Trying Tries: Stochastic Morpheme Segmentation

Denizhan Pak
Dept. of Linguistics - Indiana University
denpak@iu.edu

Abstract

There are many reasons to apply morphological analysis for complex linguistic tasks. The application of computational tools to corpus data is heavily improved by increased information. Morphological analysis can provide novel information that can benefit downstream tasks. Semantics and dependencies can also be captured more easily given more morphological information. To determine the list of morphemes in a language however is a time consuming task and an unsupervised algorithm could relieve quite a few researchers and grad students. In this paper we propose a potentially useful unsupervised machine learning to accomplish just this task.

1 Introduction

The process of finding meaningful sub-sequences in a larger sequential dataset is a difficult and important task. This is especially true in linguistics. Morphology is a crucial element of any coherent study of language. Being able to identify morphemes inside linguistic data is therefore crucial. Since it is such a core linguistic concept many morphologies have already been mapped out and gold standards have been generated for languages such as Finnish and English. This task however becomes much hard for languages which don't have clear word boundaries and which have don't have large amounts of data available.

We propose using a markovian model inspired the construction of markov chains in other algorithms such Metropolis Monte Carlo Markov chains. The algorithm will be capable of generating a markov chain which can be used to derive the morphemes in a language as well as used to parse a corpus into a constituent morpheme sequence. Since the algorithm is designed in terms of sequences and sub-sequences it could be used in other contexts as well ranging from word boundary detection to binary sequence parsing.

2 Related Work

Although much morphological data has already been generated using traditional linguistic tools. The emergence of computational linguistics has included the introduction of many other morphological parsing algorithms into the literature. Many of these algorithms involve the supervised learning however unsupervised methods have also been developed. This methods have varying degrees of complexity and have applied a variety of different techniques. Ranging from expectation maximization to Bayesian inferencea.

Including general morphological segmentation there have also been developments in recognition specialized morphemes based on occurrence location such as suffix-segmentation. Much of the work being done has been focused on languages which strong morphologies Turkish, Finnish, and because of the availability of data English and Chinese.

3 Data

The data for this project comes from a variety of sources. For corpus data and comparison to other unsupervised algorithms the model will be tested against other unsupervised algorithms on the publicly available wikimedia data for Turkish, English, Finnish, Chinese. This dataset has been cleaned of non-lexical content.In addition the generated morpheme lists will be compared against the gold standard dataset generated for English and Finnish.

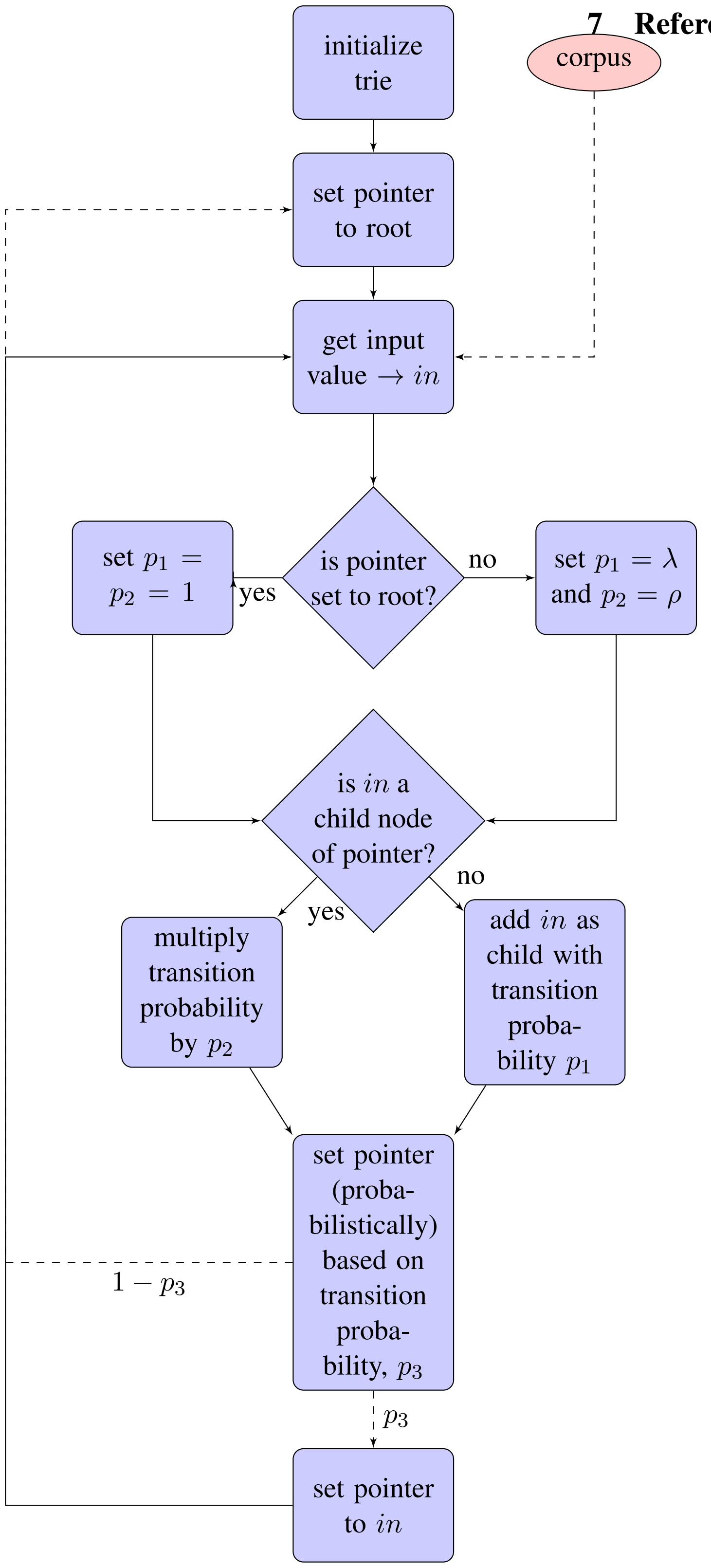
4 Method

Morpheme parsing is an important task from a computational linguistics standpoint as it provides a way to denote meaningful units within a corpus in turn this allows us to apply the many tools which use these units using the lowest components which still provide information. We propose that the semantic importance of a morpheme can be determined by its relative frequency. More explicitly if we are able to identify substrings which occur as a cohesive unit with a high relative frequency then those units correspond to morphemes or some other sort of semantic unit. The algorithm below provides an unsupervised method through which such semantic units can be distinguished.The algorithm uses a data structure similar to a "trie" however edges between nodes are assigned a transition probability, we shall call this a "p-trie." The algorithm requires two user determined hyper parameters: $0 < \lambda < 1 < \rho$. Where ρ is a reinforcement rate and λ is a learning rate. At the end of the algorithm the function will have returned a $p - trie$ with the estimated probability values. Once we have a $p-trie$ it is possible to extract potential morphemes as sequences of nodes that start at the root. We can even assign a certainty per morpheme which is the product of the weights along its edges. If the corpus is not large enough an easy approach to improving access to data would simply be to randomize the words in the corpus and run the algorithm through it again starting with the existing $p - trie$.

Algorithm 1 Build Trie

```
Require: $0 < \lambda < 1 < \rho$
  $pointer \leftarrow root$
  for $char$ in $corpus$ do
    if $pointer == root$ then
      $p_1 = 1, p_2 = 1$
    else
      $p_1 = \lambda, p_2 = \rho$
    end if
    if $char$ in $pointer \rightarrow children$ then
      $edge_{pointer \rightarrow char} \leftarrow edge_{pointer \rightarrow char} * \rho$
    else
      $edge_{pointer \rightarrow char} \leftarrow \lambda$
      $pointer \rightarrow children$ append $char$
    end if
    $r \leftarrow$ random number(0,1)
    if $r > edge_{pointer \rightarrow char}$ then
      $pointer \leftarrow root$
    else
      $pointer \leftarrow char_{pointer}$
    end if
  end for
  return $root$
```

Below is the flow diagram to explain the functioning of the algorithm.


initialize trie
corpus
set pointer to root
get input value → in
is pointer set to root?
yes
no
set $p_1 = p_2 = 1$
set $p_1 = \lambda$ and $p_2 = \rho$
is in a child node of pointer?
yes
no
multiply transition probability by p_2
add in as child with transition probability p_1
set pointer (probabilistically) based on transition probability, p_3
$1 - p_3$
p_3
set pointer to in

5 Results

Waiting on finishing runs and generating graphs.

6 Discussion

Waiting on Results

7 References