**Automatic Unsupervised Syllabification Algorithm**

By Elias Stengel-Eskin and Emily Mulhall

Obtaining an accurate syllabification of words in a language has significant applications in a number of fields in linguistics. For example, many phonological rules depend on syllable boundaries or are applied within a syllable. In phonetics, the process of syllabification is widely discussed, as the phonemes that can be nuclei vary among languages. However, there is still work to be done in syllabification algorithms.

Though a variety of syllabification algorithms exist, many are language dependent -- for example, the maximal onset algorithm, which operates on the maximal onset principle (“we should extend a syllable’s onset at the expense of the preceding syllable’s coda whenever it is legal to do so,” (Bartlett, Cherry, Kondrak)). Additionally, there are algorithms based on sonority, a principle stating “sonority should increase from the first phoneme of the onset to the syllable’s nucleus and then fall off to the coda,” (Bartlett, Cherry, Kondrak). Furthermore, there are algorithms based on legality, a principle stating that “a syllable is not allowed to begin with a consonant cluster that is not found at the beginning of some word, or end with a cluster that is not found at the end of some word,” (Bartlett, Cherry, Kondrak). However, all three of these algorithms vary greatly in accuracy depending on the language of the test set, as their precepts are violated to varying degrees among different languages.

Another category of syllabification algorithms is supervised learning algorithms. For example, Müller created an algorithm that “generates all possible onsets, nuclei, and codas, based on the phonemes existing in the language” to form a grammar and then is “made probabilistic using counts obtained by training data”. Though language-independent, this algorithm requires training data. (Bartlett, Cherry, Kondrak). Another example is algorithms based on Hidden Markov Models and support vector machines. While highly accurate, this algorithm also requires a sizeable training set.

While many of these algorithms are very successful, they still either vary greatly in accuracy based on language or are supervised, meaning that pre-existing syllabification must exist for the given language. Thus, we chose to create an automatic, unsupervised, language independent syllabification algorithm consisting of three main parts.

The first part of the algorithm generates a context free grammar (CFG) that models how a word is syllabified. The second part uses an adaptor grammar (AG) to go through all possible expansions of non-terminal nodes and determine which expansion is most likely. The third then parses through all tree expansions produced by the AG and selects the most probable one.

To implement the second part, we used a Python adaptation of previous work done by Mark Johnson built by the Cloud Computing Research Team at the University of Maryland. Mark Johnson’s algorithm applies the Pitman-Yor process to a probabilistic context-free grammar (PCFG). A CFG consists of non-terminal symbols, terminal symbols, the start symbol, and a set of rules. A PCFG also consists of these four parameters, with an added fifth, which is the set of probabilities specifying how likely the expansion of a particular node to another is (Bartlett, Cherry, Kondrak). The Pitman-Yor process then adjusts these probabilities based on how frequent the expansions are.

To use the AG algorithm, we created a CFG that models syllabification, reading in a user specified file containing a list of syllabic segments, a list of non-syllabic segments, and semi-syllabic segments. This file accounts for differences in syllabic phonemes amongst languages.

After going through the AG, the syllabification algorithm parses the tree expansions to find the most likely one and then writes the corresponding syllabification for the word into a separate file. In order to test our algorithm, we used Lexique to build a French corpus, and CELEX to build German, English and Dutch corpora. To check our accuracy, we compared the syllabifications generated by our algorithm to syllabifications obtained from CELEX or Lexique.

Initially, the algorithm was tested with subsets of the French and German corpora. While the algorithm resulted in syllabifications often very close to the correct ones, most having a Levenshtein distance from the correct syllabification of two or under, the accuracy for German was only about 19%, and French was not much better at 22%. However, we adjusted our grammar to favor onsets over codas, reflecting maximal onset, and saw a leap in accuracy. The entire German corpus, which consists of over 50,000 words, was syllabified with about 44% accuracy.

One major disadvantage of the algorithm is the runtime. Testing whole corpora can take days on single core computers. Thus, our results for other languages are from testing subsets of corpora. In English, we were able to achieve 48% accuracy for a subset of 5,000 words. In French we were able to achieve about 45% on a subset of a 2,500 words, and on Dutch, we were able to achieve an accuracy of 42% on a subset of 5,000 words.

While our algorithm does require the user to know which phonemes are syllabic in a language, the results are consistent across the four languages we tested. The efficiency of the algorithm, which is mostly due to the adaptor grammar, could be improved, and the accuracy could also be improved with further research into syllabification. However, we implemented an unsupervised, language independent algorithm that is more accurate and more consistent than some previous algorithms, such as the maximal onset algorithm. Thus, our algorithm shows great promise as a possible choice for unsupervised non-deterministic language-independent syllabification.

Works Cited

Bartlett, Susan, Grzegorz Kondrak, and Colin Cherry. "On the Syllabification of Phonemes." *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL* (2009): 308-16. *Association for Computational Linguistics*. Association for Computational Linguistics. Web. 1 Dec. 2016.

Cohen, Shay B., David M. Blei, and Noah A. Smith. *Variational Inference for Adaptor Grammars* (2010): n. pag. *Association for Computational Linguistics*. Association for Computational Linguistics. Web. 1 Nov. 2016.

Cox, Felicity, Jonathan Harrington, and Robert Mannell. "The Syllable: Introduction." *Macquarie University* (n.d.): n. pag. Macquarie University, 1 Aug. 2009. Web. 31 Oct. 2016.

Hardisty, Eric A., Jordan Boyd-Graber, and Philip Resnik. "Modeling Perspective Using Adaptor Grammars." *In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing* (2010): 284-92. *Association for Computational Linguistics*. Association for Computational Linguistics. Web. 30 Oct. 2016.

Johnson, Mark, Sharon Goldwater, and Thomas L. Griffiths. *Adaptor Grammars: A Framework for Specifying Compositional Nonparametric Bayesian Models* (2006): 641-48. Web. 31 Oct. 2016.

Zhai, Ke, Shay B. Cohen, and Jordan Boyd-Graber. "Online Adaptor Grammars with Hybrid Inference." *Association for Computational Linguistics*. Association for Computational Linguistics, 2014. Web. 31 Oct. 2016.