An application of statistical optimization with dynamic programming to phonemic-input-to-character conversion for Chinese.

Richard Sproat

AT&T Bell Laboratories

Abstract

This paper describes a method for the conversion of Chinese text written in phonemic transliteration into Chinese character text. The method uses character bigram probabilities estimated from a large corpus of Chinese text to compute the most likely path through the lattice of possible character transliterations. Dynamic programming is used to prune the search. Thus the method is statistical, like previous work by Lin and Tsai (1987), but it is algorithmically simpler than their method. Performance scores are given which demonstrate that the proposed method produces good results.

An application of statistical optimization with dynamic programming to phonemic-input-to-character conversion for Chinese.

Richard Sproat

AT&T Bell Laboratories

1. Introduction: the problems of Chinese orthography.

It is a familiar point that inputing Chinese text is non-trivial because of the thousands of characters used in Chinese orthography; this makes the development of word-processing technology for Chinese much more difficult than for languages like English. While there have been a number of different classes of solutions proposed, there has been much interest in what is often referred to as *phonetic input*, more correctly *phonemic input*. Under these schemes one types Chinese text using one of several different transliteration schemes, usually either the Mandarin Phonetic system in Taiwan, or Hanyu Pinyin on the Mainland or abroad. The text is then converted to characters. Because the number of characters in the transliteration scheme is small, a normal keyboard can be used, and the user need only possess normal typing skills. Of course, to use such a system, one is required to know (one of) the transliteration schemes assumed, and be familiar with the particular Chinese language — usually Standard Mandarin — which the system assumes, but this is often argued to be a relatively small drawback since most Chinese speakers are familiar with one or another transliteration scheme and most literate Chinese are familiar with Standard Mandarin (Becker 1984b, Lin and Tsai 1987). Because phonemic input does not require special

keyboards or the mastery of special coding schemes, it thus confers a number of advantages.

The major problem with phonemic input is of course ambiguity, since the number of syllable types in Chinese is significantly fewer than the number of characters. The syllable-to-character relation is therefore one-to-many, but since it is very skewed, highly ambiguous syllables like *shi4* are not unusual (I shall use the Hanyu Pinyin transliteration system in this paper, with numerals representing the tone):

是 事 市 式 世 示 士 視 識 試 適 室
勢 釋 氏 飾 侍 逝 誓 仕 嗜 恃 拭 噬
軾 弒 筮 柿 奭 舐 茬 伕 偍 侍

However, although syllables in isolation are quite ambiguous, most text consists of strings of many syllables, and one can significantly reduce ambiguity by taking the context into account. The most popular method is 'word-unit' based input, discussed in Becker (1984a,b). In all such systems some post-editing of the text is usually necessary, but the post-editing task is significantly reduced.

The present paper reports on a novel approach to the phoneme-to-character problem. The algorithm reported uses estimates of the conditional probability of a character given the previous character to find the optimal path through the lattice of possible characters. Dynamic programming is used to prune the search. The statistics are derived from a large corpus of Chinese text. In the next section we describe the algorithm and in the subsequent section we report on the implementation and the performance, and compare the latter with previous similar work.

2. *The statistical method.*

Church (1988, and see also Derose 1988) has proposed a statistical method for tagging English words with parts of speech. Conceptually, the problem can be laid out as follows: for any word in English text, there are a number of ways of tagging it with a part of speech label. So, *table* can be tagged as a noun in 'The suit is on the table' but as a verb in 'Let us table the suggestion.' Since each word may be many-ways ambiguous in this regard, a sentence of text may generally have a richly branching lattice of part of speech label possibilities associated with it. Church gives the following example (where 'UH' means 'interjection'):

I	see	a	bird
PRON	V	ART	N
PRON	V	P	N
PRON	UH	ART	N
PRON	UH	P	N
PROPNOUN	V	ART	N
PROPNOUN	V	P	N
PROPNOUN	UH	ART	N
PROPNOUN	UH	P	N

The general problem to be solved here is to find the most probable path through the lattice of parts given the lattice of words, or in other words, for a sentence of length n:

$$argmax \quad p(part_0 \cdots part_{n-} | word_0 \cdots word_{n-1})$$

Church approximates this conditional probability with a second order model as follows:

$$argmax \prod_{i=2}^{n-1} \frac{p(word_i|part_i) * p(part_i|part_{i-1} part_{i-2})}{p(word_i)}$$

Conceptually one enumerates all of the possible paths through the lattice and then picks the best path according to the above metric. However, we observe from the above formula that we are considering at most a trigram of part of speech labels, and that a dynamic programming solution is possible. So, at any point while traversing the lattice, for each triple of parts of speech $part_{i-2}part_{i-1}part_i$ we only need to remember the *best* path ending in that triple, since only the best such path can compete further down the lattice. This effects a significant reduction in the search space.

The relevance of this to the issue addressed in the present paper can be seen when we observe that the problem of deciding the best path through a lattice of part of speech labels is identical in relevant respects to the problem of deciding the best path through a lattice of Chinese characters generated from a phonemic input. So, consider the following example:

25 B

tai2	wan1	you3	tai2	feng1
(台)	(灣)	(有)	台	(風)
臺	彎	友	臺	豐
抬	蜿	勘	抬	封
颱	豌	酉	(颱)	峰
檯	剜	莠	檯	鋒
苔		牖	苔	蜂
抬			抬	瘋
檯			檯	楓
郃			郃	丰

25A

The circled characters represent the most likely path through the lattice, that is, the sentence 'Taiwan has typhoons.'

The problem is approached in a way entirely analogous to Church, except that a first order model is used. (Until recently we did not have enough data to train a second order model.) Replacing 'part of speech' with 'character' (c) and 'word' with 'syllable' (σ) we arrive at the following expression:

$$argmax \prod_{i=1}^{n-1} \frac{p(\sigma_i|c_i)*p(c_i|c_{i-1})}{p(\sigma_i)}$$

We do not currently have a corpus of Chinese text with phonemic transcription so it is not currently possible to estimate $p(\sigma_i|c_i)$, i.e. the probability of a syllable σ given c, where σ is a possible pronunciation for c. However, since the majority of Chinese characters are unambiguous in their pronunciation (unlike the situation in Japanese) we can assume that this term is 1 in the general case. Furthermore, the denominator term

does not effect the maximization since it is simply the probability of σ_i, which merely serves as a constant multiplier for probabilities of paths ending in that syllable. We can therefore simplify the above equation to:

$$argmax \prod_{i=1}^{n-1} p(c_i | c_{i-1})$$

In the current method, as in Church's method, dynamic programming is used to prune the search.

3. Performance issues.

3.1 Implementation and performance.

The statistics used by the version of the system which has been evaluated to date are derived from a corpus of 2.6 million characters of Chinese newspaper text (kindly provided by United Informatics, Inc., Taiwan). The current corpus contains 4846 distinct characters for which we have Pinyin transliterations. The bigram probabilities are estimated by dividing the frequency of occurrence of a sequence of characters by the size of the corpus. The unigram probabilities are estimated in a similar manner. (In practice the corpus size terms are omitted since they do not affect the maximization.) The algorithm has been implemented in C and runs on a Sun 3 at a rate of approximately 25 characters per second.

We turn now to a discussion of the performance of the method. Seven short samples of text of differing length were chosen, representative of various writing styles ranging from very classical to colloquial:

i) Ad [classical]

ii) Report [classical]

iii) Newspaper social commentary taken from the training corpus [semi-classical]

iv) Essay [semi-colloquial]

v) Narrative [semi-colloquial]

vi) Short story [colloquial]

vii) Exposition [colloquial]

The appendix contains the Pinyin along with the character renditions for the first text. Incorrect characters are circled.

The performance is given in the following table in terms of percentage correct by character (*hit rate*) for each of the styles; the hit rate from the algorithm is given in column three and should be compared with the hit rate achieved by merely picking the most common character given the pronunciation, which is given in the second column:

Style	Lexical Probabilities Only	Current Algorithm
i [class.]	76%	93%
ii [class.]	73%	90%
iii [semi-class.]	76%	98%
iv [semi-coll.]	69%	73%
v [semi-coll.]	72%	86%
vi [coll.]	71%	89%
vii [coll.]	71%	92%
Total	73%	90%

A trend evident in these data is that there is some dependency upon style: our current training corpus is heavily classical in style since it is mostly derived from newspapers.

As a consequence, texts (i-iii) which are classical in style are better rendered than the more colloquial texts, with the exception of (vii). One can expect that this style dependency would become less marked if the training corpus were expanded to include other styles.

3.2 Related research.

The only previous statistically based work on the phoneme-to-character problem of which we are aware is reported in Lin and Tsai (1987) and with subsequent modifications in Fan and Tsai (1988). Lin and Tsai employ the relaxation technique to obtain the optimal path through the lattice of possible characters, making use of the lexical probabilities of the characters given the syllables and the transition probabilities of adjacent characters and adjacent syllables. Their model was trained on a corpus of 196,573 characters of newspaper text written for elementary school students, and tested on similar materials. This text, unlike our corpus, contains phonemic transcriptions along with the character text. Thus Lin and Tsai are able to measure $p(\sigma_i|c_i)$, which is important in the few cases where a character has multiple pronunciations. In evaluating their results it is important to consider that elementary school texts represent a much simpler style than the texts with which we have been dealing, and are expected to be biased towards much more common words (and characters) than adult texts. One consequence of this is that they report an average hit rate of 77.6% by merely choosing the lexically most probable character, given the syllable, which is higher than our average rate of 73%. Their overall average hit rate is 95.4%, and the algorithm runs at a speed of about 2 characters per second. The speed appears to be slower than the one reported here, though this is at least in part due to a difference in

hardware. As far as their hit rate is concerned, it is important again to realize that their training and test corpus was restricted to fairly elementary texts; one may expect not to get such a high score if the model were trained and tested on more mature Chinese texts.

4. Summary

The system reported here is clearly useful for reducing the amount of post-editing necessary for Chinese text entered with phonemic transliteration. The system is still under development and we have recently increased the size of the training corpus over the 2.6 million characters reported on above and are currently evaluating the behavior of the system under those conditions.

References.

Becker, J. 1984a. Multi-lingual word processing. *Scientific American* 251(1), 96-107.

Becker, J. 1984b. Typing Chinese, Japanese and Korean. *Computer* 18(1), 27-34.

Church, K. 1988. A stochastic parts program and noun phrase parser for unrestricted text. *Second Conference on Applied Natural Language Processing*, 136-143.

DeFrancis, J. 1984. *The Chinese Language*. Honolulu: University of Hawaii Press.

DeRose, S. 1988. Grammatical category disambiguation by statistical optimization. *Computational Linguistics* 14(1), 31-39.

Fan, C.-K. and Tsai, W.-H. 1988. Disambiguation of phonetic Chinese input by relaxation-based word identification. *Proceedings of ROCLING I*, 145-160.

Lin, M.-Y. and Tsai, W.-H. 1987. Removing the ambiguity of phonetic Chinese input by the relaxation technique. *Computer Processing of Chinese and Oriental Languages* 3(1), 1-24.

Appendix

Test text (i): circled characters are errors.

1. Pinyin:

pin3 zhi2 gao1 chao1 de0 guang3 gao4 yan2 liao4 shi4 you2 duo1 nian2 de0 jing1 yan4 . zai4 xian4 dai4 de0 ji4 shu4 she4 bei4 xia4 , cai3 yong4 gao1 ji2 yUan2 liao4 pei4 zhi4 er2 cheng2 . ge4 zhong3 se4 liao4 bu4 dan4 ke3 xiang1 hu4 jiao3 hun4 shi3 yong4 . qie3 se4 shang4 jia1 se4 , xiang1 die2 shi3 yong4 shi2 , yi4 wu2 tou4 lou4 xian3 chu1 di3 se4 zhi1 lU4 . ji2 shi3 hei1 se4 shang4 tu2 bai2 se4 yan2 liao4 , ye3 neng2 wan2 qUan2 yan3 gai4 .

2. Character rendition:

品 質 高 超 的 廣 告 顏 料 是 由 多 年 的 經 驗 。
在 現 代 的 (計) (數) 設 備 下 、
採 用 高 級 原 料 配 (置) 而 成 。
各 種 色 料 不 但 可 相 互 (皎) 混 使 用 。
且 色 上 加 色 、 相 疊 使 用 (食) 、
亦 無 透 露 顯 出 底 色 之 (律) 。
即 使 黑 色 上 塗 白 色 顏 料 、
也 能 完 全 掩 蓋 。