Reprint: Int. Workshop on Modern Modes of Man-Machine-Communication, Maribor, Slovenia, pp. 7/1-7/10, Jun. 1994

SYMBOL RECOGNITION IN HANDWRITTEN MATHEMATICAL FORMULAS

Hans-Jürgen Winkler

ABSTRACT:

In this paper an efficient on-line recognition system for handwritten mathematical formulas is proposed. After formula preprocessing a symbol hypotheses net is generated by extracting different features for stroke unity. Each element of the symbol hypotheses net represents a possible symbol. The data of each symbol hypotheses are preprocessed and an image is calculated. Features for symbol classification are extracted by moving a horizontal and vertical window along the image. The symbol recognition is based on a first-order, left-to-right Hidden Markov Model. The final classification of the handwritten input is done by calculating the best fitting path through the symbol hypotheses net under regard of the stroke group probabilities and the probabilities obtained by the HMM symbol recognizer.

1 INTRODUCTION

Automatic recognition of handwritten mathematical numerals and characters is a subject of research due to its potential for intelligent human-machine-interfaces. Especially, if the input to a computer are mathematical formulas with two-dimensional information (in comparison to a line of text, where characters are placed subsequently on the same line), the use of a keyboard is uncomfortable.

Handwriting recognition can be carried out on-line and off-line [1]. On-line recognition requires a stylus and an electronic tablet connected to a computer, which captures the data while it is written. Off-line recognition means that a person writes on a paper and the handwriting is captured by a scanner after the writing is completed. The advantage of on-line devices is that they capture the temporal information of the writing like the number of strokes, the order of the strokes and the direction of the writing of each stroke. A stroke, in this connection, is the writing from pen down to pen up.

Automatic recognition of handwritten mathematical expressions implies both symbol recognition and structure interpretation. This paper describes the symbol recognition process. Knowledge resulting from this process will drive the structure analysis [2][3][4].

2 SYSTEM OVERVIEW

The recognition process is divided into the following stages:

1. formula preprocessing;
2. symbol hypotheses net;
3. symbol preprocessing and feature extraction;
4. symbol recognition;
5. formula recognition.

A block diagram of this recognition system is illustrated in fig. 1.

The data input consists of the on-line captured pen positions of a handwritten mathematical formula sampled on an electronic tablet over the time. Fig. 2 shows an example for a handwritten

Reprint: Int. Workshop on Modern Modes of Man-Machine-Communication, Maribor, Slovenia, pp. 7/1-7/10, Jun. 1994

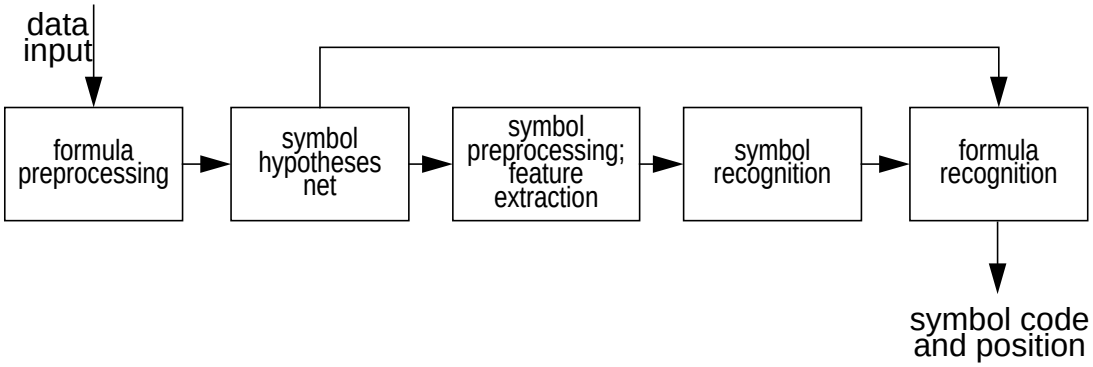


Figure 1: System Overview

mathematical expression and the corresponding input data subdivided into *M* temporal successive strokes. Each stroke consists of a sequence of (x,y)-coordinates of the pen positions.

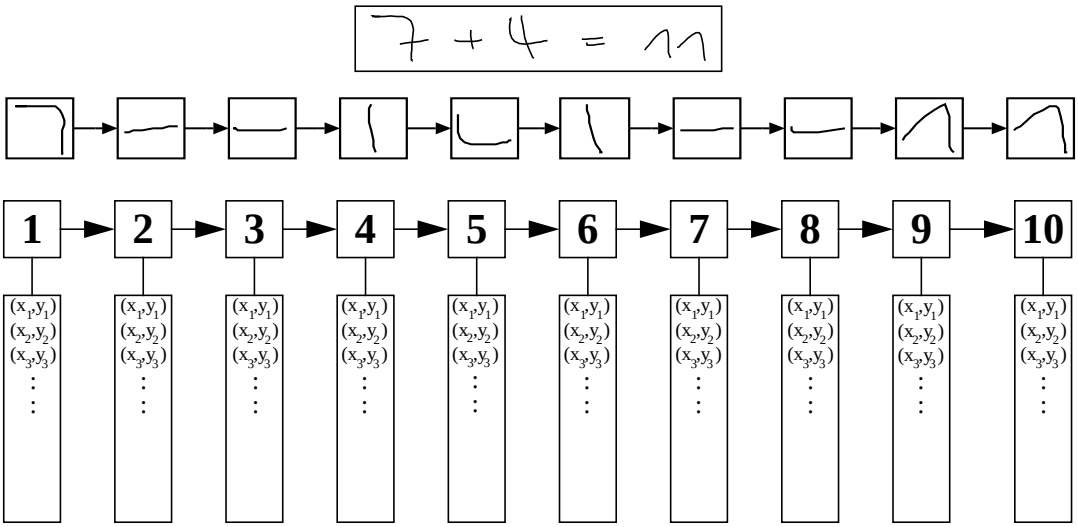


Figure 2: Handwritten input, stroke sequence and corresponding data representation

3 FORMULA PREPROCESSING

In the same manner as filters are used for speech preprocessing before attempting recognition, invariant information must be extracted from the handwritten input while discarding the vast majority of redundant variations.

Normalisation is of prime concern in preprocessing handwritten input. In this stage the area of the surrounding rectangles of each stroke from the data input is calculated. The median of the areas is the scale used for size normalisation.

After normalising the size, the slant is estimated and corrected. Therefore, the near-vertical parts of each stroke are detected. These parts are averaged with regard to there height, and a shear carried out to remove the slant. Fig. 3 shows a slant-corrected mathematical expression.

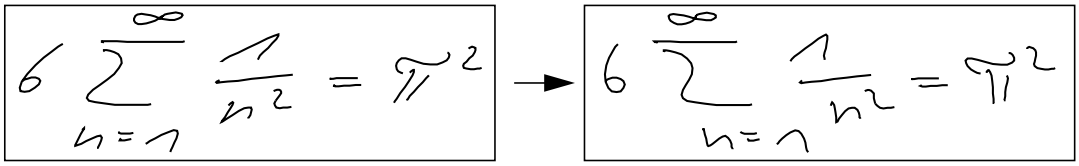


Figure 3: Mathematical formula before and after slant correction

Reprint: Int. Workshop on Modern Modes of Man-Machine-Communication, Maribor, Slovenia, pp. 7/1-7/10, Jun. 1994

4 SYMBOL HYPOTHESES NET

4.1 Conditions for symbol segmentation

The next step is to decide, which strokes could belong to one symbol. To achieve reasonable results, the writer has to fulfil a few prerequisites:

- the handwritten symbols must be within the given alphabet of currently 82 different symbols shown in fig. 4. This alphabet includes upper and lower case letters as well as digits, mathematical operators and other special symbols. There are no conditions relative to the style of writing these symbols.
- each symbol consists of four strokes at the outside.
- before writing a new symbol, the writing of the actual symbol has to be finished. This condition is normally fulfilled in unrestricted handwritten formulas.

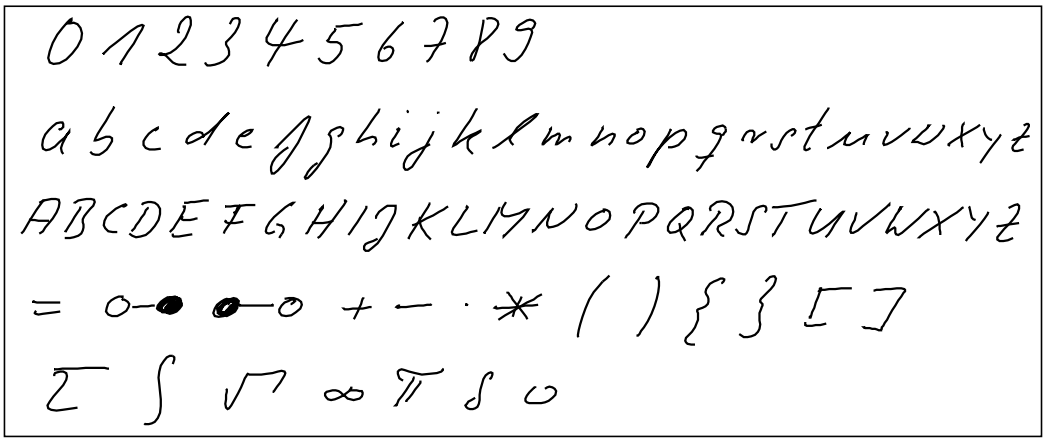


Figure 4: Given alphabet with 82 different symbols, written by writer "whfa"

The result of this prerequisites is that four temporal successive strokes at the outside could belong to one symbol.

4.2 Features for symbol segmentation

If the handwritten formula consists of M strokes, $4M-6$ different symbols can be generated by collecting one or more strokes.

To shrink this number, different features [5] for stroke unity are calculated between stroke m and stroke $m+g;\ g = 1, 2, 3$:

- the minimum distance.
- the overlapping of the surrounding rectangles.
- the distance between the starting points.
- the backward movement.
- the strokes are classified into one out of three categories (primitive, standard, complex) by calculating
  - the overall angle alteration of the stroke and
  - the minimum of the two eigenvalues (the variance vertical to the main axis of the stroke) calculated by the pen positions of the stroke.

By applying these categories to the given alphabet (fig. 4), the following results are yield:

- The more strokes belong to the same symbol, the simpler are these strokes.
- Only certain combination of these categories are possible (by restricting the alphabet).

After combining these features, a probability

$$P(m, g)\ ;\ g\ =\ 1, 2, 3;\quad m\ =\ 1, 2, \ldots, M - g; \tag{1}$$

is calculated that stroke *m* and the next *g* strokes belong to the same symbol. The probability that a symbol consist of only one stroke (the stroke *m*) is calculated by

$$P(m, 0)\ =\ 1 - \sum_{g\,=\,1}^{3} P(m, g)\ . \tag{2}$$

4.3 Generating the symbol hypotheses net

Defining an upper and a lower threshold P_U and P_L, a symbol hypotheses net is generated by observing:

- $P(m, g) > P_U$: Only this hypotheses (stroke m to m+g belong to a symbol) is represented.
- $P(m, g) < P_L$: This stroke combination is not possible and therefore not represented in the symbol hypotheses net.

By applying these threshold to $P(m, g)$, a normalisation may be necessary so that

$$\sum_{g\,=\,0}^{3} P(m, g)\ =\ 1\,. \tag{3}$$

Fig. 5 shows for the handwritten mathematical expression given in fig. 3 the subdividing into strokes and the symbol hypotheses net. Each element of the net represents a possible stroke group and therefore a possible symbol.

The handwritten input shown in fig. 5 consists of 18 strokes, therefore 66 different stroke groups can be generated. By calculating the stroke group probability this number is reduced to 21 possible symbols.

Next to each ramification in the symbol hypotheses net the probability is given.

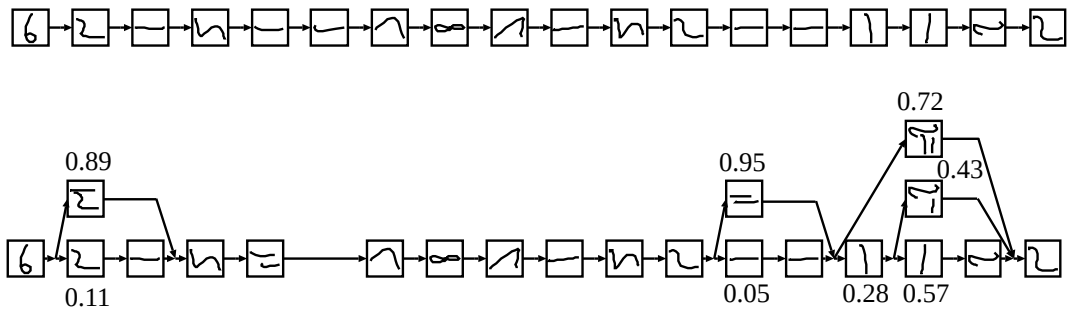


Figure 5: Strokes and symbol hypotheses net to the handwritten input given in fig. 3

Reprint: Int. Workshop on Modern Modes of Man-Machine-Communication, Maribor, Slovenia, pp. 7/1-7/10, Jun. 1994

5 SYMBOL PREPROCESSING AND FEATURE EXTRACTION

The formula preprocessing is necessary for generating the symbol hypotheses net. In this stage the symbol hypotheses net is generated, global features like size, position of each symbol are stored. Now each element of the hypotheses net has to be normalized to generate reliable features for symbol recognition.

Firstly the slant of the symbol is estimated. The average slant of the formula was already corrected, but the slant of each symbol may vary from the average slant (fig. 3) so another correction may be necessary.

The height of the symbol has to be normalized because the size does not only depend of the symbol itself but also from the context within the mathematical formula. Two thresholds y_{10} and y_{90} are calculated from the histogram of the y-coordinates of the stroke data, but no normalization is carried out. The threshold y_{10} describes the value where 10% of the *y*-coordinates are below, the threshold y_{90} is the value where 10% of the data are above.

For feature extraction an image is calculated by interpolating the on-line sampled data. A vertical feature selection window is moved along the symbol from left to right [6]. The horizontal size of the window is calculated from the width of the symbol in relation to the difference of the thresholds y_{10} and y_{90}. Each window is divided into seven small subwindows (fig. 6). By adapting the horizontal borders of the subwindows height normalization is done. The upper border of the lowest subwindow is set to the threshold y_{10}, the lower border of the top subwindow to the threshold y_{90}. The borders of the intermediate subwindows are calculated equidistant between these thresholds. The length of the strokes within each subwindow in relation to the overall stroke length of the symbol is calculated and results in a seven-dimensional feature vector for each window.

Hence, the on-line sampled handwriting data are transformed into a sequence $\{x\}$ of vectors.

In the same manner as the feature vectors $\{x\}$ are calculated, a second set of feature vectors $\{y\}$ is calculated. Before transforming the stroke data into an image, the thresholds x_{10} and x_{90} of the symbol are calculated, where 10% of the *x*-coordinates are on the left respectively on the right side of these thresholds. After image calculation a horizontal feature selection window is moved along the symbol from the top to the bottom (fig. 6), the vertical borders of the seven subwindows are defined by the thresholds x_{10} and x_{90} and therefore adapted to the width of the symbol. The vertical size of the window is calculated by the height of the symbol in relation to the difference of the thresholds x_{10} and x_{90}.

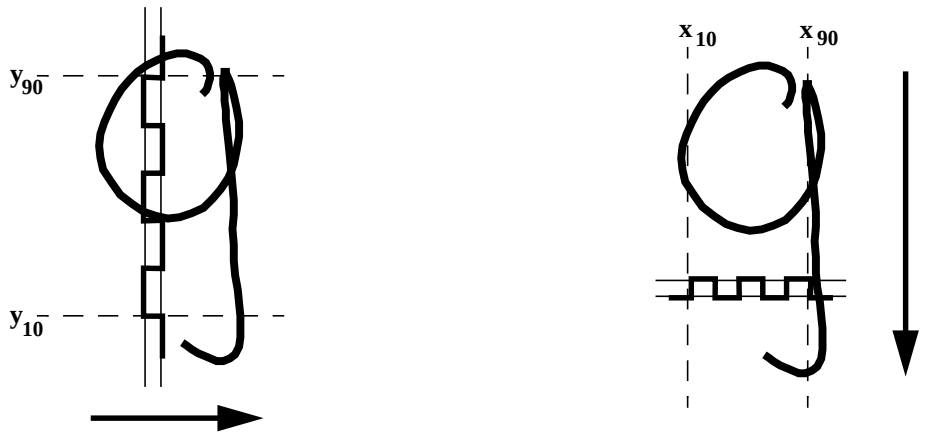


Figure 6: Feature extraction: vertical and horizontal window moved along the handwritten number "9"

Reprint: Int. Workshop on Modern Modes of Man-Machine-Communication, Maribor, Slovenia, pp. 7/1-7/10, Jun. 1994

6 SYMBOL RECOGNITION

For each sequence $\{x\}$ and $\{y\}$ of feature vectors a separate recognition system based on Hidden Markov Models (HMMs) is used. HMMs are widely and successfully used for automatic speech recognition [7]. Based on this success, there are many on-going researches to recognize printed text or handwritten script [8] by this approach.

6.1 Topology of Hidden Markov Models

A HMM is a doubly stochastic process with an underlying stochastic process that is not observable. HMM recognition requires an input sequence of so-called observations $\{O\}$. According to the notation in [9], in general a N-state HMM λ is described by two probability matrices $[A]$ and $[B]$ and a probability vector Π:

- a state transition matrix $A = [a_{ij}]$, where $[a_{ij}]$ denotes the transition probability from state i to state j, $1 \le i, j \le N$;

- the observation probability distribution matrix $B = [b_{it}]$, where $[b_{it}]$ denotes the probability to observe O_t, $1 \le t \le T$ in state i; and

- an initial state distribution vector Π.

A HMM is completely specified by $\lambda = (A, B, \Pi)$.

In this system a so-called semi-continuous, first-order left-to-right HMM [10] is used.

Semi-continuous means that a set of K discrete codewords c_k and associated covariance matrix $[C_k]$ obtained by soft-decision vector quantisation of the overall feature space R is generated [11]. The observation probability for a given feature vector x_t at state i is given by:

$$P(x_t|i) = \sum_{k=1}^{K} P(x_t|c_k)\, P(c_k|i) \; . \tag{4}$$

The probability $P(x_t|c_k)$ is calculated by the Mahalanobis distance.

First-order left-to right HMM means that transitions from state i to state j are forbidden if $j < i$ or $j > i + 2$. Furthermore, the initial state distribution vector Π is fixed so the models always starts in the first state.

6.2 Training and Classification by HMMs

The different HMMs λ_v (one for each symbol) are initialized by default values.

The training problem is to find the most probable parameters λ_v for a given training set to maximize $P(O|\lambda_v)$.

The recognition problem is to find the best fitting model λ_v out of V models for the observation set O. Classifying the symbol as class v^*, the classification is done by calculating

$$v^* = \text{argmax}\,[P(O|\lambda_v)\,;\, v = 1,2,\ldots,V] \; . \tag{5}$$

Possible classification hypotheses are obtained by taking the most fitting models.

There are two different training and recognition algorithms, the Baum-Welch algorithm and the Viterbi algorithm. Both algorithms are iterative procedures.

During training the Baum-Welch algorithm increases the overall probability for the training set by reestimating λ. During recognition the Baum-Welch algorithm calculates the total probability by summation the probability of each possible path through the model for an observed (unknown) sequence. The Viterbi algorithm, on the other hand, maximizes during training the best fitting path through the model by reestimating λ for the given training set. During recognition the total probability of the most likely path for an observed (unknown) sequence is calculated for each model.

In this system the Viterbi algorithm is used because this algorithm needs less computing performance and the recognition results are almost identical compared to the Baum-Welch algorithm.

6.3 Symbol recognition results

The training vocabulary for the symbol recognizer are single sampled symbols out of the alphabet given in fig. 4. Each writer (currently three) contributed 4100 symbols, thus each of the 82 symbol was written 50 times.

For quality verification of the symbol preprocessing and feature extraction algorithms writer-dependent recognition experiments were conducted by choosing 10 out of the 50 versions for recognition, the remaining 40 versions for training of the HMMs. This was done five times for each writer, thus each version belongs exactly one time to the recognition data pool and four times to the training data pool.

The results of the writer dependent recognition experiment are summarized in tab. 1. The first column shows the writer, the second column displays the recognition results by classifying the sequence $\{x\}$ of vectors generated by the vertical feature selection window. The results given in the third column are based on the sequence $\{y\}$ of vectors obtained by the horizontal feature selection window.

Writer	Classification by $\{x\}$	Classification by $\{y\}$
hmfa	91.6 %	90.6 %
kmfa	90.1 %	89.5 %
whfa	92.7 %	91.1 %

Table 1: Recognition rates for each writer by classifying the sequences $\{x\}$ or $\{y\}$

Tab. 1 shows that the classification of the features $\{x\}$ extracted by the vertical window deliver better recognition results.

Analysing the recognition results, it is realized that the recognizer has almost no way to distinguish between ambiguous symbols such as ’s’ and ’S’, ’x’ and ’X’ or ’0’ and ’O’. This is caused by the size normalization during feature extraction (to discard contextual information). Even a human would have problems to distinguish between these symbols. This fact is exemplary shown in tab. 2 for writer "whfa" whose alphabet is given in fig. 4. The task to distinguish between these symbols could only be solved by contextual knowledge.

Symbol	Classification by $\{x\}$ as symbol (frequency [%])	Classification by $\{y\}$ as symbol (frequency [%])
0	0 (58); O(36)	0 (58); O(34)
O	O(72); 0 (20)	O(60); 0 (22)
c	c (78); C (16)	c (72); C (22)
C	C (78); c (20)	C (80); c (20)
s	s (86); S (12)	s (84); S (16)
S	S (62); s (30)	S (72); s (22)

Table 2: Classification results between almost not distinguishable symbols (writer "whfa")

Reprint: Int. Workshop on Modern Modes of Man-Machine-Communication, Maribor, Slovenia, pp. 7/1-7/10, Jun. 1994

Symbol	Classification by {*x*} as symbol (frequency [%])	Classification by {*y*} as symbol (frequency [%])
v	v (84); V (14)	v (86); V (14)
V	V (84); v (16)	V (80); v (20)
x	x (78); X (20)	x (54); X (22)
X	X (72); x (26)	X (74); x (26)
y	y (78); Y (22)	y (84); Y (16)
Y	Y (78); y (22)	Y (82); y (18)
z	z (56); Z (42)	z (58); Z (42)
Z	Z (60); z (40)	Z (82); z (18)

Table 2: Classification results between almost not distinguishable symbols (writer "whfa")

Taking this fact into account and tolerating confusions between the symbols given in tab. 2, the recognition rates shown in tab. 3 are obtained.

Writer	Classification by {*x*}	Classification by {*y*}
hmfa	97.6 %	97.8 %
kmfa	96.0 %	96.8 %
whfa	97.5 %	96.7 %

Table 3: Recognition rates for each writer by classifying the sequences {*x*} or {*y*}; confusions between the symbols given in tab. 2 are not considered

If these confusions are not considered, the average error rate shrinks from 9.1% to 2.9%.

In comparison to tab. 1, there are almost no differences between the recognition results of the features {*x*} extracted by the vertical window and the features {*y*} extracted by the horizontal window.

7 FORMULA RECOGNITION

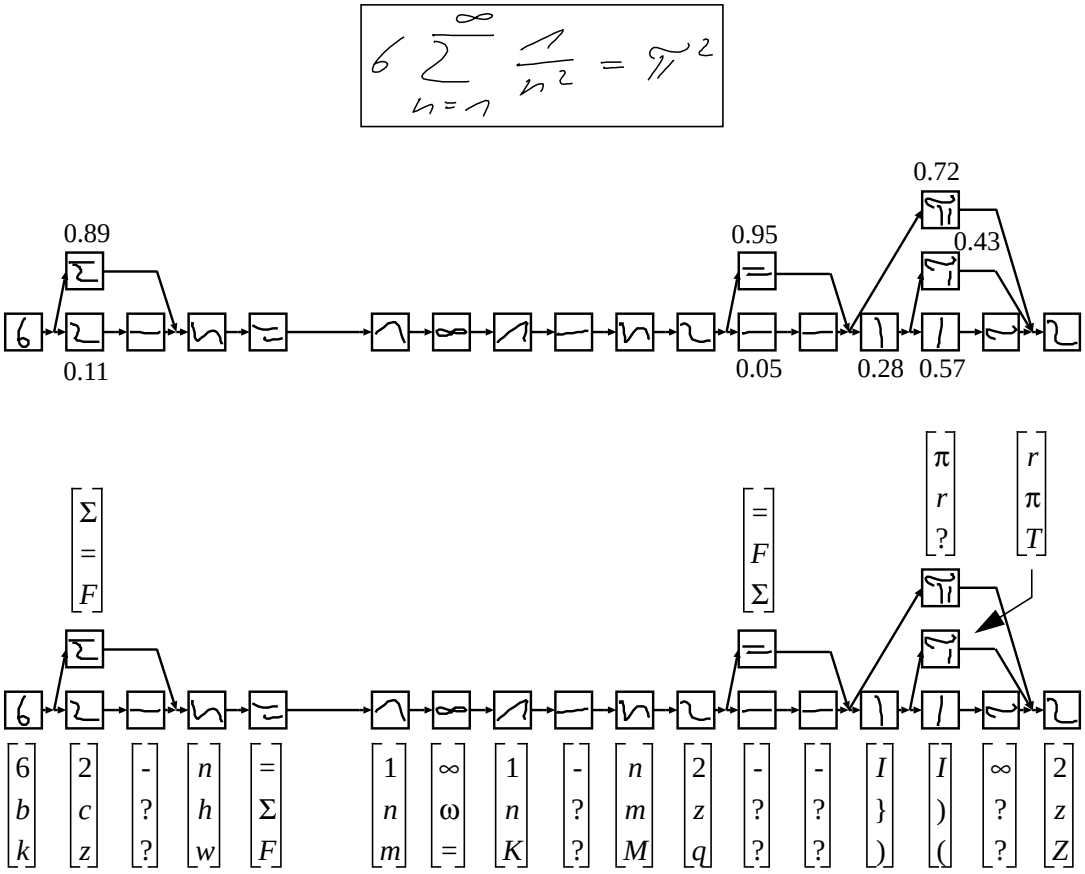
Classifying symbols within a formula means calculating the most likely path(s) through the symbol hypotheses net.

The probability of each path is given by the probability $P(m, g)$ obtained by generating the net and by the absolute probabilities $P(O|\lambda_v)$ obtained by symbol classification of the elements within the path. This combination is necessary because a probably preferred wrong path caused by a high $P(m, g)$ is devaluated by a poor symbol recognition probability $P(O|\lambda_v)$, because the path element is unknown to the recognizer.

Fig. 7 shows for a better overview the handwritten mathematical expression given in fig. 3 and the symbol hypotheses net with $P(m, g)$ next to each ramification (already shown in fig. 5). Furthermore, next to each element the symbol recognizer results (up to three hypotheses from top to bottom) and, by combining the single probabilities, the three most likely paths (and there probability) through the net are given. The symbol recognizer results are based on the feature sequence extracted by the vertical window.

The correct sequence of symbols is classified, but the probabilities of the second and third (incorrect) hypotheses are only a little smaller. The second hypotheses is caused by the symbol recognizer, which was almost not able to distinguish between the symbols "2" and "z". The overall probability of the path followed in the third hypotheses was almost the same compared to the path of the first hypotheses (the smaller probability of the stroke group is almost compensated by a higher symbol recognizer probability).

Reprint: Int. Workshop on Modern Modes of Man-Machine-Communication, Maribor, Slovenia, pp. 7/1-7/10, Jun. 1994



1. hypotheses (probability 24%): 6; Σ; n; =; 1; ∞; 1; -; n; 2; =; π; 2.
2. hypotheses (probability 23%): 6; Σ; n; =; 1; ∞; 1; -; n; z; =; π; 2.
3. hypotheses (probability 20%): 6; Σ; n; =; 1; ∞; 1; -; n; 2; =; I; r; 2.
4. hypotheses ……

Figure 7: Handwritten mathematical formula, symbol hypotheses net, symbol recognizer results and final classification (three hypotheses) of the expression

8 CONCLUSIONS AND FURTHER WORK

In this paper an efficient on-line recognition system for handwritten mathematical formulas is proposed. After formula preprocessing a symbol hypotheses net is generated by extracting different features for stroke unity. Each element of the symbol hypotheses net represents a possible symbol and therefore is classified by a HMM. Before classification the data of each symbol hypotheses are preprocessed, an image is calculated and features are extracted by moving a horizontal and vertical window along the image. The final classification of the handwritten input is done by calculating the best fitting path through the symbol hypotheses net under regard of the stroke group probabilities and the probabilities obtained by the HMM symbol recognizer.

In order to verify the effectiveness of the symbol recognition system, experiments are performed by single sampled symbols. The results indicate that the proposed scheme is very promising for writer-dependent recognition, however further work is required to examine these results by more writers and by a writer-independent modelling. Therefore more training data will be necessary. Furthermore it has to be examined if a combination of the classification results obtained by the horizontal and vertical window will improve the recognition results.

Even if the symbol recognizer results are very promising, its performance has to be examined by symbols out of a mathematical formula.

REFERENCES

[1] C. C. Tappert, C. Y. Suen, T. Wakahara, "On-line handwriting recognition - a survey," 9-th International Conference on Pattern Recognition, Vol. 2, pp. 1123-1131, Nov. 1988

[2] Z.-X. Wang, C. Faure, "Structural analyses of handwritten mathematical expressions," 9-th International Conference on Pattern Recognition, Vol. 1, pp. 32-34, Nov. 1988

[3] H. Wesnitzer, H.-J. Winkler, "Syntactical analyses of handwritten mathematical formulas," Institute for Human-Machine-Communication, Technical University Munich, 1993, internal report

[4] H. Fahrner, H.-J. Winkler, "Fuzzy approach for syntactical analyses of handwritten mathematical formulas," Institute for Human-Machine-Communication, Technical University Munich, 1994, internal report

[5] M. Koschinski, H.-J. Winkler, "Symbol segmentation in handwritten mathematical formulas," Institute for Human-Machine-Communication, Technical University Munich, 1994, internal report

[6] T.Caesar, J. Gloger, A. Kaltenmeier, E. Mandler, "Recognition of handwritten word images by statistical methods," in [TWHR93], pp. 409-416

[7] Proceedings of the 1993 IEEE International Conference on Acoustics, Speech, and Signal Processing, April 1993

[8] Proceedings of the Third International Workshop on Frontiers in Handwriting Recognition, May 1993

[9] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," Proceedings of the IEEE, Vol.37, No. 2, pp. 257-286, Feb. 1989

[10] D. Gehrung, H.-J. Winkler, "Implementation of a Hidden Markov Model and its application to on-line handwriting recognition," Institute for Human-Machine-Communication, Technical University Munich, 1993, internal report

[11] Y. Linde, A. Buzo, R. M. Gray, "An algorithm for vector quantizer design," IEEE Trans. COM, Vol. 28, No. 1, pp. 84-95, Jan. 1980