

13 A Neural Network Approach for Multifont and Size-Independent Recognition of Ethiopic Characters

Yaregal Assabie¹ and Josef Bigun²

School of Information Science, Computer and Electrical Engineering
 Halmstad University, Sweden
¹yaregal.assabie@ide.hh.se , ²josef.bigun@ide.hh.se

Abstract. Artificial neural networks are one of the most commonly used tools for character recognition problems, and usually they take gray values of 2D character images as inputs. In this paper, we propose a novel neural network classifier whose input is 1D string patterns generated from the spatial relationships of primitive structures of Ethiopic characters. The spatial relationships of primitives are modeled by a special tree structure from which a unique set of string patterns are generated for each character. Training the neural network with string patterns of different font types and styles enables the classifier to handle variations in font types, sizes, and styles. We use a pair of directional filters for extracting primitives and their spatial relationships. The robustness of the proposed recognition system is tested by real life documents and experimental results are reported.

13.1 Introduction

Ethiopic script is a writing system used mainly in Ethiopia by several languages like Geez, Amharic, Tigrigna, Guragegna, etc. The recently standardized alphabet has a total of 435 characters. However, the most commonly used alphabet has 34 base characters and six other orders conveniently written as shown in Table 1. The first column represents the base character and others are modifications of the base character representing vocalized sounds.

Table 1. Part of the Ethiopic alphabet

	Base Sound	Orders						
		1 st (ä)	2 nd (u)	3 rd (i)	4 th (a)	5 th (e)	6 th (o)	7 th (o)
1	ሀ	ሀ̣	ሀ̣̣	ሀ̣̣̣	ሀ̣̣̣̣	ሀ̣̣̣̣̣	ሀ̣̣̣̣̣̣	ሀ̣̣̣̣̣̣̣
2	ለ	ለ̣	ለ̣̣	ለ̣̣̣	ለ̣̣̣̣	ለ̣̣̣̣̣	ለ̣̣̣̣̣̣	ለ̣̣̣̣̣̣̣
3	ሐ	ሐ̣	ሐ̣̣	ሐ̣̣̣	ሐ̣̣̣̣	ሐ̣̣̣̣̣	ሐ̣̣̣̣̣̣	ሐ̣̣̣̣̣̣̣
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
34	ህ	ህ̣	ህ̣̣	ህ̣̣̣	ህ̣̣̣̣	ህ̣̣̣̣̣	ህ̣̣̣̣̣̣	ህ̣̣̣̣̣̣̣

Character recognition has been an area of research and development since 1950s (Suen, Mori, Kim, and Leung 2003). Unlike Latin and Asian scripts, the recognition of Ethiopic script is at its early stage, with the first published work appearing only recently in (Cowell and Hussain 2003). Moreover, Ethiopic script is a modification-based script having structurally complex characters with similar shapes posing an additional difficulty for classifiers. Artificial neural networks have been commonly used as classifiers in many pattern recognition problems, especially in handwritten character recognition. The traditional approach is to design a neural network classifier that takes the gray values of 2D document images as inputs (Dreyfus 2005). In this paper, we present a neural network classifier that takes 1D string patterns as inputs. The string patterns are generated from the primitive structures of Ethiopic character and their spatial relationships. We use a structural and syntactic model to generate patterns of primitives and their spatial relationships out of the structurally complex Ethiopic characters. The structural and syntactic model is described below and further exposed in detail in (Assabie and Bigun 2006a).

13.2 The Structural and Syntactic Model

13.2.1 Structural analysis of Ethiopic characters

Ethiopic characters are characterized by thick vertical and diagonal structures, and thin horizontal lines. The thick horizontal and diagonal structures are prominent structural features which give the general shape of the characters, and they form a set of *seven* primitive structures. The primitives differ from each other by their relative length, orientation, spatial position and structure. The classes of primitive structures (with example characters) are: *long vertical line* (**H**), *medium vertical line* (**፩**), *short vertical line* (**ሐ**), *long forward slash* (**ረ**), *medium forward slash* (**ሩ**), *backslash* (**ለ**), and *appendages* (**፳**). By extracting the relative size of each primitive structure, we are able to develop size-independent recognition system that doesn't need size normalization of document images.

Horizontal lines connect two or more of these primitive structures to form the overall complex structure of characters. Connections between primitives occur at one or more of the following three connection regions: *top* (*t*), *middle* (*m*), and *bottom* (*b*). The first connection detected as one goes from top to bottom is considered as *principal* and the other connections, if there exist, are *supplemental* connections. A total of 18 connection types are identified between primitives of the Ethiopic characters and a summary is given in Table 2. The principal connection is used to determine the spatial relationships between two primitives. Two connected primitives α and β are represented by the pattern $\alpha\mathbf{z}\beta$, where \mathbf{z} is an ordered pair (x,y) of the connection regions $t, m, \text{ or } b$. In this pattern, α is connected to β at region x of α , and β is connected to α at region y of β .

Table 2. Connection types between primitives

Principal Connection	Supplementary Connections					
	None	(m,b)	(b,m)	(b,b)	(m,m)+(b,m)	(m,m)+(b,b)
(t,t)	∏	∏	∏	∏	∏	∏
(t,m)	∏		∏			
(t,b)	∏					
(m,t)	∏	∏	∏	∏		
(m,m)	∏					
(m,b)	∏					
(b,t)	∏					
(b,m)	∏					
(b,b)	∏					

The spatial relationships of primitives representing a character are modeled by a special tree structure as shown in Fig. 1a. Each node in the tree stores data about the primitive and its connection type with its parent node. The child nodes correspond to the possible number of primitives (three to the left and four to the right) connected to the parent primitive. Primitive tree for a character is built first by selecting the left top primitive of a character as the root primitive. Then, based on their spatial positions, other primitives are appended to the tree recursively in the order of {left {top, middle, bottom}, parent, right {bottom, middle1, middle2, top}}. An example of a primitive tree is shown in Fig. 1b.

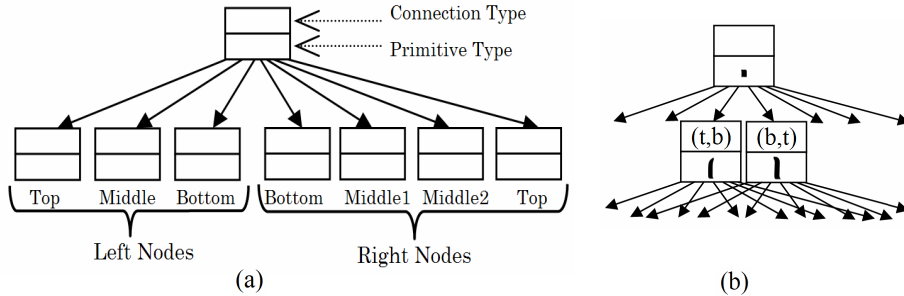


Figure 1. (a) General tree structure of characters, (b) primitive tree for the character \hat{n}

13.2.2 Extraction of structural features

Structural features are extracted by making use of a pair of directional filters both of which are computed from direction field tensor. Direction field tensor, also called the structure tensor S , is discussed in detail in (Bigun 2006) and its computation is presented in (Assabie and Bigun 2006b). Since the directional features are observed along lines, the local direction is also called Linear Symmetry (LS). The LS property of a local neighborhood $f(x,y)$ of an image f can be estimated by eigenvalue analysis of the direction field tensor using complex moments formulated as:

$$I_{mn} = \iint ((D_x + iD_y)f)^m ((D_x - iD_y)f)^n dx dy \quad (1)$$

where m and n are non-negative integers, and D_x and D_y Gaussian derivative operators. Among other orders, of interest to us are I_{10} , I_{11} , and I_{20} which are derived as follows.

$$I_{10} = \iint ((D_x + iD_y)f) dx dy \quad (2)$$

$$I_{11} = \iint |(D_x + iD_y)f|^2 dx dy \quad (3)$$

$$I_{20} = \iint ((D_x + iD_y)f)^2 dx dy \quad (4)$$

I_{10} is equivalent to the ordinary gradient field in which the angle shows the direction of intensity differences and the magnitude shows the average change in intensity. I_{11} measures the optimal amount of gray value changes in a local neighborhood. I_{20} is complex valued where its argument is the optimal local direction of pixels (the direction of major eigenvector of \mathbf{S}) in double angle representation and its magnitude is measure of the local LS strength (the difference of eigenvalues of \mathbf{S}). Fig. 2 shows I_{10} and I_{20} images displayed in color where the hue represents direction of pixels with the red color corresponding to the direction of zero degree.

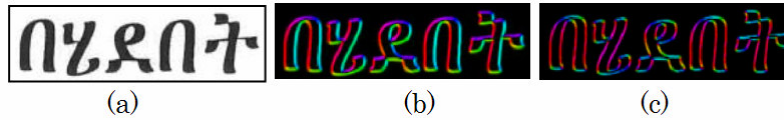


Figure 2. (a) Ethiopic text, (b) I_{10} of a, (c) I_{20} of a

In Fig. 2, it can be seen that a primitive structure in the text image results in two lines (at left and right edges) in the I_{10} and I_{20} images. The opposite directions for left and right edges in the I_{10} image provide convenient information for extraction of structural features. On the other hand, I_{20} has the following advantages over I_{10} :

I_{20} image encodes the optimal direction of pixels in the total least square sense, and therefore amplifies linear structures and suppresses non-linear structures.

Smoothing a primitive structure in the I_{10} image (with two lines of opposite directions) over a window leads to cancellation effects, whereas the I_{20} image gives the same result for gradients of opposite directions and therefore it is possible to smooth the structural features in the I_{20} image.

The magnitude in the I_{10} image depends on variations in the intensity of features against the background, whereas this kind of variation in the I_{20} image can be normalized by using the magnitude of $\frac{I_{20}}{I_{11}}$.

Therefore, we use the synergy effect of I_{10} and I_{20} to extract structural features. Extraction of the structural features is done on segmented characters. The horizontal area that lacks LS in the I_{20} image segments text lines, and the vertical area that lacks LS segments individual characters within each text line. The I_{20} image is used to group pixels into parts of primitives and connectors based on the direction information. After converting the double angle of I_{20} into a simple angle representation (by halving the argument of I_{20}), pixels of linear structures with directions of $[0..60]$ degrees are considered as parts of primitives and those with directions of $(60..90]$

degrees are considered as parts of connectors. The extracted linear structures in the I_{20} image are mapped onto the I_{10} image to classify them into left and right edges of primitives. A primitive is then formed from the matching left and right edges, as shown in Fig. 3.

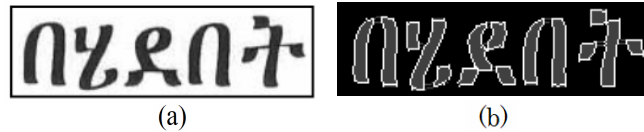



Figure 3. (a) Ethiopic text, (b) extraction of primitives

13.3 Neural Network Classifier

Neural networks are classification techniques inspired by neuronal operations in biological systems. Artificial neurons are typically organized into three layers: input, hidden and output. The input layer takes data of the unknown pattern whereas the output layer provides an interface for generating the recognition result. The hidden layer contains many of the neurons in various interconnected structures hidden from the outside view (Bishop 1995). Neural networks have been used for character recognition problems, and specifically widely used for recognition of handwritten text. Usually, the inputs to the neural network systems are pixel intensity values in two-dimensional space (Dreyfus 2005).

13.3.1 The neural network model

In Section 1, we introduced the design of the neural network model as a classifier system whose inputs are 1D string patterns of primitives and their spatial relationships. Data stored in the primitive tree of a character is converted into 1D string by recursively traversing the tree in the order of **left**{top, middle, bottom}, **parent**, **right**{bottom, middle1, middle2, top}. This is similar to *in-order* traversal of binary search trees and produces a unique string pattern for each primitive tree.

Since neural networks take numerical data as inputs, we assigned numbers to primitives and their spatial relationships. Three-digit and five-digit binary numbers are sufficient to represent the seven primitives and the eighteen spatial relationships, respectively. The longest string pattern generated from the primitive tree is for the character  which has 9 primitives (3x9 binary digits) and 9 spatial relationships (5x9 binary digits) generating a total string size of 72 binary digits. Each Ethiopic character in the alphabet is also encoded with a nine-digit binary number which is sufficient to represent all the 435 characters. Therefore, the neural network model has 72 input nodes and 9 output nodes. The hidden layer also has 72 nodes which is set optimally through experiments. Fig. 4 shows the diagram of the neural network model.

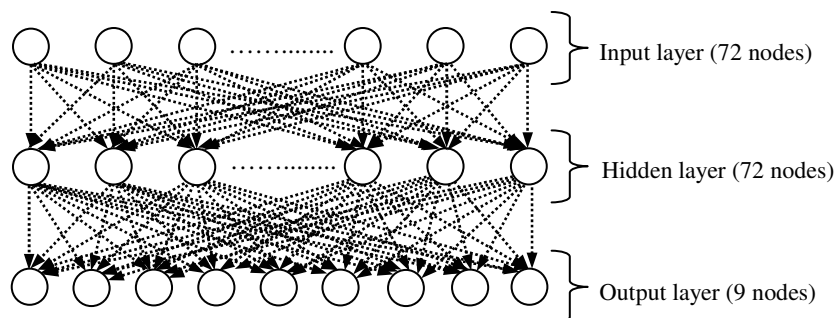


Figure 4. The neural network model

13.3.2. Training the network

The general structures of primitives and their spatial relationships remain similar under variations in fonts and their sizes. For example, irrespective of variations in font type and size, the character **U** is represented as two *Long Vertical Line* primitives both connected at the bottom, i.e., **{L, (b,b), J}**. Its italicized version (*U*) is represented as two *Forward Slash* primitives both connected at the bottom, i.e., **{I, (b,b), J}**. Thus, the set of patterns **{L, (b,b), J, I, (b,b), J}** represents the character **U** for various font types, styles and sizes. Such possibly occurring patterns of primitives and their spatial relationships are prepared for each character. The neural network is trained with such patterns of primitives and their relationships. For each Ethiopic character, possibly occurring 1D strings of primitives and their spatial are used as training samples. Table 3 shows training data created from the string patterns using NetMaker which is a data processing component in BrainMaker neural network tool.

Table 3. Training data

	Pattern	Pattern	Pattern	Pattern	Pattern	Pattern	Pattern	Pattern	Pattern	Input	Input	Input	Input	Input	...
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	...
308	0	1	0	0	1	1	0	1	1	0	0	0	0	0	...
309	0	1	0	0	1	1	0	1	1	0	0	0	0	0	...
310	0	1	0	0	1	1	1	0	0	0	0	0	0	0	...
311	0	1	0	0	1	1	1	0	0	0	0	0	0	0	...
312	0	1	0	0	1	1	1	0	1	0	0	0	0	0	...
313	0	1	0	0	1	1	1	1	0	0	0	0	0	0	...
314	0	1	0	0	1	1	1	1	0	0	0	0	0	0	...
315	0	1	0	0	1	1	1	1	0	0	0	0	0	0	...
316	0	1	0	0	1	1	1	1	0	0	0	0	0	0	...
317	0	1	0	0	1	1	1	1	1	0	0	0	0	0	...
318	0	1	0	0	1	1	1	1	1	0	0	0	0	0	...
319	0	1	0	1	0	0	0	0	0	0	0	0	0	0	...
320	0	1	0	1	0	0	0	0	0	0	0	0	0	0	...
321	0	1	0	1	0	0	0	0	1	0	0	0	0	0	...
...

Output data (9 values)
Input data (72 values)

A back propagation algorithm, which is one of the most commonly used supervised learning techniques, was used to train the neural network. The non-linear sigmoid function was used as a transfer function for the multilayer network architecture. Fig. 5 shows the first runs of the training progress using BrainMaker neural network tool. The graph depicts that the average error is declining over the course of training, which is a desired property of a good neural network model.

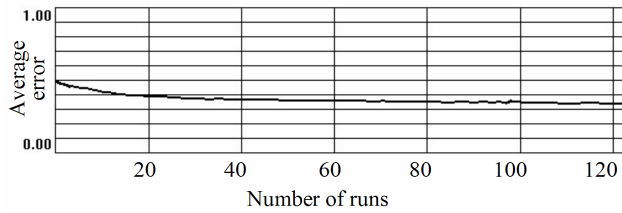


Figure 5. Progress of training

13.4 Experiments

13.4.1 Database development

Experiments are done using documents from Ethiopic Document Image Database (EDIDB) that we develop for testing the recognition system with different real life documents. The database consists of 1,204 images of documents (scanned with a resolution of 300dpi) taken from a wide range of document types such as books, newspapers, magazines, and printouts with various font types, sizes and styles. Sample documents from EDIDB are shown in Fig. 6.

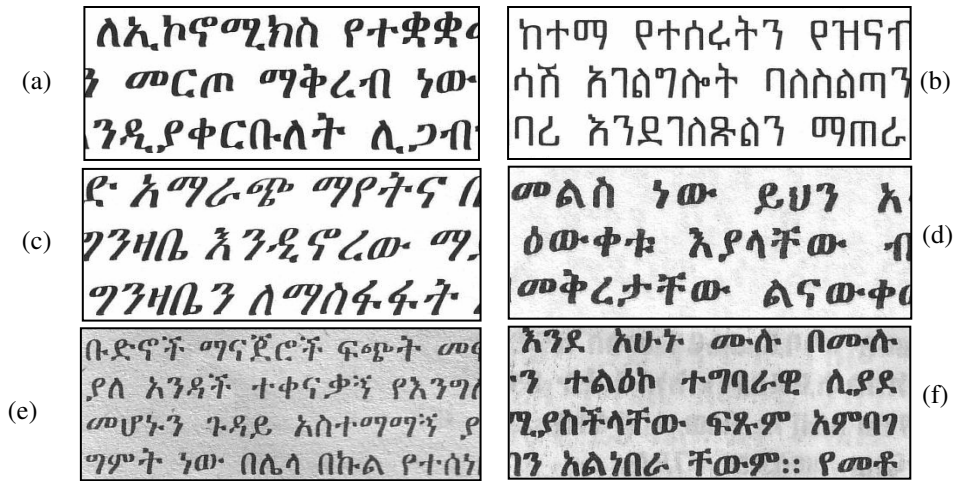


Figure 6. Sample documents with (a) VG2000 Main font, (b) VG2000 Agazian font, (c) VG Unicode italic font (d) book, (e) newspaper, (f) magazine

13.4.2 Recognition process

The only parameter that is changed during the recognition process is the size of Gaussian window which must be optimized according to font sizes and document types. For clean documents, we used a window of 3x3 pixels for texts with font sizes of less than or equal to 12, a window of 5x5 pixels for font sizes of 16, and a window of 7x7 pixels for font sizes of 20. On the other hand, for most documents taken from books, newspapers, and magazines (font sizes are equivalent to about 12), a window of 5x5 pixels was used because of their higher level of noise.

13.5 Result

The robustness of the system is tested with about 68 images of documents taken from EDIDB. Each image consists of an average number of 1,400 characters. For books, newspapers and magazines an average accuracy of 89% was achieved. The recognition system also tolerates documents with skewness of up to 10° . For clean printouts with Visual Geez Unicode font type and with 8, 10, 12, 16, and 20 font sizes, we obtained recognition rates of 93%, 93%, 94%, 95% and 95%, respectively. For printout documents with Visual Geez Unicode font type and 12 font size, recognition rates of 94%, 92% and 95% were achieved for normal, italic and bold font styles, respectively. The recognition system is also tested with various font types, each with 12 font size and the result is summarized in Table 4.

Table 4. Recognition results for different fonts

Font Type	Recognition (%)
Visual Geez Unicode	94
Visual Geez 2000 Main	94
Visual Geez 2000 Agazian	96
Power Geez	93
Geez Type	92

13.5 Discussion and Conclusion

It is quite common to see real life documents with varying font types, styles, and sizes. Thus, it has become important that recognition systems should be able to recognize documents irrespective of the variations in the characteristics of the text. To this effect, we have designed a structural and syntactic analyser that easily represents complex structures of characters using the spatial relationships of primitives which are easier to extract. Since we encode only the relative length of primitives, the structural and syntactic component of the system handles variations in font sizes without resizing the image. By training the neural network with primitive patterns of characters with different font types and styles, the network classifier is able to recognize the characters. The common errors in the recognition process arise from extraction of primitive structures and segmentation of characters. The recognition accuracy can be further improved by additional studies of these algorithms. The use of spell checker and parts of speech analyser can also help to increase the recognition result.

References

- Assabie, Y. and Bigun, J. (2006a) Structural and syntactic techniques for recognition of Ethiopic characters. In: *Structural, Syntactic, and Statistical Pattern Recognition*, volume LNCS-4109. Springer, Heidelberg, pp. 118-126.
- Assabie, Y. and Bigun, J. (2006b) Ethiopic character recognition using direction field tensor. In: *Proc. of the 18th Int'l Conf. Pattern Recognition-ICPR2006*. IEEE Computer Society, Piscataway, NJ, pp. 284-287.
- Bigun, J. (2006) *Vision with Direction: A Systematic Introduction to Image Processing and Vision*. Springer, Heidelberg.
- Bishop, C. M. (1995) *Neural Networks for Pattern Recognition*. Oxford University Press, USA.
- Cowell, J. and Hussain, F. (2003) Amharic Character Recognition Using a Fast Signature Based Algorithm. In: *Proc. of the 7th Int'l Conf. Information Visualization*. pp. 384-389.
- Dreyfus, G. (2005) *Neural Networks: Methodology and Applications*, Springer, Heidelberg.
- Suen, C. Y., Mori, S., Kim, S. H., and Leung, C. H. (2003) Analysis and Recognition of Asian Scripts-The State of the Art. In: *Proc. ICDAR2003*, vol. 2, Edinburgh, pp. 866-878.