A statistical approach of the tokenization problem in the design of Chinese input method

Jiahao Li

Abstract

Tokenization is the process of dividing Chinese sentence romanization into tokens that represent the romanizations of each Chinese character. This paper proposes a statistical algorithm based on dynamic programming to address the tokenization problem in the design of Chinese, and more specifically, Cantonese input methods. It is an algorithm with $\Theta(n^3)$ time complexity, in which n is length of input romanization sequence. In the end of the paper, the accuracy of the algorithm is evaluated based on several Chinese language corpora.

1 Introduction

Romanization of Chinese characters is the process of transforming Chinese characters into sequence of Roman alphabet letters. Usually, this is done by mapping each individual Chinese character to its Roman representation through certain rules, and then combine these individual romanizations into the romanization for the entire character sequence. For example, in Jyutping, a romanization pattern that corresponds specifically Cantonese (a dialect of Chinese), Chinese characters “你” and “好” are mapped to Roman word “nei” and “hou” respectively. And therefore, the Jyutping romanization sequence for the Chinese word “你好” is then “neihou”.

Inputing Chinese characters on a computer then follows the reverse process. The first step is to break the romanization sequence into subsequences that correspond to the original Chinese characters. For example, in the Jyutping example presented, the sequence “neihou” would be broken into subsequences “nei” and “hou.” This process is called tokenization in the design of Chinese input methods.

Although the process of romanizing Chinese characters is deterministic, the process of tokenization does not yield unique valid result. The sequence “ngaan,” for example, can both be treated as a single token or two valid tokens “ng” and “aan.” Therefore, one of the goals of tokenizing algorithms is to achieve high accuracy in tokenization.

In the following chapter, I will present a statistical algorithm of tokenization that is based on dynamic programming. Then, I will evaluate the accuracy of such algorithm with various Chinese language corpora.

2 Algorithm

The tokenization algorithm presented in this section is based on a concept of pairwise proximity, which measures the probability that two particular tokens appear together in the tokenized sequence. The probability measure of a complete tokenization sequence is then defined to be the geometric mean of all pairwise proximities of adjacent pairs of tokens in the sequence. For example, consider a case where the sequence to be tokenized is denoted as S, and a particular tokenization solution is denoted as $t = (t_1, t_2, \ldots, t_n)$. Define the function of pairwise proximity to be $P(a, b)$. Then, the probability measure of the complete tokenization sequence, p, can be calculated with the following equation:

$$p = \left(\prod_{i=1}^{n-1} P(t_i, t_{i+1}) \right)^{\frac{1}{n-1}}$$

Based on the concept of pairwise proximity, the algorithm presented in this section then attempts to find the tokenization solution with the highest probability measure.

Such notion is based on the fact that certain Chinese syllables (corresponding to individual tokens) go together more frequently than others. And by utilizing these systematic patterns, the algorithm is able to identify the tokenization solution that is most likely to occur in the Chinese language, thus achieving higher tokenization accuracy.

The pairwise proximity function, $P(a, b)$, as defined above is then calculated by running statistics on existing Chinese language corpora. Let $R(a)$ denote the romanized sequence of the Chinese character a. For each pair of tokens (a, b), $P(a, b)$ is then defined as the number of pairs of characters (x, y) with $R(x) = a$ and $R(y) = b$ that appear immediately adjacent in the corpora.

With the notion of pairwise proximity defined and its calculation explained, the algorithm then utilizes a dynamic programming model to calculate the optimal solution, that is, the tokenization solution with the largest probability measure (geometric mean of adjacent pairwise proximities).

Let T be the set of all valid romanization tokens, and let $L(t)$ denote the length of the token t. Then let $D_{i,t,p}$ denote the largest probability measure attainable in tokenizing the first i characters of S into p tokens with the last token being t, where $1 \leq i \leq n$, $t \in T$, and $1 \leq p \leq n$.

The terminating cases of this dynamic programming process is self-explanatory: for each pair of tokens (a, b), $D_{L(a)+L(b),b,2} = P(a, b)$ if the first $L(a) + L(b)$ characters of the sequence S match $a + b$. All statuses except those set in this way are then set to be unattainable.

Then, we calculate each D value in the order of increasing i and then increasing p (the order of t does not matter here because the set of tokens itself is an unordered structure). For each triple of (i, t, p), let s be the first i characters of S. We first verify if it is a valid combination, that is, whether the last $L(t)$ characters of s matches t. Then, we iterate through all tokens as the potential token that precedes t in the sequence. For each token o iterated, we check if $D_{i-L(t),o,p-1}$ is attainable. If so, we can then transit from $D_{i-L(t),o,p-1}$ to $D_{i,t,p}$ by adding t to the sequence denoted by $D_{i-L(t),o,p-1}$ according to the following equation:

$$D_{i,t,p} = \left(P(o,t) D^{p-2}_{i-L(t),o,p-1}\right)^{\frac{1}{p-1}}$$

if such is more optimal, where $D^{p-2}_{i-L(t),o,p-1}$ stands for the optimal pairwise proximity product of the subsequence from character 1 to character $i - L(t)$ that is divided into $p - 1$ tokens with the last one being o. $P(o, t)$ is the proximity product given by putting t next to the token o. Multiply them and we get a potentially optimal pairwise proximity product of the subsequence s that is divided into p tokens with the last one being t.

Continue with this process until all D values are calculated. Then we can identify the optimal pairwise proximity product by enumerating over all possible combinations of 1) last token as t and 2) number of tokens divided into as p and identify the maximal $D_{n,t,p}$.

Finally, to assemble to actual token sequence, we record for each state triple (i, t, p) the state triple from which it is transited as $C_{i,t,p}$. Then we can retrace the steps from the optimal state all the way to one of the initial terminating cases. For brevity, explanation of this process is omitted from this paper as it is a frequently employed technique in the implementation of dynamic programming.

3 Time complexity analysis

This section will present time complexity analysis of the algorithm proposed in the last section. To begin with, a sample implementation of the algorithm is provided here in pseudocode (symbols defined as in the last section) as Algorithm 1.

Notice that the retrace process is omitted in the pseudocode, for both the reasons posed in the end of last chapter and that its presence does not alter the time complexity measure of the algorithm.

Algorithm 1 Tokenize romanization sequence S

```
Require: $S$ be a valid sequence
Ensure: $D$ calculated as defined
Ensure: $O$ is the optimal probability
  set all $D$ states to be unattainable
  for all token $p$ in $T$ do
    for all token $q$ in $T$ do
      if $p + q$ matches the start of $S$ then
        if $P(p, q) > D_{L(p)+L(q),q,2}$ then
          $D_{L(p)+L(q),q,2} \leftarrow P(p, q)$
        end if
      end if
    end for
  end for
  for $1 \leq i \leq n$ do
    $s \leftarrow$ the first i characters of $S$
    for all token t in $T$ do
      if $t$ does not match the end of $s$ then
        skip this iteration
      end if
      for $3 \leq p \leq n$ do
        for all token $o$ in $T$ do
          if $s$ is shorter than $o$ then
            skip this iteration
          end if
          if $D_{i-L(t),o,k-1}$ is unattainable then
            skip this iteration
          end if
          $x \leftarrow \left(P(o, t) D^{k-2}_{i-L(t),o,k-1}\right)^{\frac{1}{k-1}}$
          if $x > D_{i,t,p}$ then
            $D_{i,t,p} = x$
          end if
          if $i = n$ and $D_{i,t,p} > O$ then
            $O \leftarrow D_{i,t,p}$
          end if
        end for
      end for
    end for
  end for
```

Also notice that the main time-consuming part of the algorithm consists of four layers of embedded loops of basic operations that are of dimensions n, $|T|$, n, and $|T|$ respectively. Thus, the time complexity of the algorithm is $\Omega(|T|^2 n^2)$, that is, $\Omega(n^2)$.

4 Simulation and evaluation

In this section I will carry out simulations of the proposed algorithm on a general purpose corpus of the Chinese languages, which consist of materials including newspaper articles, novels.

To demonstrate that the algorithm proposed can be combined with learning behaviors to yield better results for particular kinds of input work, I will also run the simulation on a single-type corpus of newspaper articles, and then compare the accuracies in these two cases.

And in both cases, the corpus used is divided into two halves that are equal in size. The first half is used to train the pairwise proximity function, that is, to calculate the P function. The algorithm is then run the other half, and the output is then compared with the correct answers to determine the accuracy of the algorithm.

In the Table 1 the simulation results in these two cases is shown.

	General	News
#	120210	54730
# Correct	111758	51741
Accuracy	93.0%	94.5%

Table 1: Simulation results for proposed algorithm

To help put the results in perspective, the simulations are repeated with a naive algorithm (one that always prefers longer tokens), and the results is noted in Table 2.

From the results we can see that in both cases, the proposed algorithm out-performs the straightforward approach, by 11.1% and 11.2% respectively. And it achieves the highest accuracy, 94.5%, on the corpus that is

	General	News
#	120210	54730
# Correct	98548	45576
Accuracy	81.9%	83.3%

Table 2: Simulation results for naive algorithm

consists mainly of newspaper articles.

Also notice that on the newspaper-only corpus the algorithm performs noticeably better than on the general-purpose corpus. This is because training the proximity function P with particular kinds of source materials helps it retrieve more intricate patterns in the language structure. This suggests that the learning behavior can be incorporated into the algorithm to enable it to adapt to the particular kinds of input tasks at hand.

5 Conclusion and further research

To conclude, this paper proposes a dynamic-programming-based algorithm that tackles the problem of tokenization in the design of Chinese (or specially, Cantonese) input method. Simulations are then carried out to illustrate the improvement in tokenization accuracy this algorithm brings about. And the results of simulations also suggest that the algorithm can be used in connection with a learning behavior to achieve adaptability for particular types of input tasks.

In the process of research, various insights worth further exploring have risen. For example, it is noticed that a majority of the errors that occur in the process of tokenization is in the form of incorrectly dividing “sing” as “si” and “ng” (or “ting” as “ti” and “ng,” etc.) Therefore, further research on the problem of tokenization should include the exploring of such frequently occurring cases of systematic errors, as it will likely yield results that can further improve the accuracy of the algorithm that is proposed in this paper.

References