Short-term projects, long-term benefits: Four student NLP projects for low-resource languages

Alexis Palmer and Michaela Regneri
Department of Computational Linguistics
Saarland University
Saarbrücken, Germany
{apalmer,regneri}@coli.uni-saarland.de

Abstract

This paper describes a local effort to bridge the gap between computational and documentary linguistics by teaching students and young researchers in computational linguistics about doing research and developing systems for low-resource languages. We describe four student software projects developed within one semester. The projects range from a front-end for building small-vocabulary speech recognition systems, to a broad-coverage (more than 1000 languages) language identification system, to language-specific systems: a lemmatizer for the Mayan language Uspanteko and named entity recognition systems for both Slovak and Persian. Teaching efforts such as these are an excellent way to develop not only tools for low-resource languages, but also computational linguists well-equipped to work on endangered and low-resource languages.

1 Introduction

There is a strong argument to be made for bringing together computational and documentary linguistics in order to support the documentation and description of endangered languages (Abney and Bird, 2010; Bird, 2009). Documentation, description, and revitalization work for endangered languages, as well as efforts to produce digital and machine-readable resources for languages currently lacking such data, benefit from technological support in many different ways. Here we focus on support via (a) tools facilitating more efficient development of resources, with easy learning curves, and (b) linguistic analysis tools.

Various meetings and workshops in recent years have helped to bring the two fields closer together, but a sizeable gap remains. We've come far enough to, for example, have a relevant workshop at a major computational linguistics conference, but not so far that issues around language endangerment are well-known to even a large subset of the computational linguistics community. One way to get computational linguists thinking about issues related to endangered languages is for them to get their hands dirty – to work directly on related projects. In this paper we describe our own local effort to bridge this gap: a course for Master's and Bachelor's students in computational linguistics in which small teams of students each produced working, non-trivial natural language processing (NLP) tools for low-resource languages (LRLs) over the span of a single semester. The individual projects are described in Section 3.

Such a course benefits the students in a number of ways. They get hands-on experience in system building, they learn about a new subfield within computational linguistics, with a different set of concerns (some of these are discussed in Section 2), and, in some cases, they get the opportunity to develop tools for their own native languages. From the perspective of computational work on endangered languages, the positive outcomes are not only a new set of NLP tools, but also a group of students and young researchers armed with experience working on low-resource languages and better equipped to take on similar projects in the future.

2 Teaching NLP for LRLs

Working on LRLs from a computational perspective requires training beyond the typical computational linguistics curriculum. It is not the case that the most widely-used methods from computational linguistics can be straightforwardly adapted for any arbitrarily-selected language. Thus an important part of our teaching agenda in this context is to familiarize students with the challenges inherent to NLP for LRLs as well as some of the main

Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages, pages 86–90,
Baltimore, Maryland, USA, 26 June 2014.

approaches for addressing these same challenges. This section briefly surveys some of the relevant issues, with pointers to representative studies.

The first and most obvious concern is *data sparsity*. Many of the most successful and widely-taught methods and models in computational linguistics rely on either large amounts of labeled data or massive amounts of unlabeled data. Methods and models explicitly addressing LRLs need to maximize the utility of available data. Approaches for addressing data sparsity range from data collection proposals (Abney and Bird, 2010) to leveraging high-resource languages (Xia and Lewis, 2007) to maximizing annotation effort (Garrette and Baldridge, 2013). A second concern is *model suitability*. Many existing models in computational linguistics implicitly encode or expect characteristics of high-resource languages (Bender, 2011); for example, much work on computational syntax uses models that exploit linear ordering of elements in utterances. Such models are not straightforwardly applicable for languages with free or flexible word order, nor for highly agglutinative languages where, for example, complete utterances are encoded as single words. Approaches to this issues include adaptation of models using linguistic knowledge and/or universals (Boonkwan and Steedman, 2011; Naseem et al., 2010). The third issue to note is the *difficulty of evaluation*. The output of systems or tools performing automated analysis are predictions of analyses for new data; these predictions must be evaluated against a ground truth or human-supplied analysis of the same data. Evaluation is difficult in the low-resource setting, both because of limited availability of expert-labeled data and because, in some cases, the ground truth isn't known, or analyses are shifting as knowledge about the language develops.

We began the course with a discussion of these issues, as well as an introduction to a range of existing tools, projects and resources. We did not explicitly teach programming skills in the course, but we also did not require extensive programming background. Rather, we aimed to balance the teams such that each contained a mix of backgrounds: a bit more than half of the students had previous experience with software development, and the rest had at least taken one introductory programming course. The projects were scoped such that there were clear ways for students without programming experience to contribute. For example, in some cases, students with extensive background in linguistics performed linguistic analysis of the data which informed the design of the system.

Evaluation of students was designed to emphasize three objectives: production of a working system, communication of challenges faced and solutions to those challenges, and personal development of professionally-relevant skills. Students were graded on their weekly progress (more detail in Section 3), one 15-20 minute talk per student, individual written reports detailing specific contributions to the project, and a conference-style end-of-semester poster and demo session. Systems were required to be working and demonstratable both at the midway point of the semester (as a simplified prototype) and at the end of the semester.

3 Four projects in four months

The course described here ("NLP tools for Low-Resource Languages") was offered as part of the regular curriculum for undergraduate and graduate students in the Computational Linguistics department at Saarland University. We started with 10 students and formed four teams (based on preferences for general topics and programming languages). The teams could choose their own project or select from a set of proposed topics.

During the teaching period, we regularly monitored the student's progress by using some methods of agile software development.[1] For each weekly meeting, each team had to set three goals which constituted their homework. Goals could be minor tasks (*fixing a certain bug*), bigger chunks (*choosing and implementing a strategy for data standardization*) or course requirements (*preparing a talk*). Not fulfilling a (project-related) goal was acceptable, but students had to analyze why they missed the goal and to learn from the experience. They were expected over the course of the semester to become better both at setting reachable goals and at estimating how long they would need to meet each goal. Under this obligation to make continuous, weekly progress, each team had a working system within three months. At the end of month four, systems were suitable for demonstration at the poster session.

The projects differ according to their scopes and goals, as well as their immediate practical utility.

[1] http://en.wikipedia.org/wiki/Agile_software_development

One project (3.1) makes previous research accessible to users by developing an easy-to-use frontend; a second project (3.2) aims to extend the number of languages addressed for an existing multilingual classification task; and the remaining two (3.3 and 3.4) implement language-specific solutions for individual language processing tasks. We additionally required that each project be open-source; the public code repositories are linked in the respective sections.

3.1 Small-vocabulary ASR for any language

This project[2] builds on existing research for small-vocabulary (up to roughly 100 distinct words) speech recognition. Such technology is desirable for, among other things, developing speech interfaces to mobile applications (e.g. to deliver medical information or weather reports; see Sherwani (2009)), but dedicated speech recognition engines are available only for a relatively small number of languages. For small-vocabulary applications, though, an existing recognizer for a high-resource language can be used to do recognition in the target language, given a pronunciation lexicon mapping the relevant target language words into sequences of sounds in the high-resource language. This project produces the required lexicon.

Building on the algorithms developed by Qiao et al. (2010) and Chan and Rosenfeld (2012), two students developed an easy-to-use interface that allows a user with no knowledge of speech technologies to build and test a system to recognize words spoken in the target language. In its current implementation, the system uses the English-language recognizer from the freely-available Microsoft Speech Platform;[3] for this reason, the system is available for Windows only. To build a recognizer for a target language, a user needs only to specify a written form and upload one or more audio samples for each word in the vocabulary; generally, the more audio samples per word, the better the performance. The students additionally implemented a built-in recorder; this means a user can spontaneously make recordings for the desired words. Finally, the system includes implementations of two different variants of the algorithm and an evaluation module, thus facilitating use for both research and development purposes.

The main challenges for this project involved managing the interaction between the algorithm and the Microsoft speech recognition platform, as well as getting familiar with development in Windows. The practical utility of this project is immediately evident: any user with a Windows machine can install the necessary components and have a working small-vocabulary recognizer within several hours. Of course, more time and data may be required to improve performance of the recognizer, which currently reaches in the mid-70s with five audio samples per word. These results, as well as further details about the system (including where to download the code, and discussion of substituting other high-resource language recognizers), are described in Vakil et al. (2014).

3.2 Language ID for many languages

This project[4] addresses the task of language identification. Given a string of text in an arbitrary language, can we train a system to recognize what language the text is written in? Excellent classification rates have been achieved in previous work, but for a relatively small number of languages, and the task becomes noticeably more difficult as the number of languages increases (Baldwin and Lui, 2010; Lui and Baldwin, 2012, for example). With few exceptions (Brown, 2013; Xia et al., 2010; Xia et al., 2009), existing systems have only attempted to distinguish between fewer than 200 of the thousands of written languages currently in use. This team of three students aimed to expand coverage of language identification systems as much as possible given existing sources of data.

To do this, they first needed to gather and standardize data from various sources. They targeted three sources of data: the Universal Declaration of Human Rights, Wikipedia,[5] ODIN (Lewis and Xia, 2010), and some portions of the data available from Omniglot.[5] The challenges faced by this group lay primarily in two areas: issues involving data and those involving classification. In the first area, they encountered expected and well-known issues such as clean-up and standardization of data, dealing with encoding issues, and managing large amounts of data. The second set of challenges have to do with the high degree of skew in the data collected. Though their system covers over 1000 languages, the amount of data per language ranges from a single sentence to hundreds of thousands of words. Along the way, the students realized that this collection of data in a stan-

[2] https://github.com/lex4all/lex4all

[3] http://msdn.microsoft.com/en-us/library/hh361572

[4] https://github.com/alvations/SeedLing

[5] http://www.wikipedia.com, http://www.omniglot.com

dard, machine-readable form is useful for many other purposes. The corpus and how to access it are described in Emerson et al. (2014). A second paper presenting the language identification results (including those for low-resource languages) is planned for later this year.

3.3 A lemmatizer for Uspanteko

The third project[6] involved implementing a lemmatizer for the Mayan language Uspanteko. Using data that had been cleaned, standardized (as described in Palmer et al. (2010)), and made available through the Archive of Indigenous Languages of Latin America,[7] these three students implemented a tool to identify the citation form for inflected word forms in texts. The lemmatization algorithm is based on longest common substring matching: the closest match for an inflected form is returned as the lemma. Additionally, a table for irregular verb inflections was generated using the annotated source corpus (roughly 50,000 words) and an Uspanteko-Spanish dictionary (Can Pixabaj et al., 2007), to map inflected forms translated with the same Spanish morpheme.

This group more than any other faced the challenge of evaluation. Not all lemmas covered in the texts appear in the dictionary, and the Uspanteko texts, though fully analyzed with morphological segmentation and glossing, part of speech tags, and translation into Spanish, do not include citation forms. Manual evaluation of 100 sentences, for which a linguist on the team with knowledge of Spanish determined citation forms, showed accuracy of 59% for the lemmatization algorithm.

3.4 NER for Slovak & Persian

Finally, the fourth project[8] (two students) chose to tackle the task of named entity recognition (NER): identifying instances of named entities (*NEs*, e.g. people, locations, geopolitical entities) in texts and associating them with appropriate labels. The students developed a single platform to do NER in both Slovak and Persian, their native languages. The approach is primarily based on using gazetteers (for person names and locations), as well as regular expressions (for temporal expressions). The students collected the gazeteers for the two languages as part of the project. Their system builds on a modular design; one can swap out gazetteers and a few language-specific heuristic components to perform NER in a new language.

In this project, resource acquisition and evaluation were the main challenges. The students used some existing resources for both languages, but also devoted quite some time to producing new gazetteers. For Slovak, additional challenges were presented by the language's large number of inflectional cases and resulting variability in form. For example, some inflected forms used to refer to people from a given location are string-identical to the names of the locations with a different case inflection. In Persian, the main challenges were detection of word boundaries (many names are multi-word expressions) and frequent NE/proper noun ambiguities. For evaluation, the students hand-labeled over 35,000 words of Slovak (with 545 NE instances) and about 600 paragraphs of Persian data (306 NE instances). Performace varies across named entity category: temporal expression matching is most reliable (f-score 0.96 for Slovak, 0.89 for Persion), followed by locations (0.78 Slovak, 0.92 Persian) and person names (0.63 Slovak, 0.87 Persian). Note that for Persian, only NEs with correctly matched boundaries are counted (which are 50% for persons).

4 Conclusion

In this paper we have presented four student software projects, each one addressing a different NLP task relevant for one or more low-resource languages. The successful outcomes of the four projects show that much progress can be made even with limited time and limited prior experience developing such systems. Local teaching efforts such as these can be highly successful in building a group of young researchers who are both familiar with issues surrounding low-resource and endangered languages and prepared to do research and development in this area in the future. We think of this as planting seeds for an early harvest: with one semester's combined effort between instructors and students, we reap the rewards of both new tools and new researchers who can continue to work on closing the gap between computational and documentary linguistics.

Course materials are publicly available from the course homepage,[9] and from the project repositories linked from the descriptions in Section 3.

[6] https://code.google.com/p/mayan-lemmatizer/

[7] http://www.ailla.utexas.org

[8] https://code.google.com/p/named\-entity\-tagger/

[9] http://www.coli.uni-saarland.de/courses/cl4lrl-swp/

Acknowledgements

First of all, we want to thank the students who participated in our course and put so much effort and passion in their projects. They are (in alphabetical order): Christine Bocionek, Guy Emerson, Susanne Fertmann, Liesa Heuschkel, Omid Moradiannasab, Michal Petko, Maximilian Paulus, Aleksandra Piwowarek, Liling Tan and Anjana Vakil. Further, we want to thank the anonymous reviewers for their helpful comments. The second author was funded by the Cluster of Excellence "Multimodal Computing and Interaction" in the German Excellence Initiative.

References

Steven Abney and Steven Bird. 2010. The Human Language Project: Building a universal corpus of the world's languages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 88–97. Association for Computational Linguistics.

Timothy Baldwin and Marco Lui. 2010. Language identification: The long and the short of the matter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 229–237, Stroudsburg, PA, USA. Association for Computational Linguistics.

Emily M Bender. 2011. On achieving and evaluating language-independence in NLP. *Linguistic Issues in Language Technology*, 6(3):1–26.

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

Prachya Boonkwan and Mark Steedman. 2011. Grammar induction from text using small syntactic prototypes. In *IJCNLP*, pages 438–446.

Ralf D Brown. 2013. Selecting and weighting n-grams to identify 1100 languages. In *Text, Speech, and Dialogue*, pages 475–483. Springer.

Telma Angelina Can Pixabaj, Oxlajuuj Keej Maya' Ajtz'iib' (Group) Staff, and Centro Educativo y Cultural Maya Staff. 2007. *Jkemiix yalaj li uspanteko*. Cholsamaj Fundacion, Guatemala.

Hao Yee Chan and Roni Rosenfeld. 2012. Discriminative pronunciation learning for speech recognition for resource scarce languages. In *Proceedings of the 2nd ACM Symposium on Computing for Development*, page 12. ACM.

Guy Emerson, Liling Tan, Susanne Fertmann, Alexis Palmer, and Michaela Regneri. 2014. SeedLing: Building and using a seed corpus for the Human Language Project. In *Proceedings of ACL Workshop on the use of computational methods in the study of endangered languages (ComputEL)*.

Dan Garrette and Jason Baldridge. 2013. Learning a part-of-speech tagger from two hours of annotation. In *Proceedings of NAACL-HLT*, pages 138–147.

William D Lewis and Fei Xia. 2010. Developing ODIN: A multilingual repository of annotated language data for hundreds of the world's languages. *Literary and Linguistic Computing*, 25(3):303–319.

Marco Lui and Timothy Baldwin. 2012. Langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, ACL '12, pages 25–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1244. Association for Computational Linguistics.

Alexis Palmer, Taesun Moon, Jason Baldridge, Katrin Erk, Eric Campbell, and Telma Can. 2010. Computational strategies for reducing annotation effort in language documentation. *Linguistic Issues in Language Technology*, 3.

Fang Qiao, Jahanzeb Sherwani, and Roni Rosenfeld. 2010. Small-vocabulary speech recognition for resource-scarce languages. In *Proceedings of the First ACM Symposium on Computing for Development*, page 3. ACM.

Jahanzeb Sherwani. 2009. *Speech interfaces for information access by low literate users*. Ph.D. thesis, SRI International.

Anjana Vakil, Max Paulus, Alexis Palmer, and Michaela Regneri. 2014. lex4all: A language-independent tool for building and evaluating pronunciation lexicons for small-vocabulary speech recognition. In *Proceedings of ACL2014 Demo Session*.

Fei Xia and William Lewis. 2007. Multilingual structural projection across interlinear text. In *Proceedings of HLT/NAACL 2007*, Rochester, NY.

Fei Xia, William D Lewis, and Hoifung Poon. 2009. Language ID in the context of harvesting language data off the web. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 870–878. Association for Computational Linguistics.

Fei Xia, Carrie Lewis, and William D Lewis. 2010. The problems of language identification within hugely multilingual data sets. In *LREC*.