CHARACTER PREDICTION FOR ON-LINE HANDWRITING RECOGNITION

Homayoon S.M. Beigi
T.J. Watson Research Center
International Business Machines
P.O. Box 704
Yorktown Heights, New York 10598

Abstract

Character prediction could be used in reducing the number of errors in applications such as handwriting recognition. Commonly, in handwriting recognition, once a character is written, its digitized $< x, y >$ coordinates are fed into a shape matcher which will match some features with a set of given prototypes and will return probabilities of each prototype being the intended writing. This is independent of the history of the text being written. One way to increase the accuracy of such recognizers is to take advantage of the context of the text being written. It is a well-known fact that as the best handwriting recognition engines, we humans use a lot of context in reading handwritten text and that our recognition accuracy will dramatically decrease once we are shown individual handwritten chracters without any context. This paper will present a study of character prediction and its potentials for increasing recognition accuracy. The studies in this paper will also provide a character predictor based on a character-level N-gram with an optimal optimal length of context for application to handwriting recognition.

1 Introduction

The notion of using a handwriting interface to computers has been of great interest to many researchers for a number of years. With today's technology, a low-priced LCD digitizer tablet for capturing handwriting is conceivable with a size no bigger than a notebook. These devices capture the $< x, y >$ coordinates of the writer's pen on the surface of the digitizer tablet. Also, with the introduction of new pen-based operating systems such as the Penpoint^{TM} and Pen-Windows^{TM} operating systems handwriting recognition applications have become very practical. For a survey of different handwriting recognition methods see [1].

Having some previous context will enable us to predict the next character with some probability associated with it. For this purpose, character N-grams could be used to predict the next character which is going to be written at the writing tablet, based on what has been previously written. For example, by having the context "th" it is very probable that an "e" will follow, due to the high frequency of words such as "the, then, them, these, etc." However, the probability of "z" following "th" is very low or close to zero, in the English language.

2 Data-Base Generation

To establish a data-base such that these predictions could be made available to a handwriting recognizer, a list of words and their frequency in the language is needed. The first problem is to find out how long a list of words is practically needed. Secondly, how large an N should be used. As N increases, the perplexity of possible characters based on the $N-1$ context decreases. To try to answer the above questions a corpus of 320,000,000 words in the form of on-line office correspondence was used. Close to 290,000 individual words were extracted from this corpus with their frequencies noted. Table 1 shows the form of this word list.

Word	Frequency
the	18,155,290
,	15,017,520
⋮	⋮

Table 1: Word Frequency Information

The first thing that was done was to remove all the words containing punctuation or special characters. This reduced the number of distinct words to about 273,000. First, the 40,000 most frequent words were used for extracting N-grams. For reasons which will be made apparent later, $N = 4$ was used. In producing N-grams, a window of length N is shifted from the left to the end of the word and the probabilities of each character following previous characters is noted. The set of characters which are supported are $a - z$ plus space denoted here as #. The N-gram data-base is made case- insensitive. Therefore, there are 27 characters to be considered, including the space. For example, when a word such as *these* is being written, $p(t|\#)$, $p(h|\#t)$, $p(e|\#th)$, $p(s|the)$, $p(e|hes)$, and $p(\#|ese)$ could be provided by such a database. Table 2 presents average fanout information about 4-grams obtained from the 40,000 most frequent words. Table 3 presents the same information about 4-grams obtained from all 273,000 words.

Context Length	Perplexity
1	21.38
2	10.71
3	4.80

Max. Fanout:	26
No. of fanouts > 13:	281
No. of tetra-grams:	28,590

Table 2: Average Fanout, N=4, 40,000 Words

Context Length	Perplexity
1	26.15
2	14.44
3	7.13

Max. Fanout:	26
No. of fanouts > 13:	1577
No. of tetra-grams:	70,043

Table 3: Average Fanout, N=4, 273,000 Words

The average fanout of the large vocabulary 4-grams with 3 letter context is over 7 whereas this value is 4.8 for the small vocabulary. Table 4 shows fanout information for the large vobulary system when all 4-grams with final letter conditional probabilities lower than 0.1% were deleted.

Context Length	Perplexity
1	25.15
2	14.17
3	4.99

Max. Fanout:	20
No. of fanouts > 13:	207
No. of tetra-grams:	46,249

Table 4: Average Filtered Fanout, N=4, 273,000 Words

As could be seen, a perplexity very close to that obtained from the small vocabulary system is obtained. However, in this data, most probable 4-grams in the language are presented as opposed to 4-grams from the most frequent words in the language. The new data-base makes a much more robust data-base. These results led to the optimal solution of N=4 which means that a character is predicted based on a three letter context.

N-grams with $N = 3$ and $N = 5$ were also extracted. $N = 3$ will produce an average fanout perplexity of about 13 which is very large and is not sensible. $N = 5$ will produce a perplexity of 3 but at the cost of generating 2.5 times the number of N-grams in N=4. Tables 5 and 6 present the fanout information for $N = 5$ extracted from 273,000 words before and after removing low probability N-grams. For practical purposes, the gain in the reduction of average number of hypotheses from 5 to 3 is not enough to use 2.5 times the N-grams which will translate into a database which is many times larger than that for $N = 4$. As the database becomes bigger, the cost of looking up an N-gram will become higher. Therefore, $N = 4$ seems to be the optimal choice.

Context Length	Perplexity
1	24.81
2	15.00
3	7.60
4	3.30

Max. Fanout:	25
No. of fanouts > 13:	856
No. of tetra-grams:	194,523

Table 5: Average Fanout, N=5, 273,000 Words

Context Length	Perplexity
1	24.05
2	14.60
3	5.30
4	3.00

Max. Fanout:	18
No. of fanouts > 13:	46
No. of tetra-grams:	117,925

Table 6: Average Filtered Fanout, N=5, 273,000 Words

4-grams were generated and with their corresponding probabilities were stored in a TRIE-like database. This database contains the probabilities associated with each letter in the tetra-gram based on its previous letters which means that there are 4 probabilities stored for each tetragram. The database has a probability granularity of 0.1% and is compressed to fit in less than 128k Bytes. The access time is negligible which makes it practical for usage in on-line handwriting recognition.

3 Recognition Probabilities

Figure 1 shows the accuracy of text prediction used with a text of 180 words. This graph shows the accuracy of prediction of this 4-gram using 1, 2, 3, or a combination of context length. The combination was done in a natural sense from the available data depending on the position within the word. For example, given a point on the graph with 92% on the y axis and 5 on the x axis means that there is an accuracy of 92% associated with characters being within the top five answers returned from the predictor. This graph is a very good measure of the top performance of this predictor.

Figure 1: Prediction accuracy of the pure tetra-gram model

The probabilities given by the above predictor are probabilities within a space of alphabets only, S_1. However, most recognizers are designed to handle more than just alphabets such as the space of numbers, special characters ($, %, etc.), and punctuations, S_2.

Figure 2: Charcater Space

Assume M_1 items in space S_1 and M_2 items in space S_2. The problem is to transform probabilities given in space S1 or S2 to probabilities in the space S in the form: $p(c_k|S) \;\; 0 \leq k < M$.

To achieve this task, let us write the probability,

$$p(c_i|S_1) = \frac{p(c_i \cap S_1)}{p(S_1|S)} \tag{1}$$

$$p(c_i|S_1) = \frac{p(c_i|S_1)}{p(S_1|S)} \tag{2}$$

Therefore,

$$\left.\begin{array}{l} p(c_i|S) = p(c_i|S_1)p(S_1|S) \\ p(c_j|S) = p(c_j|S_2)p(S_2|S) \end{array}\right\} p(c_k|S) \tag{3}$$

where i is the index in S, covering the subspace S_1 and j is the index in S, covering the subspace S_2. Therefore, the handwriting application using the recognizer, could provide the ratio of alphabet vs. numbers and hence provide $p(S_1|S)$ and $p(S_2|S)$. Using these probabilities and probabilities of characters given by the predictor, using equations 3, a useful probability value could be evaluated to be used by the handwriting recognizer.

4 Conclusion

The objective of this paper was to find an optimum number of characters to be used as the context of prediction. Optimality is measured by concurrently minimizing the average fanout (perplexity) and the amount of storage needed and maximizing the access speed such that on-line handwriting recognition is made possible. Reducing the fanout will reduce the number of prototypes that the shape matcher should match against. This will reduce the number of confusions and will also reduce the search space which will result in much better accuracies. In fact reducing the number of prototypes to about 1/3 has shown to reduce the error by about 70%. [2]

References

[1] C.C. Tappert, C.Y. Suen and T. Wakahara, "The State of the Art in On-Line Handwriting Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, N0. 8, pp. 787-808, August 1990.

[2] T. Fujisaki, H.S.M. Beigi, C.C. Tappert, M. Ukelson and C.G.Wolf, "Online Recognitoin of Unconstrained Handprinting: A Stroke-based System and Its Evaluation," _______, S. Impedovo, Editor, ELSEVIER: Italy, 1992.

[3] Homayoon S.M. Beigi, T. Fujisaki, "A Character Level Predictive Language Model and its Application to Hanwriting Recognition," Canadian Conference on Electrical and Computer Engineering, IEEE: Toronto, Canada, September 1992.