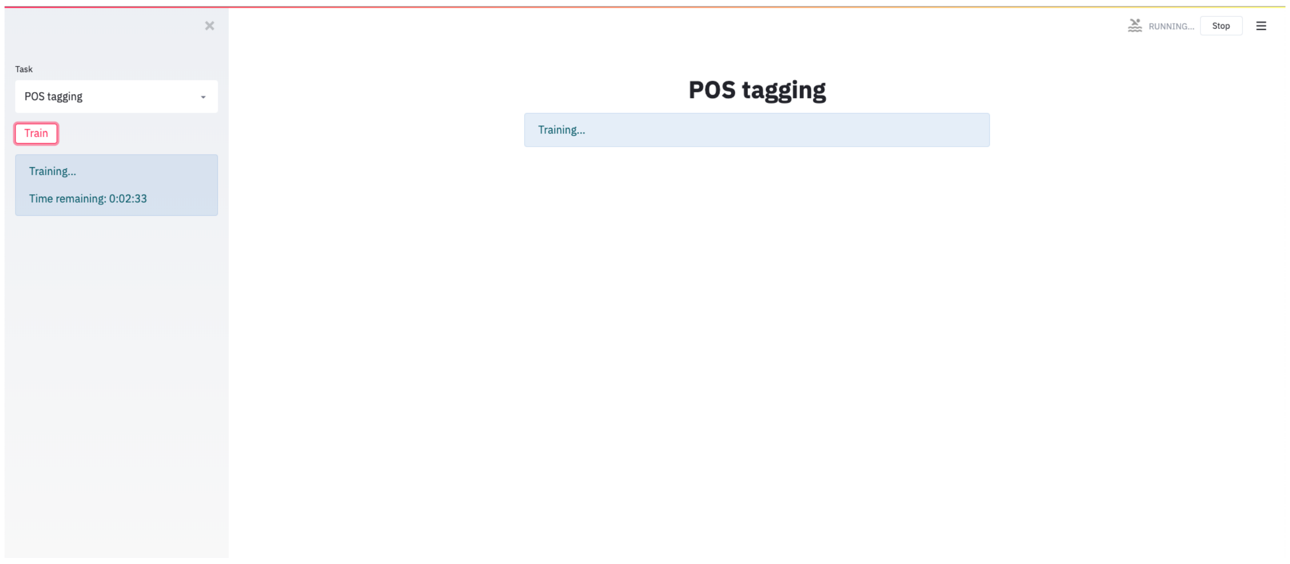
How to use Cerberus?

In brief, Cerberus can be used in four easy steps:

1. Choose a task from the drop down menu on the left.
2. Upload some initial data for Cerberus to learn from.
3. Click the "Train" button and wait until Cerberus has finished training the model.
4. Enter some text and Cerberus will annotate it for you.

Let's unpack that a little. To get started, choose a task from the drop down menu on the left. For each task, you'll need to train a model before you can use it. To do this, you'll need to give

Figure 1: The first thing users see when they start Cerberus. We designed Cerberus to be as intuitive and self-explanatory as possible. All the instructions for Cerberus' use are contained on this screen and are easily accessed from anywhere in the app.



POS tagging

Training...

Training...

Time remaining: 0:02:33

Figure 2: The user experience for training a model is designed to be maximally simple. On clicking the "Train" button, Cerberus will train a deep learning-based model for the task. In the case of part-of-speech tagging, this is a bidirectional LSTM tagger. Cerberus updates the estimated training time remaining after each of the five epochs.

how long they needed to wait and suggested an estimated time be given earlier if possible.

Using a trained model Having successfully trained models, both users found making predictions with the trained model simple and straightforward. The output was found to be easy to interpret.

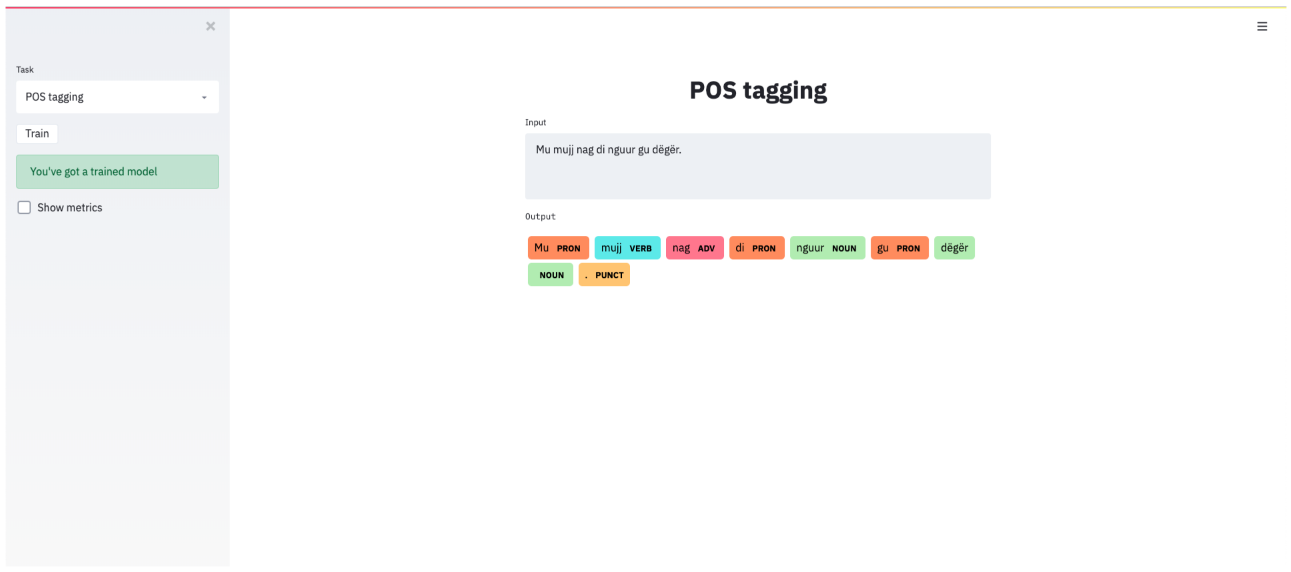
From this informal evaluation, we conclude that Cerberus partially satisfies its usability requirement with particular attention needed to improve the installation experience. We detail our steps to improve Cerberus in the next section.

5. Future work

Cerberus is a new project with plenty of room to grow. Our plans for improving Cerberus are focussed on the following four activities.

Conducting formal user evaluation To supplement the informal user evaluation discussed in section 4, we have designed a formal user evaluation based on the User Experience Questionnaire (Schrepp et al., 2017) and Jackson et al. (2011). We intend to administer the resulting survey in the immediate future and make the results public.

Improving translation results As discussed in section 4, the current translation model performs poorly on low resourced languages such as Naija and Thai. We will explore



Task: POS tagging

You've got a trained model

POS tagging

Input

Mu mujj nag di nguur gu dëgër.

Output

```
Mu PRON  mujj VERB  nag ADV  di PRON  nguur NOUN  gu PRON  dëgër NOUN  . PUNCT
```

Figure 3: Using Cerberus to tag a Wolof sentence with its part-of-speech categories. As in other parts of the app, the user interface is kept free of all but the most important information, in this case the words and their part-of-speech tags. In this example, the auxiliary verb *di* was incorrectly labelled as a pronoun but the annotation is otherwise correct.

Language	POS tagging		Classification		Translation	
	Accuracy	# training	Accuracy	# training	BLEU	# training
Afrikaans	91.5	1548	86.0	1116	-	-
Bambara	62.6	821	-	-	-	-
Belarusian	39.7	510	-	-	-	-
Buryat	55.6	742	-	-	-	-
Cantonese	63.8	804	-	-	-	-
Coptic	-	-	67.1	693	-	-
Erzya	65.5	1124	75.7	877	-	-
Faroese	74.9	967	92.8	896	-	-
Hungarian	82.2	1440	81.8	1260	-	-
Irish	82.0	1411	88.5	980	-	-
Maltese	89.9	1660	-	-	-	-
Mbya-Guarani	39.7	916	-	-	-	-
Naija	69.8	759	-	-	0.0	759
North Sami	82.4	2498	90.5	2013	-	-
Scottish Gaelic	82.4	1747	89.7	1202	-	-
Teugu	67.9	1063	-	-	-	-
Thai	50.2	800	-	-	0.0	800
Upper Sorbian	38.6	517	-	-	-	-
Uyghur	88.2	2765	90.1	772	-	-
Welsh	43.8	636	81.7	505	-	-
Wolof	74.8	951	86.2	1081	-	-

Table 1: Results from our quantitative evaluation on low resourced languages. For cross-linguistic comparability, we used data from the Universal Dependencies project (Nivre et al., 2016). We measure accuracy for part-of-speech tagging and classification and BLEU for translation. The numbers of training sentences available for each task are also given. Cerberus performed well on part-of-speech tagging, consistently well for subject number classification, but poorly on machine translation.

more complete hyperparameters searches as well as low resource neural machine translation architectures (Sennrich and Zhang, 2019; Gu et al., 2018; Ramesh and Sankaranarayanan, 2018) to improve our results.

Adding more tasks Many NLP methods usefully apply to linguistic annotation tasks. Van Esch et al. (2019) suggested many tasks, from which we immediately plan to implement spelling correction and constituency parsing models in Cerberus.

Growing user base and contributors Cerberus is an open-source project that welcomes contributions. The project's homepage[7] hosts all information for those interested in contributing or with suggestions for improvements.

[7] https://github.com/geoffbacon/cerberus

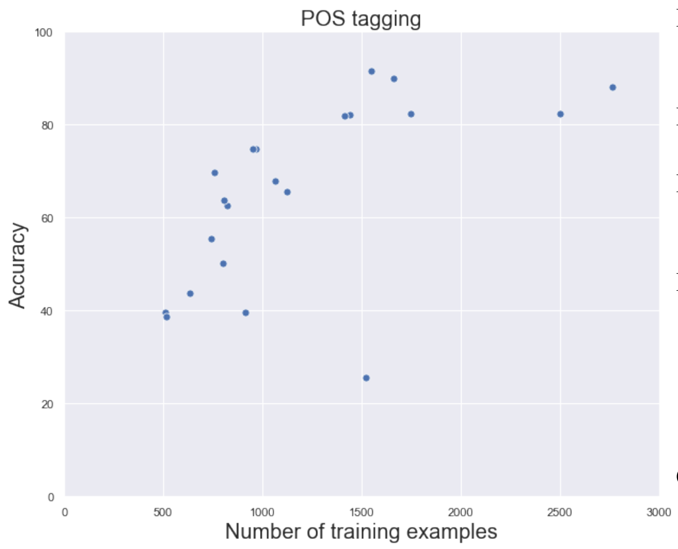


Figure 4: Accuracy on the part-of-speech tagging data as a function of the number of training examples. Each point represents one of the 20 low resourced languages. Accuracy increases with the number of training examples.

6. Conclusion

Deep learning has revolutionized NLP for the world's high resourced languages, but this technology has not been easily accessible to linguists working on less resourced and endangered languages. Building off existing toolkits such as Elpis (Foley et al., 2018) and Perspephone (Adams et al., 2018), van Esch et al. (2019) suggested future projects on NLP support for linguistics and language documentation. Van Esch et al. (2019) called for increased dialogue and collaboration between developers and fieldwork linguists. In this paper, we introduced Cerberus as our contribution to that dialogue. Cerberus is an easy-to-use and performant app for training and deploying NLP models for low resourced languages. Initial quantitative and qualitative evaluation has demonstrated its benefit in part-of-speech tagging and classification tasks, with more work needed on tuning its machine translation abilities. We outlined future directions for Cerberus and invited contributions from the community. We hope Cerberus will bring powerful computational tools to bear on new and as yet untested language resources, bringing modern linguistics and NLP into closer fruitful contact.

7. Bibliographical References

Adams, O., Cohn, T., Neubig, G., Cruz, H., Bird, S., and Michaud, A. (2018). Evaluating phonemic transcription of low-resource tonal languages for language documentation. In 11th international Conference on Language Resources and Evaluation (LREC 2018), pages 3356–3365.

Arkhipov, A. and Thieberger, N. (2018). Reflections on software and technology for language documentation. *Reflections on Language Documentation*, page 140.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Bird, S. and Chiang, D. (2012). Machine translation for language preservation. In 24th International Conference on Computational Linguistics, page 125.

Bird, S. (2009). Natural language processing and linguistic fieldwork. *Computational Linguistics*, 35(3):469–474.

Boersma, P. and Weenink, D. (2019). Praat: Doing phonetics by computer (version 6.1. 06)[computer program]. *Retrieved October*, 14.

Bohnet, B., McDonald, R., Simões, G., Andor, D., Pitler, E., and Maynez, J. (2018). Morphosyntactic tagging with a meta-bilstm model over context sensitive token encodings. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Chen, M. X., Firat, O., Bapna, A., Johnson, M., Macherey, W., Foster, G., Jones, L., Schuster, M., Shazeer, N., Parmar, N., and et al. (2018). The best of both worlds: Combining recent advances in neural machine translation. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Chiu, C.-C., Sainath, T. N., Wu, Y., Prabhavalkar, R., Nguyen, P., Chen, Z., Kannan, A., Weiss, R. J., Rao, K., Gonina, E., et al. (2018). State-of-the-art speech recognition with sequence-to-sequence models. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 4774–4778. IEEE.

Evans, N. (2011). Dying words: Endangered languages and what they have to tell us, volume 22. John Wiley & Sons.

Foley, B., Arnold, J. T., Coto-Solano, R., Durantin, G., Ellison, T. M., van Esch, D., Heath, S., Kratochvil, F., Maxwell-Smith, Z., Nash, D., et al. (2018). Building speech recognition systems for language documentation: The coedl endangered language pipeline and inference system (elpis). In SLTU, pages 205–209.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N. F., Peters, M., Schmitz, M., and Zettlemoyer, L. S. (2017). Allennlp: A deep semantic natural language processing platform.

Gu, J., Hassan, H., Devlin, J., and Li, V. O. (2018). Universal neural machine translation for extremely low resource languages. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 344–354.

Himmelmann, N. P. (2018). Meeting the transcription challenge. *Reflections on Language Documentation*, page 33.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jackson, M., Crouch, S., and Baxter, R. (2011). Software evaluation: criteria-based assessment. *Software Sustainability Institute*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Manning, C. D. (2015). Computational linguistics and deep learning. *Computational Linguistics*, 41(4):701–707.

Merity, S., Keskar, N. S., and Socher, R. (2017). Regularizing and optimizing lstm language models.

Nivre, J., De Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R. T., Petrov, S., Pyysalo, S., Silveira, N., et al. (2016). Universal dependencies v1: A multilingual treebank collection. In LREC.

Plank, B., Søgaard, A., and Goldberg, Y. (2016). Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Ramesh, S. H. and Sankaranarayanan, K. P. (2018). Neural machine translation for low resource languages using bilingual lexicon induced from comparable corpora. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop, pages 112–119.

Schrepp, M., Hinderks, A., and Thomaschewski, J. (2017). Construction of a benchmark for the user experience questionnaire (ueq). *IJIMAI*, 4(4):40–44.

Seifart, F., Evans, N., Hammarström, H., and Levinson, S. C. (2018). Language documentation twenty-five years on. *Language*, 94(4):e324–e345.

Sennrich, R. and Zhang, B. (2019). Revisiting low-resource neural machine translation: A case study. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Sloetjes, H. and Wittenburg, P. (2008). Annotation by category-elan and iso dcr. In 6th international Conference on Language Resources and Evaluation (LREC 2008).

Sutskever, I., Vinyals, O., and Le, Q. (2014). Sequence to sequence learning with neural networks. *Advances in NIPS*.

van Esch, D., Foley, B., and San, N. (2019). Future directions in technological support for language documentation. In Proceedings of the Workshop on Computational Methods for Endangered Languages, volume 1, page 3.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need.

Wang, P., Qian, Y., Soong, F. K., He, L., and Zhao, H. (2015). Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *arXiv preprint arXiv:1510.06168*.