Handwritten Character Recognition based on Competitive Neural Trees

Theingi Htike
University of Computer Studies, Yangon
htike83@gmail.com

Dr.Yadana Thein
University of Computer Studies, Yangon

Abstract

Competitive Neural Trees (CNeT) are widely used for classification in pattern recognition. This paper applies this technique for recognizing of Myanmar handwritten characters. This paper involves three important steps, typically preprocessing, feature extraction and classification. The aim of preprocessing is to improve the quality of the images for further processing. For the extraction of features, four of the 3×3 masks are applied to word images to extract horizontal, vertical, right and left-diagonal lines. Afterwards, decomposed images should be partitioned to eight sectors around the center of image and the number of black pixels in each sector calculated and normalized by dividing them upon the total number of black pixels in word images for feature vector. These feature vector from word images which are used in Competitive Neural Trees (CNeT) for recognition purpose. This paper introduces a global search method for the CNeT, which is utilized for training.

Keywords – Myanmar handwritten characters, CNeT, global search method

Introduction

Optical Character Recognition (OCR) is a field of research in pattern recognition, artificial intelligence and machine vision. It refers to the mechanical or electronic translation of images of handwritten, typewritten or printed text into machine-editable text. Nowadays, the accurate recognition of machine printed characters is considered largely a solved problem. However, handwritten character recognition is comparatively difficult, as different people have different handwriting styles. So, handwritten OCR is still a subject of active research. The domain of hand written text recognition has two completely different problems of On-line and Off-line character recognition.

On-line character recognition [1] involves the automatic conversion of characters as it is written on a special digitizer, where a sensor picks up the pen-tip movements as well as pen-up/pen-down switching. The off-line character recognition is comparatively difficult, as different people have different handwriting styles and also the characters are extracted from documents of different intensity and background [2].

A review of the character recognition work done on Myanmar languages is excellently reviewed. Paper [4] proposed a system to recognize off –line Myanmar handwriting. They used discrete Hidden Markov Model. In paper [3] proposed Handwritten Myanmar Optical Character Recognition System. They used histogram labeling method for recognition. Recognition accuracy rate 98.18% was obtained. In paper [7] proposed an approach for extracting the user-entered date information from bank cheque images. They have analyzed the bank cheque characteristics and proposed approach that can adjust the position and extract filled in terms.

The Myanmar language is the official language and is more than one thousand years old. Myanmar script is considered a complex script by software developers, as it originated from Indic scripts like Thai or Khmer. The Myanmar (formerly known as Burmese) script developed from the Mon script, which was adapted from a southern Indian script during the 8th century. The earliest known inscriptions in the Burmese script date from the 11th century. Myanmar language is widely used in many offices such as passport, bank and tax etc. So, it is a very importance to develop the high accuracy character recognition system for Myanmar language. Texts in the Myanmar language use the Myanmar script.

In this paper, competitive neural trees are proposed for character recognition. In Myanmar character recognition fields, competitive neural trees had not been applied for recognition. Therefore, competitive neural trees are applied to develop off-line Myanmar handwritten recognition system. It is one of the fast supervised neural networks. Neural trees were introduced for character recognition in an attempt to combine advantages of neural networks and decision trees. The aim of the proposed system is to implement an effective approach which is able to recognize for Myanmar handwritten characters. The style of writing characters is different and they come in various sizes and shapes. The proposed system is easily to recognize handwritten characters of several different writing styles. Competitive neural trees are used to improve accuracy rate.

The remainder of the paper is organized as follows: **Section 2** describes Handwritten Myanmar language Nature. **Section 3** explains proposed system design and the various steps involved in the OCR System. **Section 4** presents competitive neural trees and **Section 5** explains the conclusion.

2. Handwritten Myanmar Language Nature

The interests in Myanmar handwritten characters recognition research have grown over the past few years but practical research is only a few works in research field. Because, the problem of Myanmar characters recognition is more difficult than English languages in respects including the similarity of characters, absence space between each word, etc. So, various character recognitions method not enough complete recognize Myanmar character. They are still in research field, not complete work.

The Myanmar language belongs to the Sino-Tibetan family of languages of which the Tibetan-Myanmar (Tibeto-Burman) subfamily forms a part. It has been classified by linguists as a monosyllabic or isolating language with agglutinative features. It is a tonal and analytic language. Texts in the Myanmar language use the Myanmar script, which derives from a Brahmirelated script borrowed from South India in about the eight century for the Mon language. The first inscription in Burmese dates from the following years and is written in an alphabet almost identical with Mon inscriptions. The earliest Myanmar and Mon language can be seen in MyaZeDi Stone inscription.

2.1 Myanmar Language Pattern

Myanmar language is the official language and widely used in many Myanmar states. Myanmar Language includes Kachin, Kayar, Kayin, Chin, Myanmar, Rakhine and Shan, etc. Myanmar language consists of (33) consonants, (12) vowels and (4) medials.

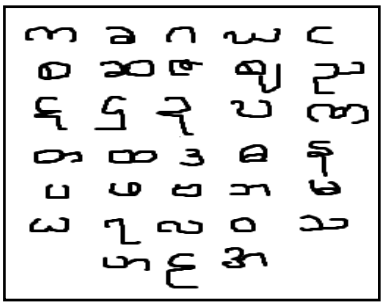


Figure.1. A Set of Myanmar Characters

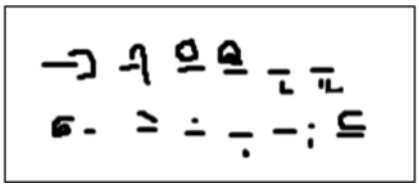


Figure.2. A Set of Myanmar Vowels

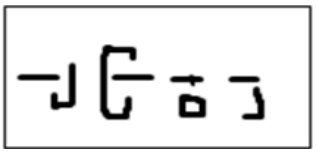
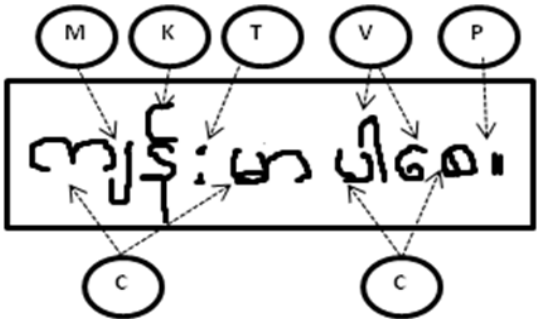


Figure.3. A Set of Myanmar Medials

2.2 Myanmar Language Writing Style

In Myanmar (Burmese) writing system, each letter has an inherent vowel. Other vowels are indicated using separate letters or diacritics which appear above, below, in front of, after or around the consonant. The rounded appearance of letters is a result of the use of palm leaves as the traditional writing material. Straight lines would have torn the leaves. The Burmese name for the script is 'round script', is written from left to right, as shown in Figure 4.

In Myanmar syllable structure, syllables or compound words are formed by consonants combining with vowels or medials. However, some syllables can be formed by just consonants, without any vowel. In our proposed system, we consider to implement not only 33 consonants but also the compound words.



C – Consonants P – Punctuation K – Killer
T – Tone V – Vowels M – Medial

Figure.4. Terms of Character's Sample

3. Proposed System Design

A typical character recognition system is characterized by a number of steps, which include

(1) Image Acquisition
(2) Preprocessing
(3) Feature Extraction, and
(4) Recognition.

The steps required for character recognition are described here in detail:

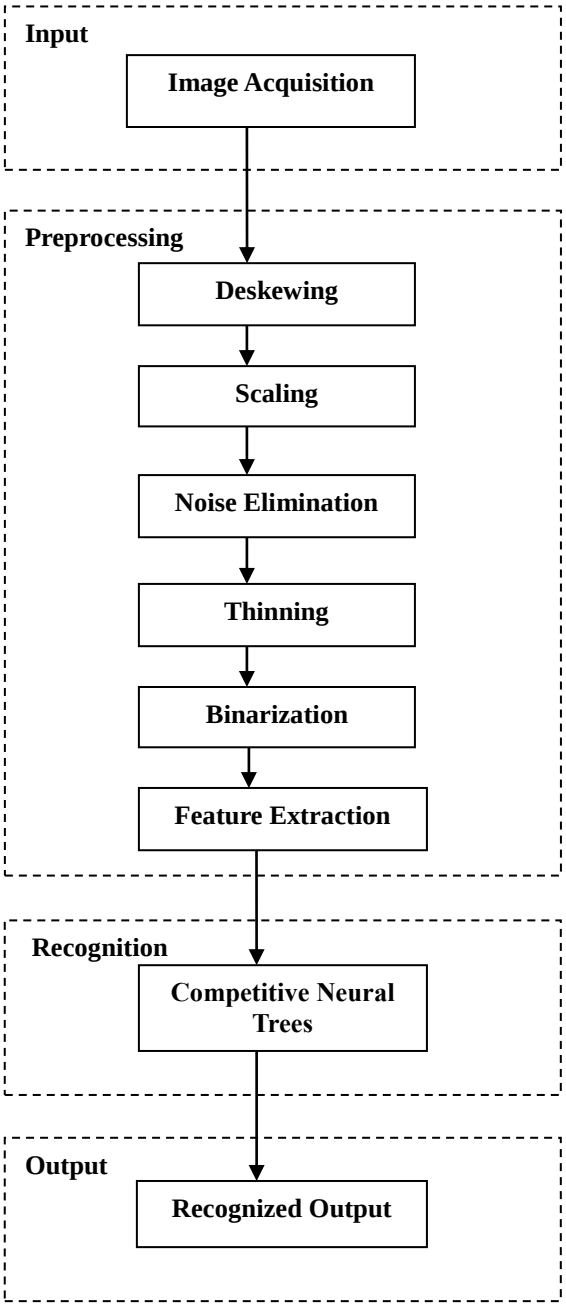


Figure.5. Proposed System Design

3.1 Image acquisition

It performs getting image from other sources like scanned image.

3.2 Preprocessing

Preprocessing aims at eliminating the variability that is inherent in hand-printed characters. The preprocessing techniques that have been employed in an attempt to increase the performance of the recognition process are as follows:

Deskewing: It is the process of first detecting whether the handwritten word has been written on a slope and then rotating the word if the slope's angle is too high so the baseline of the word is horizontal.

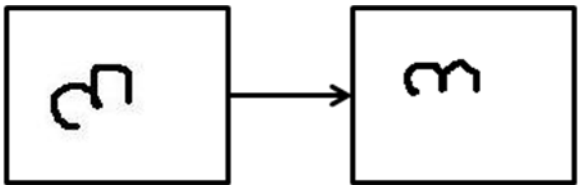


Figure.6. Deskewing

Scaling: Each character is scaled to fit within suitable matrix like 32x32 or 64x64 so that all characters have same data size.

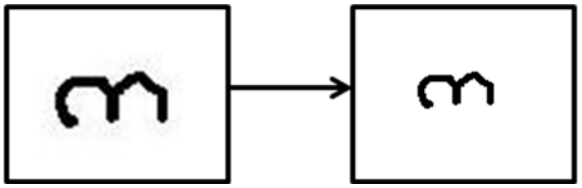


Figure.7. Scaling

Noise elimination: A little noise on the image can result in incorrect recognition. Median filters were applied on the image to reduce the salt and pepper noise and to smooth the images.

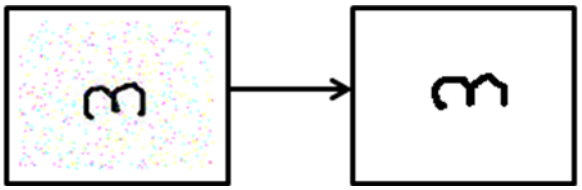


Figure.8. Noise elimination

Thinning: It is a process in which the skeleton of the word image is used to normalize the stroke width. The thinning stage caused reduction of samples thickness into a single pixel. In this paper, canny edge detector is used to reduce all lines to single pixel.

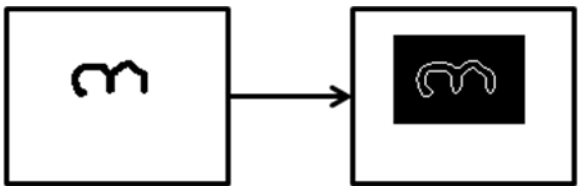


Figure.9. Thinning

Binarization: All hand printed characters are scanned into gray scale images. Each character image is traced vertically after converting the gray scale image into binary matrix.

Feature extraction: Four of the 3×3 masks are applied to word images to extract horizontal, vertical, right and left-diagonal lines. Images store separately correspond to each line. Afterward, decomposed images should be partitioned to eight sectors around

the center of image and the number of black pixels in each sector calculated and normalized by dividing them upon the total number of black pixels in word images for feature vector. Indeed this masks used for scanning horizontal, vertical, right and left diagonal lines to produce their equivalent images. The first mask used for separating the horizontal lines from the word images. Instead of input word image considered a zero matrix with the size of original image. This mask moved from left to right over the image and specified new values of elements of this matrix. The assigned value of this matrix depends on the value of pixels in original word image. That is, at position of **(i,j),** the value of **(i,j), (i,j+1)** and **(i,j-1)** pixels from matrix are the same and equal to one if all of these three pixels from slightly image have been one and zero, otherwise.

This procedure repeated for other three masks and each word image decomposed into four separate images corresponding to these masks. In the other word, with these masks horizontal, vertical, right and left diagonal lines in word images are separated. Finally, each of separated images is uniformly partitioned into eight sectors around the image center. The number of black pixels is calculated in each sector. These values normalized by dividing them upon the total number of black pixels in word images and used for feature value of that sector. This method solves the problem of scale invariance, has a low feature dimension and high recognition rate.

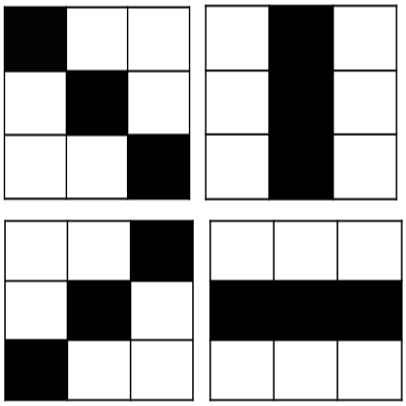


Figure.10.Four of 3×3 masks

3.3 Character Recognition with CNeT

This is the stage where an automated system declares that the inputted character belongs to a particular category. Competitive neural trees are used to recognize characters. An efficient character recognition scheme must be capable of producing appropriate class labels for input vectors that do not belong to the training set. This is called generalization.

Paper [6] proposed a structural adaptive intelligent tree (SAINT). The input feature space is hierarchically partitioned by using a tree-structured network that preserves a lattice topology at each sub network. Experimental results reveal that SAINT is very effective for the classification of a large set of real-world handwritten characters.

Neural trees are grown and pruned. Some algorithms grow a perfect tree that classifies all samples correctly [5]. Then a set of pruned subtrees is checked for performance on an independent testing set of samples and the best performing subtree is selected. This method is called backward pruning. Paper [8] proposed a structured parameter adaptive (SPA) NT which is a multilevel competitive NN with a tree topology. In this network, Hebbian learning algorithm is used for the adaptation of the network parameters, and some specific operations are used for adapting the tree structure: the creation of new nodes, the splitting of nodes in to more nodes, and the deletion of nodes from the network.

4. Competitive Neural Trees (CNeT)

The CNeT contains **m-ary** nodes and grows during learning by using inheritance to initialize new nodes. An **m-ary** tree is a data structure employed to improve external sorting in which for every node in the tree there are no more than **m** child nodes. Binary trees are a specific implementation of an m-ary tree where there are **m=2** child nodes for every node on the tree.

At the node level, the CNeT employs unsupervised competitive learning. The CNeT performs hierarchical clustering of the feature vectors presented to it as samples, while its growth is controlled by forward pruning. Because of the tree structure, the prototype in the CNeT close to any sample can be determined by searching only a fraction of the tree.

4.1. CNeT Architecture

The CNeT has a structured architecture. A hierarchy of identical nodes forms an **m-ary** tree. The tree is called an **m-ary** tree if every internal vertex has exactly **m** children. An **m-ary** tree with **m = 2** is called a binary tree. Each node contains **m** slots $S_1, S_2, .., S_m$ and a counter age that is incremented each time a sample is presented to that node. The behavior of the node changes as the counter age increases. Each slot $\mathbf{S_i}$ stores a prototype $v_i \in V \subset R^n$, a counter count, and a pointer to a node. The prototypes are updated to represent clusters of samples. The slot counter count is incremented each time the prototype of that slot is updated to match a sample.

.4.2 CNeT Learning

In the learning phase, the tree grows starting from a single node, the root. The prototypes of each

node form a minuscule competitive network. When a sample $x \in X$ arrive at a node, all of its prototypes $v_1, v_2, \ldots, v_m$ compete to match it. If $\mathbf{d}(\mathbf{x}, \mathbf{v_j})$ denotes the distance between $\mathbf{x}$ and $\mathbf{v_j}$, the prototype V_k is the winner if $d(x, v_k) < d(x, v_j), \forall_j \neq k$. The distance measure used in this paper is the squared Euclidean norm, defined as

$$d(x, v_j) = \| x - v_j \|^2 \quad (1)$$

According to this scheme, the winner V_k is the only prototype that is attracted by the input X arriving at the node. More specifically, the winner V_k is updated according to the equation

$$v_k^{new} = v_k^{old} + a(x - v_k^{old}) \quad (2)$$

where a is the learning rate. The learning rate a decreases exponentially with the ***age*** of a node according to the equation

$$a = a_0 \exp(-a_d age) \quad (3)$$

where a_0 is the initial value of the learning rate and a_d determines how fast a decreases. The update (2) moves the winner V_k closer to the sample x and, therefore, decreases the distance between the two. After a sequence of sample presentations and updates, each of the prototypes will respond to samples from a particular sub region of the input space.

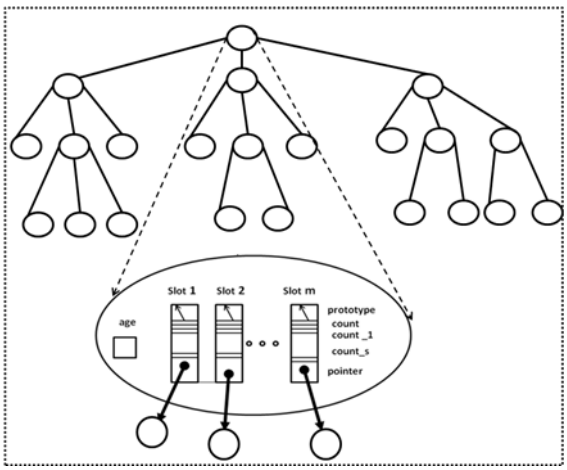


Figure.11. Tree structure of CNeT and node detail

Life Cycle: Each node goes through a life cycle. The life cycle of a node may be partitioned into the following phases.

Step 1: Creation (at age 0): the node is initialized; the node inherits properties from the parent slot such as the prototype and a fraction of the class counters.

Step 2: Youth (before the maturity age is reached): the prototypes compete to respond to the samples; the winning prototype is updated; the prototypes split the region of the input space that the node sees into sub regions.

Step 3: Maturity (after the maturity age has been reached): the prototypes still compete for the samples and they are updated; if a splitting criterion is $TRUE$, then a new child is created and is assigned to a slot.

Step 4: Frozen (as soon as a child is assigned): the prototypes compete for the inputs but they are not updated; if the winner has a child-node assigned, then it sends the sample to the child.

Step 5: Destruction (after all children have been destroyed).

Training Procedure: Do while stopping criterion is $FALSE$:

Step 1: Select randomly a sample x. Let C_j be class X that belongs to.

Step 2: Traverse the tree starting from the root to find a terminal prototype v_k that is close to x. Let n_l and s_k be the node and the slot that v_k belongs to, respectively.

Step 3: If the node n_l is not frozen, then update the prototype v_k according to (2).

Step 4: If a splitting criterion for the slot s_k is TRUE, then assign a new node as child to s_k and freeze the node n_l.

Step 5: Increment the counter $count_j$ for class C_j, the counter $count$ in slot s_k, and the counter age in node n_l.

Recall Procedures: The objective of the recall procedure is to produce a class label $i = \ell(x), i \in \mathcal{L}$, for each input vector x. Presentation of samples in the recall phase begins at the root of the tree and proceeds down to the leaves. The prototypes belonging to internal slots are used as signposts for the search. More specifically, these prototypes guide the search algorithm to find a terminal prototype $v_j = v(x)$ that is close to the input vector without looking at all terminal prototypes. The same search method used during training is also called for recall with the input vector x as the argument. The search will return a terminal prototype $v_j = v(x) \in V$, close to x. The recall strategy for the CNeT can be modified to allow the rejection of some samples in order to improve recognition accuracy.

4.3 Global (w) Search Method

The global (w) search method expands the nodes of the tree level by level, starting at the root. After this is done for all the nodes that are to be expanded at this level of the tree, the prototypes with the smallest distances are selected. If a selected prototype has a child-node assigned, this child-node will be expanded during the next expansion step. Suppose a selected prototype is a terminal prototype. If its distance to the given feature vector is the smallest so far, then the smallest distance is updated and the prototype is the new candidate to be selected for return. When no more prototypes are to be expanded, the global (w) search method terminates and returns the best terminal prototype seen.

The global (w) search method searches a subtree that has a width of at most w. Hence, the time required by the global (w) search method to return a terminal prototype is $O(wD_{tree})$. Clearly, the speed of this search method depends on the choice of the search width. If $w > 1$, then the search by the global (w) method takes longer but the probability that the search returns the closest prototype increases. Thus, the selection of w allows the user to balance the tradeoff between the time required for the search and the generalization ability of the CNeT.

4.4 Classification Phase

The function $\ell(x)$ can be approximated by the composition $q(v(x))$, where $q(\cdot)$ is a function that assigns class labels to the prototypes $v_k = v(x) \in V$. The approximation of $\ell(\cdot)$ by $q(v(\cdot))$ is efficient if the number of clusters inherent in the sample set is small compared to the number of samples. A class label $i = \ell(x)$ is then assigned to the input vector x by composing $q(\cdot)$and $v(\cdot)$ as $\ell(x) = q(v_k) = q(v(x))$. If v_k belongs to the slot s_k, then $q(v_k)$ is computed on the basis of the class counters $count_1, count_2, \ldots, count_s$ stored in the slot s_k. These class counters indicate how often the prototypes in the slot s_k responded to samples from each class. The classification function $q(\cdot)$ can be evaluated according to the following two methods.

- **Crisp Classification**: The class label $\ell(x) = q(v_k)$ is produced by majority vote. The feature vector x is classified to belong to the class C_i with the highest class counter $count_i$ in the slot s_k that responded to the input vector.
- **Likelihood Estimation**: This method does not return a class label but an estimate of the likelihood that a certain input vector belongs to the classes $C_1, C_2, \ldots, C_s$. The likelihood that an input vector belongs to the class C_j is determined as the ratio between the corresponding class counter $count_j$ and the sum $count$ of all class counters. Furthermore the distance between the input vector x and the responding prototypev_k can be utilized as a measure of confidence, with a smaller distance corresponding to higher confidence.

5. Conclusion

This paper describes CNeT for character recognition. The performance of a trained CNeT depends on the search method employed in the learning phases. The global search method is presented in this paper. The CNeT grown using the global search method was allowed to reject some ambiguous samples in order to improve its recognition accuracy. The reliability and recognition accuracy of the trained CNeT can be improved by a recall strategy. The speed of the learning process is mainly determined by the computational complexity of the search method.

REFERENCES

[1]A. Bharath and S. Madhvanath," Free Pad: a novel handwriting-based text input for pen and touch interfaces", Proceedings of the 13th international Conference on Intelligent User Interfaces, pp. 297-300, 2008.

[2]Bhardwaj, F. Farooq, H. Cao and V. Govindaraju," Topic based language models for OCR correction", Proceedings of the Second Workshop on Analytics For Noisy Unstructured Text Data, pp. 107-112, 2008.

[3]M.M.Nge,"Automatic Segmentation Of handwritten Myanmar Digits and Scripts on the Voucher Sheet", 2004

[4]K.Sandar, ''Off-line Myanmar Handwriting Recognition using Hidden Markov Models", 2005

[5] A. Sankar and R. J. Mammone, "Optimal pruning of neural tree networks for improved generalization," in *Proc. Int. Joint Conf. Neural Networks*, Seattle, WA, July 8–12, 1991, pp. 219–224.

[6] H. H. Song and S. W. Lee, "A self-organizing neural tree for large-set classification," IEEE Trans. Neural Networks, vol. 9, pp. 369–380, May 1998

[7] H.H.Thaung, "Automatic Extraction of Date from Myanmar bank Cheques", 2004

[8] T. Li, Y. Y. Tang, and L. Y. Fang, "A structure parameter adaptive (SPA) neural tree for the recognition of large character set," Pattern

Recognition, vol. 28, no. 3, pp. 315–329, 1995.