Vietnamese word segmentation

Problem

Word segmentation is the problem of dividing a text into words. Word delimiters are not used in Vietnamese, so word segmentation is key in Vietnamese NLP. Word segmentation would be useful for the extraction of morpheme/word contexts and frequencies from texts.

Example. There is a many-to-one correspondence between morphemes and their representations in the Latin script but a nearly one-to-one correspondence between morphemes and their representations in the Han script. Thus, it would be helpful to be able to transliterate the Latin representation of a text into Han. There is a nearly one-to-one correspondence between the representations of words in the Latin script and the representations of words in the Han script. To illustrate, *công* could be 工, 功, 公, or 攻 (labor, achievement, public, and attack, respectively), but *công tác* must be 工作 (official business). Thus, the transliteration of the Latin representation of a word into Han is easy; hence, word segmentation would be useful.

Note that although Vietnamese is not written with word delimiters, it *is* written with *sentence* delimiters. Thus, the problem is reduced to dividing a sentence into words. Furthermore, note that Vietnamese is written with syl-

lable delimiters (usually spaces and sometimes dashes).

Approach

State-of-the-art Vietnamese word segmentation approaches include dictionary and rule-based approaches, HMMs, CRFs, and SVMs. This project explores approaches based on transitional frequencies and HMMs. All text will be normalized by converting to Unicode NFC form, then converting to lowercase, and finally removing nonalphabetical characters.

For evaluation, the corpus will be divided into a training set and a test set. The accuracy, precision, recall, and F1 will be computed for both approaches on the test set. For these metrics, what is being considered is the syllable boundaries. A true positive will be a syllable boundary that is also correctly posited as word boundary, a true negative will be a syllable boundary is that correctly posited as *not* being a word boundary, etc. They will also be computed for two state-of-the-art algorithms on the same test set.

Data

The following data will be used:

EVBNews 2.0 corpus: 832,441 words over 45,531 sentences over 1,000 files; includes annotations for word segmentation (Ngo et al. 2013).

Wortschatz Leipzig Vietnamese 2022 news corpus: 1,000,000 sentences;

does not include annotations for word segmentation (Leipzig University 2022).

Note that to run the code, you need to put the EVBCorpus_EVBNews_v2.0 and vie_news_2022_1M directories in the same folder as the script. The former should contain files named "NXXXX.sgml", and the latter should contain a file called "vie_news_2022_1M-sentences.txt".

Transitional frequencies

Saffran et al. 1996 suggests that infants use syllable bigram frequencies to perform word segmentation.

Modeling after Saffran et al. 1996, the following algorithm was implemented. First, count the syllable bigram frequencies of the training data, and then, for each sentence in the test data, posit a boundary between two syllables if the syllable bigram frequency for those two syllables is sufficiently low. The threshold was learned by withholding a small part of the test data (N0001.sgml). The syllable bigram frequencies were normalized by scaling to occurrences per million.

HMMs

Modeling after Jurafsky and Martin 2023, chapter 17 and appendix A, word segmentation was modeled using an HMM in the following manner. The HMM has three hidden states: beginning of word, inside of word, and outside of word.

The observations are syllables. The probabilities were learned by counting (cf. the small hot-cold example in Jurafsy and Martin 2023, appendix A page 11). Then, for each sentence in the test data, word boundaries were posited by finding the best hidden state sequence given the HMM that was learned. The algorithm used for that was the Viterbi algorithm.

Evaluation

	accuracy	precision	recall	F1
my Saffran algo	0.64	0.64	0.96	0.77
my Saffran algo with loners	0.70	0.70	0.91	0.79
my HMM	0.67	0.73	0.79	0.76

Note that there is a paper that considers recall to be a better metric than precision (Shao 2019). This makes sense because recall is to be preferred when false negatives are more costly than false positives. Longer words are rarer, so missing a longer word is more costly, so recall should be preferred over precision. Also, accuracy is usually not a good metric for imbalanced data sets because of overfitting.

At first, the Saffran-style algorithm had very low accuracy, precision,

and F1 but a surprisingly high recall. Examining some of the segmentations it produced, immediately there was the impression that it favored lumping over splitting. I also found a few potential places for improvement. For example, certain high frequency monosyllabic words would keep getting lumped with other words, such as *là* (the verb *to be*). In English, *is* is often followed by *a*, but *is a* is not a word; something similar was happening here. I attempted to improve the situation by having the algorithm always treat certain syllables as their own words. I manually picked out a few words for my set. I call the set "loners". Unfortunately, the recall took a hit from adding in the loners, but accuracy, precision, and F1 went up a bit (~+0.5). The set I picked did include some troublesome words, words that although often occur freely also often occur bound. Curiously, the threshold learned without this lexical knowledge was quite high (370), whereas with this tiny bit of lexical knowledge the threshold was quite low (24).

The HMM had surprising poor performance, worse than the Saffran-style algorithm, which I did not expect. Looking at the segmentations it produced, nothing seemed extremely weird. It seems to favor splitting over lumping. At the present, I do not have any ideas for improvement. I would need to do more reading.

I said in my project proposal that I would run some state-of-the-art algorithms and compare their scores with mine, but I opted not to for now.

Concluding remarks

I am not sure if further investigation into improvements for the Saffran-style algorithm would yield much. It was very fast and required little lexical knowledge, but the HMM was just as fast to train and only a bit harder to understand/implement.

I also find some of the choices of word segmentation in EVBNews a bit strange, but also reasonable. For example, N0777.sgml, no. 192, puts *nhiều nốt ruồ* as one word. Here, *nhiều* (many) does not appear in the English sentence. The whole three syllables corresponds to *moles* in the English sentence, but really, only *nốt ruồ* means *moles*. This may have affected the results quite a bit.

References

Jurafsky, Dan, and Martin, James H. 2023. *Speech and Language Processing*, third edition draft. https://web.stanford.edu/~jurafsky/slp3/.

Ngo Quoc Hung, Werner Winiwarter, and Bartholomaus Wloka. 2013. EVB-Corpus: A multi-layer English-Vietnamese bilingual corpus for studying tasks in comparative linguistics. *Proceedings of the Eleventh Workshop on Asian*

Language Resources, 1–9. Taipei: Asian Federation of Natural Language Processing.

Saffran, Jenny R., Richard N. Aslin, and Elissa L. Newport. 1996. Statistical learning by 8-month-old infants. *Science* 274, 1926–1928. Washington, D.C.: American Association for the Advancement of Science.

Shao Yan, Christian Hardmeier, and Joakim Nivre. 2017. Recall is the proper evaluation metric for word segmentation. *Proceedings of the Eighth International Joint Conference on Natural Language Processing* 2, 86–90. Taipei: Asian Federation of Natural Language Processing.

Vietnamese news corpus based on material from 2022. 2022. Leipzig Corpora Collection. https://corpora.uni-leipzig.de/en?corpusId=vie_news_2022.