Self-Organizing Maps for Clustering in Document Image Analysis

Simone Marinai, Emanuele Marino, and Giovanni Soda

University of Florence
Dipartimento di Sistemi e Informatica (DSI)
Via S. Marta, 3, I-50139, Firenze, Italy
{marinai,marino,soda}@dsi.unifi.it

Summary. In this chapter, we discuss the use of Self Organizing Maps (SOM) to deal with various tasks in Document Image Analysis. The SOM is a particular type of artificial neural network that computes, during the learning, an unsupervised clustering of the input data arranging the cluster centers in a lattice. After an overview of the previous applications of unsupervised learning in document image analysis, we present our recent work in the field. We describe the use of the SOM at three processing levels: the character clustering, the word clustering, and the layout clustering, with applications to word retrieval, document retrieval and page classification. In order to improve the clustering effectiveness, when dealing with small training sets, we propose an extension of the SOM training algorithm that considers the tangent distance so as to increase the SOM robustness with respect to small transformations of the patterns. Experiments on the use of this extended training algorithm are reported for both character and page layout clustering.

1 Introduction

Supervised classifiers are important components of most Document Image Analysis (DIA) systems. Few systems rely on unsupervised learning, or clustering. In unsupervised learning there is no explicit teacher and the training algorithm takes into account unlabeled samples. Systems that include clustering algorithms discover "natural" groupings, or clusters, of the input patterns that can be used in subsequent processing steps. Some specific knowledge about the nature of the patterns to be processed is considered in the design of the clustering algorithms, for instance to fix in advance the desired number of clusters to be found in the training data. An appropriate choice of this parameter is essential for maximizing the performance of clustering algorithms. Another important feature is the type of distance function that is embedded in the clustering algorithm.

In this chapter, we focus our attention on a particular kind of clustering algorithm, the Self Organizing Map (SOM) [1], that is well suited to

S. Marinai et al.: *Self-Organizing Maps for Clustering in Document Image Analysis*, Studies in Computational Intelligence (SCI) **90**, 193–219 (2008)
www.springerlink.com

dimensionality reduction and exploratory data analysis. We analyze various applications of SOM-based clustering in document image analysis sub-tasks and the incorporation of the tangent distance in the SOM training.

The chapter is organized as follows. In Section 2 we summarize the main features of SOM, as well as its standard training algorithm. In Section 3 we survey some recent applications of clustering algorithms in DIA with a particular emphasis on the use of the SOM. In Section 4 and in Section 5 we discuss our work related to SOM clustering at the character, word, and page levels. Lastly, we present our concluding remarks in Section 6.

2 Self Organizing Maps

Self Organizing Maps are a particular kind of unsupervised Artificial Neural Network (ANN). In this section we summarize some aspects of ANNs that can be relevant to appreciate the SOM peculiarities. We describe also some strategies that have been proposed in the literature to obtain invariant training systems. Additional information on ANNs can be found in other chapters of this book or in specific survey papers (see e.g. [2] for an overview of ANNs applications in the field of Document Image Analysis).

2.1 Artificial Neural Networks

Artificial Neural Networks are biologically inspired processing systems composed of a set of units, referred to as neurons, and a set of weighted connections between neurons where the signals are propagated. One of the first models for artificial neurons was the perceptron, which operates on continuous inputs and returns a Boolean output usually regarded as a classification of the input pattern. Perceptrons were dismissed primarily because of their limited capacity for function approximation. In contrast, Multi-Layer Perceptrons (MLPs) exhibit a universal interpolation capacity (see e.g. [3, 4]). In Multi-Layer Perceptrons the neurons are arranged into layers, and the connections link one layer to the next one. The input is regarded as a special layer which is propagated forward to hidden layers and, lastly, to the output. The term MLP usually denotes networks with sigmoidal neurons. Also RBF networks present the same layered architecture, but are often organized in two layers composed of RBF units and sigmoidal units, respectively.

The neural network's learning process can be either supervised or unsupervised. In the former case an expected target value is assigned to each example. In the latter case there is no external teacher that pre-defines the desired behavior of the network for the training samples. The supervised training of feed-forward architectures is performed by searching in the weight space for a set of parameters minimizing the mismatch between the target values and the network outputs. This search is typically made with the Back-propagation algorithm [5]. Competitive learning is another class of training algorithms where

the output neurons compete among themselves. Both unsupervised (e.g. *Vector Quantization* and *Self Organizing Maps* [6]) and supervised learning (e.g. *Learning Vector Quantization*) can be considered. In the Self-Organizing Map the neurons are usually arranged in a two dimensional lattice and each neuron receives inputs from the input layer and from the other neurons in the lattice. During the learning process the network performs clustering. The mapping of the neurons to class membership can be carried out upon completion of the learning process. For instance, each neuron can be labeled with the most frequent class among the training patterns belonging to its cluster.

2.2 Invariant Recognition

One of the most important goals in pattern recognition is the ability to achieve invariant recognition with respect to irrelevant distortions of the input patterns. For this purpose, prior knowledge on the pattern recognition problem to be solved can be considered in the training process adopting three main strategies: invariance by feature extraction, invariance by training, and invariance by structure [7].

In **invariance by feature extraction**, the prior knowledge is considered in the design of the feature extraction algorithm. A pattern can be represented either with a flat feature vector or with higher level structural representations (e.g. lists, trees, or graphs). In pattern recognition applications, including DIA, this is the prevailing strategy and large emphasis is placed on the design of appropriate feature extraction algorithms adopting standard paradigms for the classification.

In **invariance by training**, the prior knowledge is used when building the training set that is designed to contain samples depicting different aspects of the patterns to be processed. For instance, in the case of OCR, invariance to translations can be obtained by "building" training samples that are slightly shifted with respect to the original position (in [8] this type of processing is referred to as dithering). A similar approach has been proposed, for instance, for the generation of touching characters required to train neural segmentation algorithms [9]. A related application for the generation of synthetic handwritten data is described in the chapter by Varga and Bunke in this book.

In **invariance by structure** the classifier is designed to produce the same output when transformed versions of one pattern are presented to it. Convolutional networks [10] are examples of architectures based on feedforward neural networks which are invariant with respect to translations (for instance in handwriting recognition). Convolutional neural networks are based on local *receptive fields* where each hidden neuron (the receptive field) is only connected to a set of units in the previous layer. The first layers are organized as local receptive fields to extract some elementary visual features such as orientated edges, end-points and corners, independently of horizontal and vertical translation. Since the receptive fields of neighboring units overlap, a large number

of connections should be trained, thus giving raise to overfitting. To reduce the number of free parameters and, consequently, the risk of overfitting, the units in a layer are organized in planes within which all the units share the same set of weights.

One extension of this approach relies on the use of the tangent distance in the training algorithm so as to incorporate into the learning process some tolerance with respect to small known transformations [11]. For example, in [12] the tangent distance was used for computing the input-output distance of autoassociators, obtaining a recognition of handwritten characters that is invariant with respect to a set of eight transformations (x- and y-translation, rotation, scaling, axis-deformation, diagonal-deformation, x- and y-thickness). An autoassociator ([13], pp. 55, 161) is an MLP with the same number of input and output units and less neurons in the hidden layer. During the training the network, that is forced to reproduce the input to the output, is fed only with samples of one class. A modular classifier can be built by feeding in parallel one autoassociator for each class, and including one decision module which interprets the distances between the output vectors and the input pattern. The lower the distance, the higher the similarity between the pattern and the corresponding autoassociator class.

In this chapter the integration of the tangent distance in the context of character clustering will be discussed in Section 4.1, whereas the layout clustering based on tangent distance will be addressed in Section 5.2.

2.3 Self-Organizing Map

The Self Organizing Map (SOM [1]) is an artificial neural network that performs clustering by means of unsupervised competitive learning. In the SOM the neurons are usually arranged in a two dimensional lattice (the feature map). Each neuron gets information from the input layer and from the other neurons in the map. The training samples are usually described by real vectors $x(p) \in R^n$, where p is the index of the sample ($p \in [1, N_P]$), and N_P is the number of training patterns. Each node in the SOM contains a model vector $m_i \in R^n$ that can be regarded as a prototype of the patterns in the cluster. During the learning, the network performs clustering and the model vectors are modified so as to reflect cluster similarity. The goal of the mapping is to represent the patterns in the source space by corresponding points in a lower dimensional target space (e.g. a 2D space). In this mapping the distance and proximity relationships should be preserved as much as possible.

The initial values of the model vectors, $m_i(0)$, can be selected at random or can be initialized in some orderly fashion, for instance arranging the vectors along a two-dimensional subspace spanned by the two principal eigenvectors of the input data. The two most widely used learning algorithms are the on-line and the batch. We report in Algorithm 1 a summary of the on-line version that is the basis of the modified algorithm that is described later in this chapter. N_N is the number of neurons and N_{Train} is the number of training cycles.

Algorithm 1: On-line training

for all $i = 1$ to N_N **do**
  Initialize $m_i(0)$
end for
$p \leftarrow 0$;
for all $j = 0$ to N_{Train} **do**
  $p \leftarrow p + 1$;
  Find the BMU ($b(x)$)

$$||x(p) - m_{b(x)}|| = min_i\{||x(p) - m_i(j)||\} \tag{1}$$

  for all $i \in$ neighborhood of $b(x)$ **do**
    Adapt model vectors

$$m_i(j+1) = m_i(j) + h_{b(x),i}(j)(x(p) - m_i(j)), \tag{2}$$

    where:

$$h_{b(x),i}(j) = \alpha(j)exp(-\frac{||r_i - r_{b(x)}||^2}{2\sigma^2(j)}), \tag{3}$$

  end for
  for all $i \notin$ neighborhood of $b(x)$ **do**
    $m_i(j+1) = m_i(j)$
  end for
  decrease $\alpha(j)$
  decrease $\sigma(j)$
  if $p = N_P$ **then**
    $p \leftarrow 0$
  end if
end for

The algorithm is based on two main stages.

1. The training vector, $x(p)$, is compared with all the model vectors $m_i(j)$ and the *best matching unit* (*BMU*) on the map is identified. The *BMU* is the node having lowest distance with respect to $x(p)$. The final topological organization of the map is heavily influenced by the distance function considered in this step. Usually, the Euclidean distance is considered and the best matching unit $b(x)$ is identified by Eq. (1).
2. The model vector of the *BMU* as well as those of its neighboring nodes are changed so as to "move" towards the current input pattern $x(p)$ according to Eq. (2), where $h_{b(x),i}$ is the neighborhood function implemented with a smoothing kernel that is time-variable and is defined over the lattice points. The neighborhood function is a decreasing function of the distance between the i-th and the $b(x)$-th models on the map grid. The extension

of the kernel is also decreasing monotonically during the iterations. A common neighborhood function is based on the Gaussian function, Eq. (3), where $0 < \alpha(j) < 1$ is the learning-rate factor that decreases with the iterations, $r_i \in \Re^2$ and $r_{b(x)} \in \Re^2$ are the locations of the neurons in the lattice, and $\sigma(j)$ defines the width of the neighborhood function that also decreases monotonically.

One advantage of the use of the SOM with respect to other clustering algorithms is the spatial organization of the feature map that is achieved after the learning process. Basically, more similar clusters are closer than more different ones. Consequently, the distance among prototypes in the output layer of the map can be considered as a measure of similarity between patterns in the clusters, and this feature can be considered in several application domains.

3 Unsupervised Learning in Document Analysis

The aim of unsupervised learning, or clustering, is to find some structure in a set of patterns without the interaction with an explicit teacher. In particular, the goal of clustering is to identify a finite and discrete set of groupings in the patterns. There is no universally agreed definition of clusters, but in general the similarity between objects in a group is required to be larger than the similarity between objects belonging to different clusters.

When using clustering algorithms two important issues should be addressed: the choice of an appropriate similarity measure (or distance function) and the identification of a rule to select the number of clusters to be found. It is important to remark that the use of different distance measures with a given clustering algorithm can give rise to different groupings with significant differences in the final results.

The applications of clustering in pattern recognition follow three main lines ([14], page 517) that are briefly analyzed in the following.

Clustering algorithms are designed to deal with *unlabeled data*, this feature is therefore particularly helpful in applications where the human validation of the pattern membership needed to train a supervised classifier is difficult. A suitable combination of supervised and unsupervised approaches relies on the preliminary identification of the clusters on the basis of unlabeled patterns. A class is attached to the clusters relying on a reduced number of labeled patterns. We will discuss a page layout classification method based on this idea in Section 5.2.

A second approach relies on the application of unsupervised learning to *extract features* to be used, for instance, as input to a discriminant classifier. Features computed by means of unsupervised clustering can be considered also for retrieval systems. An application of this approach in DIA is character clustering that is discussed in Section 3.3.

Exploratory data analysis is an application of clustering techniques that allows to discover natural orderings of the patterns and can be suitably integrated in complex pattern recognition systems. Several techniques described in this chapter can be used in this context.

In the rest of this section we analyze some recent applications of clustering, with a special emphasis on SOM-based approaches, in the field of document image analysis.

3.1 Symbol Thinning

Thinning algorithms are used, in pre-processing, to extract features based on the symbol skeleton. These features, when used in handwritten character recognition systems, allow a recognition independent from the stroke thickness. Ahmed proposed in [15] a clustering-based skeletonization algorithm (CBSA) implemented with SOM. The CBSA is composed by two main steps: in the first step some clusters, corresponding to adjacent pixels, are located in the input image; in the second step the skeleton is built connecting together the neighboring cluster centers. The clustering step is implemented in [15] by means of a particular SOM (the self-organizing graph) where the adjacency of neurons can change during learning. More recently, a topology-adaptive self-organizing neural network has been proposed for skeletonization [16]. The map grows in size over time and improves the performance with respect to a SOM having a fixed dimension. The system can handle rotated patterns and works with binary and gray level images. A similar approach is described in [17], whereas a multi-scale skeletonization method based on SOM is described in [18].

3.2 Layout Analysis

Layout analysis is executed after the pre-processing with the aim of extracting homogeneous regions from the document image assigning a semantic meaning to each region. When dealing with color documents, the layout analysis consists in the identification of regions with uniform color. To this purpose, different colors are first identified and then pixels having the same color are grouped together. The color identification is frequently addressed with clustering in the color space. Global color clustering methods are described in [19] and in [20]. In [19] the clustering is obtained by using the Euclidean Minimum Spanning Tree (EMST) in color space, that is claimed to provide better results than the k-means algorithm. In [20] three-dimensional morphological operators are adopted to erode the regions in the color space. Iterating this erosion the cluster centers are identified. In [21], a color grouping algorithm, the *LOCUSi* method, is proposed. This algorithm extracts the clusters in the RGB space and is based on the analysis of the expected shape of clusters in this space.

Clustering at the spatial level is adopted in [22] for document image segmentation. In this case, the text lines are represented by means of the proposed *interval encoding* and subsequently clustered with the k-means algorithm. A hierarchical clustering algorithm is adopted in [23] for grouping closest connected components in an OCR system that is aimed at processing mathematical equations.

3.3 Character Clustering

Character clustering is the basis of some *character-like coding* techniques where similar objects, usually corresponding to characters, are clustered on the basis of their shape. Each word is then represented by concatenating the codes assigned to the individual objects. In character-like coding, in contrast with OCR, no alphabetical class is assigned to symbols. The query is encoded with the same algorithm used during the indexing and is compared with the indexed words taking into account various matching approaches. For instance, in [24] the words are represented with strings and compared by means of an inexact string matching technique. By adopting this symbolic representation indexed words can be sorted allowing users to retrieve words printed with different fonts as well as to satisfy partial-match queries.

Character clustering is applied also in some document image compression algorithms that first group together similar symbols roughly corresponding to characters. The characters and the background image are afterwards compressed with specific algorithms [25, 26]. In [26] the characters are compressed by extracting the *marks* (connected components) in each page, and a library of marks is built. Each mark is then replaced with a pointer to the closest item in the library. The library is obtained with a simple template matching clustering algorithm that provides good results when processing documents with a small variability of fonts and a low level of noise.

A related method is addressed in [27], where a hierarchical clustering algorithm is used to enhance degraded document images. Bitmaps of the symbols belonging to each cluster are identified and an average symbol for each class is computed. By replacing the original symbols with the average bitmap it is possible to render the document at arbitrary resolutions and enhance degraded document images.

3.4 Handwriting

Clustering algorithms, and in particular the SOM, have been widely used for the recognition of isolated handwritten digits. A three-stage recognition system dealing with handwritten numerals is described in [28]. The first stage is based on a SOM whose aim is to create prototypes representing parts of the characters that capture the similarities between digits. The gradual shape variations are represented in the feature map, since closer prototypes generally represent similar patterns. In the second stage a suitable function is used

to convert the distances between the input (unknown) pattern and the prototypes into membership values so as to obtain a fuzzy membership. The third stage performs the final classification by means of a fully connected MLP.

A three-dimensional SOM for unconstrained handwritten numeral recognition is described in [29]. The third dimension is defined by taking into account 11 layers of 9x9 SOMs. The neighborhood consists of the units that are within a cube centered on the BMU.

A hybrid handwritten word recognition system using SOM, discrete HMM, and evolutionary programming has been proposed in [30]. In this system the purpose of the SOM clustering is to partition the feature space into a set of codeword vectors to limit the number of observation symbols in discrete HMM training. The weight vectors of the trained map are used as codewords to describe the word frames, that are represented by a sequence of 2-dimensional codeword positions in the map. The neighborhood information preserved by the SOM is used for smoothing the trained HMM parameters.

In [31] handprinted character recognition is addressed by representing the characters with a fixed number of ellipses that capture the local structure of the strokes. Each ellipse is described with the center, the length and the orientation of the mayor axis. A modified SOM is used to accomplish one elastic matching and find the correspondence between the feature points. When the network converges, a mapping between the input feature patterns and the neuron support is obtained.

3.5 Structure Adaptive Classifiers

Structure adaptive classifiers automatically adjust their structure to the uneven distribution of classes in the pattern space. These classifiers are particularly useful in applications where a large number of classes is addressed, such as oriental character recognition. In this context, clustering algorithms are frequently used for grouping together most similar patterns in modular classifiers, that are mostly based on parallel and serial combinations [32].

In serial combinations [2] the classifiers are arranged in a list. For each unknown pattern the first classifier decides if a further refinement of the decision is required by one or more subsequent classifiers. The individual classifiers are usually applied in increasing order of complexity and the "simplest" symbols are recognized first, whereas the more difficult ones are processed by next classifiers. Some methods analyze the recognition results of the first classifier in order to identify the most confused classes. Examples of this approach are proposed in [33] and in [34]. In [33] maxima in the confusion table are automatically identified. In [34] the confusion table of the first stage, a feature-based OCR, is manually analyzed to find nine sets of classes corresponding to most confused groups of characters (e.g. {S,5,6}; {B,D,O,8}), which are recognized by appropriate MLPs. Alternatively, clustering algorithms can be explicitly applied to the characters belonging to the learning set. For instance, Su *et al.* [35] cluster together similar characters and organize the classifiers so

that most indistinguishable classes (e.g. "4" and "A") are recognized by the last networks. Each network is subsequently trained to recognize the patterns belonging to its classes, and to reject the patterns that should be recognized by the other classifiers.

In hierarchical combinations, the classifiers are arranged in a tree. A structure adaptation method for the recognition of Korean characters using a self organizing neural tree has been proposed in [36]. The basic idea is to automatically find a network structure and size suitable for the classification of large-set and complex patterns. The tree-structured network is based on subnetworks that are logically connected to nodes in the previous level. As a matter of fact, subnetworks define with higher resolution regions of the pattern space containing more patterns. Another hierarchical structure adaptation SOM for the recognition of handwritten digits is proposed in [37].

3.6 Text

The WEBSOM [38] is a SOM-based system that is able to organize large document collections according to textual similarities. The feature vectors describing the documents are statistical representations of their vocabularies. One peculiarity of the system described in [38] is the scaling up of the SOM algorithm in order to process large collections of high-dimensional data. In the experiments 6,840,568 patent abstracts have been mapped onto a 1,002,240-node SOM. To reduce the feature vector size some random projections of weighted word histograms are computed, obtaining 500-dimensional vectors. A similar application has been described in [39].

In [40] the documents are represented with feature vectors containing the occurrences of 489 terms in each document so as to reveal the document similarity. One hierarchical feature map is then built considering this document representation. This hierarchical representation is claimed to be well suited for text archive organization.

The SOM clustering is used in [41] to build a lexical analyzer designed to focus on a very limited sub-set of the whole dictionary. Each string S is represented by a vector $X = [X_0, X_1, \ldots, X_{25}]$ where X_i corresponds to the number of characters of class C_i (C_0 = 'A' , $\ldots$, C_{25} = 'Z') in the string S (the anagrams of S share the same representation). The map is a two dimensional array, organized as a tore to avoid singularity effects on the sides. The neighborhood relations in the projected space of the map are used to define a short list of hypotheses considered for spell checking.

3.7 Discussion

In this section, we analyzed some applications of unsupervised clustering in the domain of DIA. The main advantage of SOM clustering, with respect to other clustering algorithms like k-means, is the spatial organization of the neurons that reflects cluster similarity into prototype proximity in the 2D space. The

distance among prototypes in the SOM map can therefore be considered as an estimate of the similarity between objects belonging to the clusters. It is important to remark that in general the use of SOM for multivariate data projection on large data sets is not advisable due to the high computational cost [42]. However, usually a reduced number of objects (obtained from few documents) are used to compute the mapping that is subsequently adopted to label all the patterns to be processed.

In the next sections we describe the use of the SOM at three processing levels: the character clustering, the word clustering, and the layout clustering, with applications to word retrieval, to document retrieval and to page classification.

4 Word Indexing

Word indexing, that aims at a fast retrieval of words in a document collection, can either process the output of OCR engines or directly work on the document image. Several strategies have been proposed to deal with OCR errors [43, 44]. In most approaches the uncorrected OCR output is used for text indexing and the words are compared with the query by means of *string edit distance* algorithms. This strategy has been improved by modifying the edit costs for the most common OCR errors (e.g. [45]).

When the use of OCR is not advisable, either due to the low quality of images or to the presence of non-standard fonts, then image-based word retrieval is a viable alternative. Two main strategies have been considered in this framework: character-like coding and holistic word representation.

In methods based on character-like coding some objects, that potentially correspond to characters, are extracted from each word. The word is then represented by concatenating the codes assigned to the objects. In so doing similar words share most of the codes.

In the holistic approach each word image is encoded by means of some global features (e.g. the number of characters or the number of ascenders/descenders) [46]. A particular case of holistic word representation is zoning (e.g. [47]) that consists of overlapping the word image with a fixed-size grid. Suitable features, such as the density of black pixels, are computed in each grid region. Most keyword spotting methods are based on this kind of representation. For instance, signal processing techniques are used in [48] to allow scale and translation invariance. Holistic shape features for handwritten word image retrieval are described also in [49] where a training set is used to learn a joint probability distribution between word features and their transcriptions.

In this section, we discuss the use of SOM clustering to perform word indexing with two approaches: a character-based and a holistic one. In both cases, during the indexing each document image is first processed with a layout analysis tool that identifies the text regions and extracts the words

using an RLSA-based algorithm. To perform the word retrieval a query word is inserted into one text field in the user interface. A word image is computed by processing the query word with the LaTeX software and this image is used to compute a suitable word representation either based on character clustering or on holistic features. Lastly, the indexed words are sorted estimating their similarity to the query by appropriate processing steps that are described in detail in [50] for the character-based encoding and in [51] for the holistic-based approach. In this section we focus our attention on the peculiarities of character and word clusterings.

4.1 Character Clustering

In word indexing based on character coding each word is first split into Character Objects (CO) that in most cases correspond to isolated characters (sometimes one CO can comprise two touching characters). The COs extracted from a few random pages are used to compute appropriate collection-specific character prototypes by means of SOM clustering. To improve the clustering we include the tangent distance into the on-line training algorithm as detailed in the following.

Tangent Distance

The SOM training algorithm generally relies on the Euclidean distance to compare the training patterns and the model vectors. Unfortunately, the Euclidean distance between two patterns is very sensitive to small transformations and a limited displacement of a character can give rise to a large value of the distance, since many pixels in the two patterns are no longer aligned. To address this problem, we propose the use of the tangent distance in the SOM training algorithm.

The invariance of the training algorithm with respect to transformations of the patterns (Section 2.2) can be achieved with techniques based on invariance by structure, invariance by feature extraction, and invariance by training. The use of the tangent distance into a training algorithm can be considered an invariance by structure technique whose goal is the incorporation in the distance function of the tolerance with respect to small transformations in the pattern space.

We shortly outline in the following the tangent distance principle. Readers interested in more details can refer, for instance, to [11]. Let us suppose to transform a pattern P with a non linear transformation t that is controlled by one parameter β (for instance t can be the rotation of the pattern with the angle β). In the pattern space the set of all the transformed patterns $S_P = \{x \mid \exists\ \beta\ x = t(P, \beta)\}$ can be regarded as a one-dimensional curve parametrized by β. In the general case several transformations are combined together. The possible transformations can be characterized by a vector of n parameters ($\vec{\beta}$), that is expected to describe all the possible deformations

that can be applied to the patterns. The patterns in S_P are now arranged in a manifold and two objects P and Q in the same class are expected to generate identical manifolds $S_P \equiv S_Q$. Two problems should be addressed to compute the distance between patterns taking into account these manifolds.

The first problem is related to the identification of appropriate transformations that should generate realistic patterns. In the case of handwritten characters, some standard transformations have been proposed, such as rotation, translation, and line thickening. However, real patterns are usually subjected to transformations that are difficult to model and actual patterns usually stay close to the manifold but are not perfectly described by it.

The second problem is related to the computational cost required to evaluate the distance between two manifolds. The solution addressed by the tangent distance approach is based on a local approximation of one manifold by means of the hyperplane tangent to it in the point P. The tangent plane is defined with the linear combination of the n vectors computed by applying small independent transformations to the original pattern.

The tangent distance between patterns P and Q can be computed by first defining the tangent planes to P and Q:

$$\begin{aligned} T_P(\overrightarrow{\beta_P}) &= P + L_P\overrightarrow{\beta_P} \\ T_Q(\overrightarrow{\beta_Q}) &= Q + L_Q\overrightarrow{\beta_Q} \end{aligned} \tag{4}$$

where L_P and L_Q are the matrices containing the tangent vectors which are usually pre-computed. The *double-sided* tangent distance is defined as the minimum Euclidean distance between the tangent planes:

$$TD(P,Q) = min_{x \in T_P, y \in T_Q} ||x - y||^2. \tag{5}$$

Computing the tangent distance amounts to solve a linear least squares problem (e.g. see [11]). In some cases it is simpler to use the *one-sided* tangent distance where the minimum distance between one pattern and the plane tangent to the other pattern is computed:

$$TD1(P,Q) = min_{x \in T_P} ||x - Q||^2. \tag{6}$$

Character Clustering with Tangent Distance

Character object labeling is made on the basis of the prototypes that are obtained by clustering the COs contained in some pages of the collection to be indexed. Each CO image is scaled to fit a fixed size grid obtaining an 80-dimensional feature vector that is used as input to the clustering algorithm. The SOM-based character clustering computed using the Euclidean distance is very sensitive to small local transformations of the character. To reduce these problems in the proposed SOM_TD model we use the tangent distance instead of the Euclidean distance during the SOM training. To speed-up the

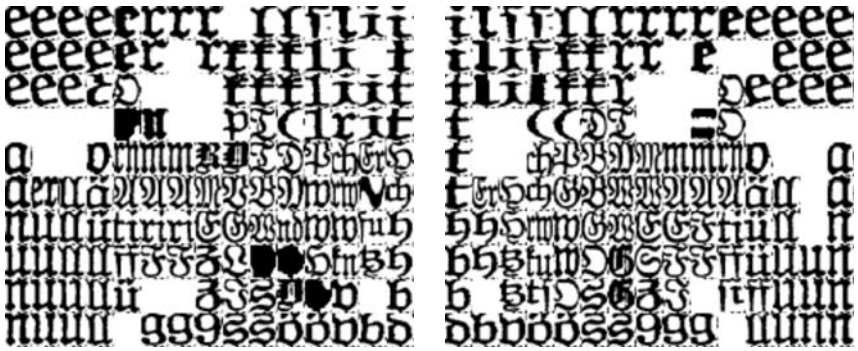

Fig. 1. Two SOMs built on the Gothic data-set. Left: map computed with the SOM_TD. Right: map computed with the standard SOM training

training we pre-compute the tangent vectors for each pattern (CO) in the training set. Even if the estimation of the tangent vectors is computationally expensive, it is important to remark that the vectors are computed only once for each pattern. During the training, we use the one-sided tangent distance, Eq. (6), to identify the BMU for each pattern, replacing Eq. (1) with:

$$||x(p) - m_{b(x)}|| = min_i\{TD1(x(p), m_i(j))\}. \tag{7}$$

In the next section we compare the results obtained performing the character clustering with the standard SOM training algorithm and with the SOM_TD model.

Experimental Results

To evaluate the effectiveness of the proposed SOM training model we used two data-sets having complementary features described in [50]: the Gothic and the French one. The Gothic data-set contains 14 pages printed with a Gothic font that is not recognized by current off-the-shelf OCR packages. The second data-set contains more than 600 pages printed with a standard font.

The tangent distance is a technique appropriate for data-sets having a small number of training examples. It is therefore not surprising that the experimental results on the French data-set do not show significant differences between the standard and the tangent SOM. Therefore, we will not discuss results for this collection in the chapter.

On the other hand, when dealing with the Gothic data we have some interesting results that it is worth to analyze. A pictorial comparison of the maps computed with the two methods is shown in Figure 1. The quantitative comparison of the two approaches is obtained by testing the retrieval system described in [50] without the use of the word alignment method that is described in that paper. The precision-recall plots shown in Figure 2 are

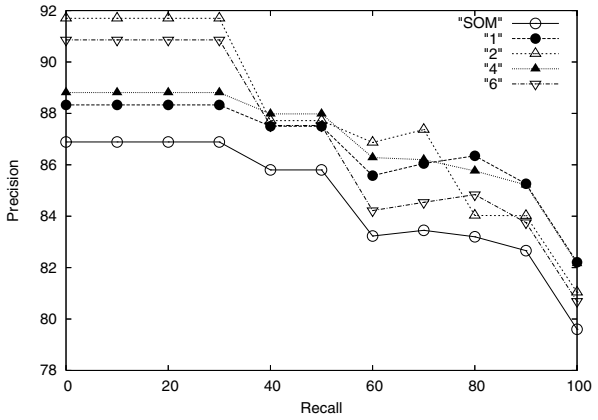

Fig. 2. Comparison of precision-recall plots with and without the use of the tangent distance. The transformations 1,2, 4, and 6 correspond to vertical shift, hyperbolic, scale and line thickness, respectively

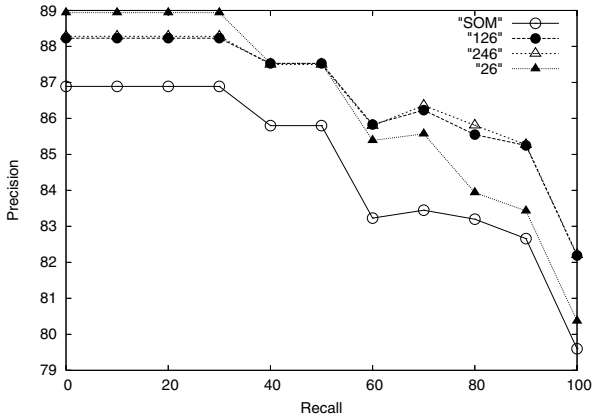

Fig. 3. Comparison of different combinations of various tangent vectors. For instance, the black circle corresponds to the combination of vertical shift, hyperbolic, and line thickness

computed by running the word retrieval system with 26 query words representative of both frequent and rare words and averaging the single plots. The five plots correspond to the standard map ("SOM") and to the results obtained with the SOM_TD model using some of the most important transformations. The tangent vectors computed with these transformations allow us to obtain better results with respect to the standard model, since the precision-recall plots are always better than the SOM one. Figure 3 shows the results that can be obtained with few combinations of the transformations analyzed in Figure 2.

Fig. 4. Graphical representation of a WordSOM

4.2 Holistic Word Clustering

In holistic word indexing the SOM is used to cluster together most similar words (from a graphic point of view). The word images are linearly scaled to appropriate normalized dimensions, obtaining a vectorial representation where the value in each vector item is the average gray level of the pixels belonging to the corresponding grid cell. To allow a uniform representation for variable size words we compute six maps. Each map contains the words having aspect-ratio in a pre-defined interval. The main drawback of this approach is the large vector size (hundreds of items) that is reflected into a long training time. However, it should be remarked that the training is performed during the indexing, that can be considered an off-line stage.

In Figure 4 we show a *WordSOM* computed by processing the pages in a book belonging to the French data-set previously mentioned. A deeper analysis of the contents of four neurons is depicted in Figure 5 where the words in the clusters are ordered from top to bottom on the basis of the distance with respect to the model vector of the neuron. As we can expect, the farthest words are generally loosely related with those closer to the model vector.

The large vector size affects the retrieval performance for problems related to the curse of dimensionality. To speed-up the search in high dimensional spaces we proposed in [51] a method based on the combination of SOM clustering with the search in a lower dimensional space obtained by the PCA projection. The main steps performed in the word retrieval are as follows. We first identify the three clusters closer to the query. In the second step we search the most similar words sorting the PCA-projected vectors. Lastly, to merge the three lists and refine the final ranking, we compute the distance in the original space between the query word and the most similar words in the three lists. Some detailed experiments on the use of the SOM for holistic word indexing are described in [51].

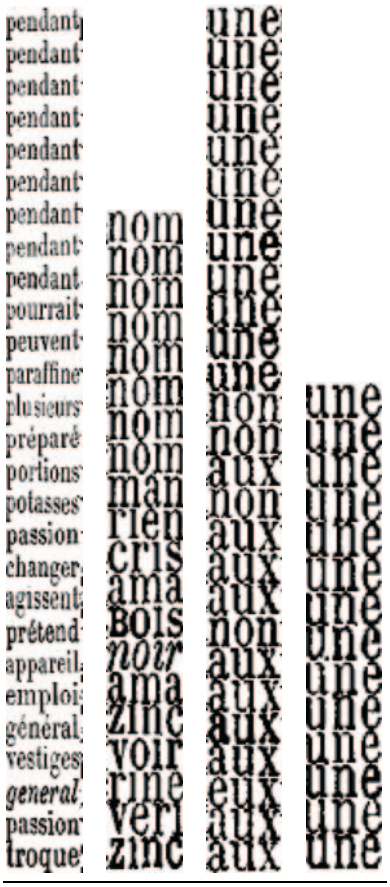

Fig. 5. Contents of four neurons of the *WordSOM* shown in Figure 4. The words are sorted on the basis of the distance from the cluster centroid

5 Page Indexing

In this section, we discuss two approaches aimed at indexing documents at the page level. The first application (Section 5.1) targets the page retrieval considering text similarities computed combining word image clustering and the *tf-idf* weighting. In Section 5.2 we explore the SOM clustering of the page layout for page classification applications. In this approach, we analyze also the use of the tangent distance between vectorial representations of the page layout in order to improve the recognition rate for data-sets containing few labeled pages.

5.1 Document Retrieval

The proposed document retrieval based on word image clustering originates from one classical approach in text-based Information Retrieval: the *vector model* (see [52], Chapter 2). This approach is based on a vectorial description of the document contents where the vector items are related to the occurrences of index terms, usually corresponding to words, in the document. Vector values are weighted to give more importance to most discriminant terms. To this purpose one common approach relies on the well known *tf-idf* weighting scheme. The basic idea is that index terms that are present in many documents of the collection should have a low weight since their presence is not discriminant. With the *tf-idf* approach the weight assigned to the k-th word in the document D_i is computed by:

$$w_{i,k} = f_{i,k} \cdot log\left(\frac{N}{n_k}\right), \tag{8}$$

where $f_{i,k}$ is the frequency of the k-th word in D_i normalized with respect to the maximum word frequency in D_i, N is the total number of documents, and n_k is the number of documents containing the k-th word.

The vector model has been designed to process textual documents where the word identification and clustering (with possible stemming and stopword removal) is quite straightforward. The document retrieval discussed in this section relies on the use of the WordSOM clusters in lieu of ASCII words in the $tf-idf$ weighting. Basically, a document is represented by a vector whose items correspond to the neurons of the six maps computed (Section 4.2). Each vector item contains the number of words in the document that are assigned to the corresponding neuron (i.e. the words having that neuron as BMU). In analogy with the vector model, we apply the $tf-idf$ weighting to this representation. It is worth to remark that the size of the overall vector (on the average 1,800 items) is smaller than the typical size of the dictionaries considered with the vector model (for instance for the well known *Reuters 21578* corpus the dictionary contains around 19,000 terms). After the indexing, a weighted vector is associated to each document.

To perform document retrieval a vector is computed from the query document taking into account the neurons assigned to each word in the page. The similarity is evaluated by comparing the query (q) with each indexed document (d) by using the *cosine of the angle* between the vectors:

$$sim(q,d) = \frac{\sum_{i=0}^{n-1}(q_i \cdot d_i)}{\sqrt{\sum_{i=0}^{n-1} q_i^2} \cdot \sqrt{\sum_{i=0}^{n-1} d_i^2}} \tag{9}$$

To obtain a global ranking of the indexed documents we compute the similarity of all the documents with respect to the query and sort the documents on the basis of the measure computed in Eq. (9).

Rank	Page	Sim.	Most frequent words
1	448	0.1624	*carbonate*, *carbon*, *sulfate*
			(*carbonate*, *coal*, *sulphate*)
2	446	0.1492	*four*
			(*oven*)
3	1007	0.1368	*mouvement*, *cylindre*, *ouvrier*, *roues*
			(*movement*, *cylinder*, *worker*, *wheels*)
4	822	0.1348	–
5	1164	0.1320	similar to 1007
6	1254	0.1313	–
7	455	0.1309	*ammoniaque*, *sulfate*
			(*ammonia*, *sulfate*)

Table 1. Most frequent words in the top ranked pages of the example query for document retrieval

Experimental Results

The experiments described in this section are made on two books (containing 1280 pages) that are part of an encyclopedia addressing machineries and techniques of the industry of the XIXth Century [1]. The images are quite clean and the OCR works well on these documents. The interest for this data-set lies on the homogeneity of the contents of the pages in each chapter, so that the evaluation is simplified.

An accurate evaluation of document retrieval systems requires a judgment of the relevance which is provided by human experts. In our application this information is not available and we evaluated the system in two ways. First, we made a global test by performing several queries for each chapter. Second, we carefully analyzed some queries by checking the contents of the neurons providing the highest contribution to the similarity measure (Eq. 9).

We describe in the following the results obtained with one query page by highlighting the most important neurons for each selected page. The query page (number 452) belongs to chapter 4 *'sodium and potassium'* and describes one specific machinery. Therefore, the page contains several technical terms (e.g. *cylinder*, *wheels*, *oven*, *movement*, *combustion*, *handle*, *heat*, *worker*, *rotation*). This machinery is used for the production of sodium and the following words are contained in the page: *sulfate*, *reaction*, *carbonate*, *chaux* (lime). Table 1 reports the first ranked pages together with the most frequent words contained in the clusters providing the higher contribution to Eq. (9). Pages 448 and 455 belong to the same chapter as the query page. Page 446 describes a machinery very similar to the one addressed in the query page. Also the machineries described at page 1007 and 1164 are very similar to the query, but are designed to work with other materials. Another interesting point is the presence among the top ranked pages of few pages belonging to chapter 4

[1] *Les Merveilles De l'Industrie*: downloaded from the web site of the *National Library of France*.

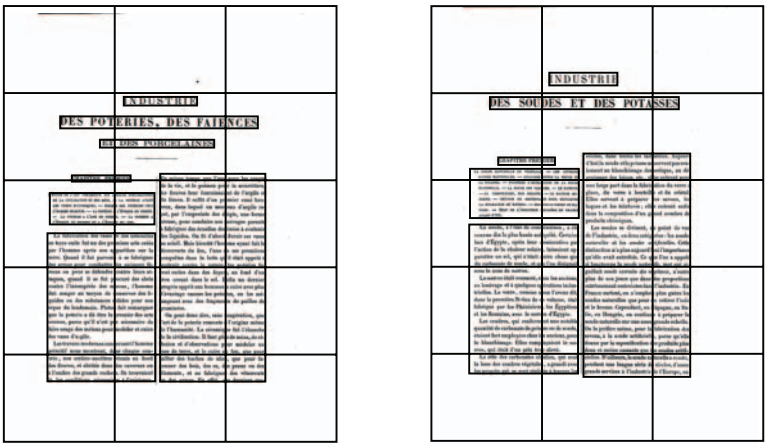

Fig. 6. Two pages of the Issue2 class. We show the regions identified by the layout analysis and the grid overlapped to the pages

(pages 419-480), since the query page contains few references to sodium and potassium.

5.2 Layout Clustering

Another application of SOM clustering is the page layout classification. The approach proposed in this section is particularly useful when dealing with data sets where only few pages have been manually annotated, and therefore the training of a discriminant classifier can be difficult.

Page classification methods generally represent the page layout either with fixed-size feature vectors [53, 54] or with graphical structures (e.g. graphs and trees [55]). In [55] we compared some approaches for SOM-based page clustering relying on the MXY-tree page representation. Graphical representations usually suffer important non-linearities and small transformations (e.g. translations) in the image space can give rise to significant changes in the subsequent representation.

In this chapter we propose the use of the tangent distance for layout clustering. This clustering is subsequently used to perform the page classification. We propose a vectorial page representation that smoothly depends on small transformations in the image space. The latter is a requirement to adopt the tangent distance approach.

A layout analysis tool is used to extract the homogeneous regions in the pages. Each page is then represented with a fixed-size feature vector obtained by computing appropriate features in the cells defined by a regular grid superimposed to the page. Two segmented pages of the same class are shown in Figure 6 together with the overlapping grid. From the figure we can notice

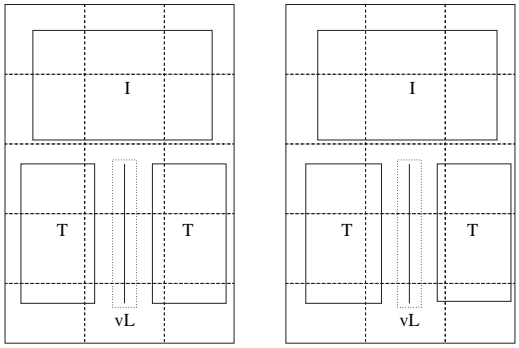

Fig. 7. Simulation of a horizontal displacement of a page. For each page we show the layout regions and the overlapping grid

that the right page has a similar layout but is vertically shifted. For each cell we compute the percentage of its area that is covered by text, image and line regions, respectively. For horizontal and vertical lines we assign a virtual area around each line (see the dotted area around vL in Figure 7) and evaluate the percentage of the cell covered by the union of all the virtual areas. In this way broken lines provide a contribution similar to continuous ones.

Two of the most important transformations that should be addressed in the layout clustering are horizontal and vertical displacements. To simulate these transformations in the tangent distance paradigm we compute, for each training page, the new representations that are obtained by slightly shifting the grid in the horizontal and vertical directions (an example of horizontal displacement is shown in Figure 7). By means of these transformed page representations we can compute a sort of tangent vectors that can be used to train the SOM_TD with the method described in Section 4.1.

Experimental Results

In this section we discuss the performance of the SOM_TD model for page layout clustering. The clustering is afterwards used to perform a layout-based page classification with particular interest on data-sets having a reduced number of labeled training patterns. Two experiments have been performed to analyze the approach under different conditions. In the first experiment we considered a data-set that is typical of digital library repositories that contains a digitized Encyclopedia. The second experiment deals with a collection of commercial invoices.

The Encyclopedia data-set is composed by 6 books containing 4035 pages that have been assigned to more than ten classes[2]. We made our experiments considering the seven most important classes (Figure 8). The SOM training

[2] This data-set is a superset of the one used in Section 4.1.

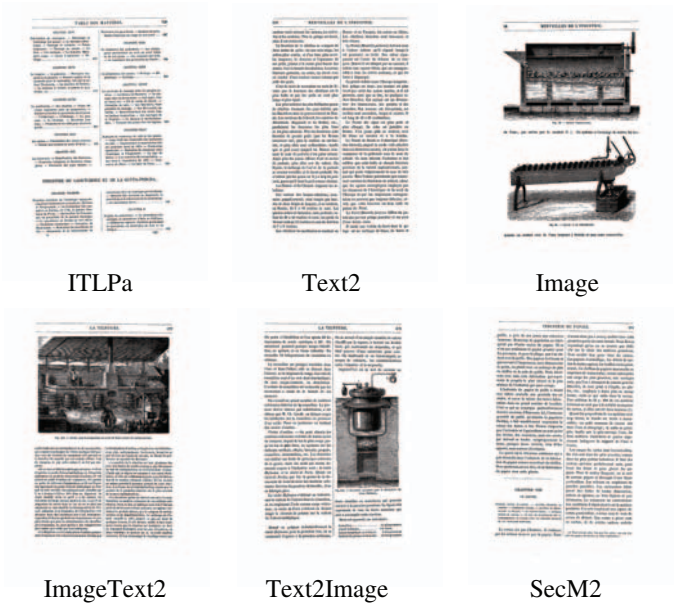

Fig. 8. Examples of pages of the main classes considered in the layout retrieval

10x8		12x10		15x10	
SOM	**SOM_TD**	**SOM**	**SOM_TD**	**SOM**	**SOM_TD**
55.59	67.51	59.91	60.97	70.74	71.41

Table 2. Encyclopedia Data Set: average recognition rate for various sizes of the SOM

has been made with the 617 pages of the first book. Figure 9 shows one SOM trained on this data-set. Some map regions containing similar pages have been identified in order to make easier the visual analysis of the map. After training the SOM map either with the standard or with the SOM_TD algorithm we label each neuron on the basis of the most frequent class among the pages in the training set firing the neuron.

During the test (that is made considering the remaining five books) we classify all the pages in the test set belonging to the nine classes of interest. The class assigned to each page is the label of the closest neuron. The comparison of the two approaches is made computing the recognition rate for each class when using the standard SOM and the SOM_TD. To reduce the variations in the training we used the same starting map for both learning methods. In Table 2 we compare the average recognition rate for the two approaches considering various map sizes. We can notice that in all the cases the tangent map provides better results. Moreover, larger maps allow us to obtain higher recognition rates. To further investigate these results we compare, in

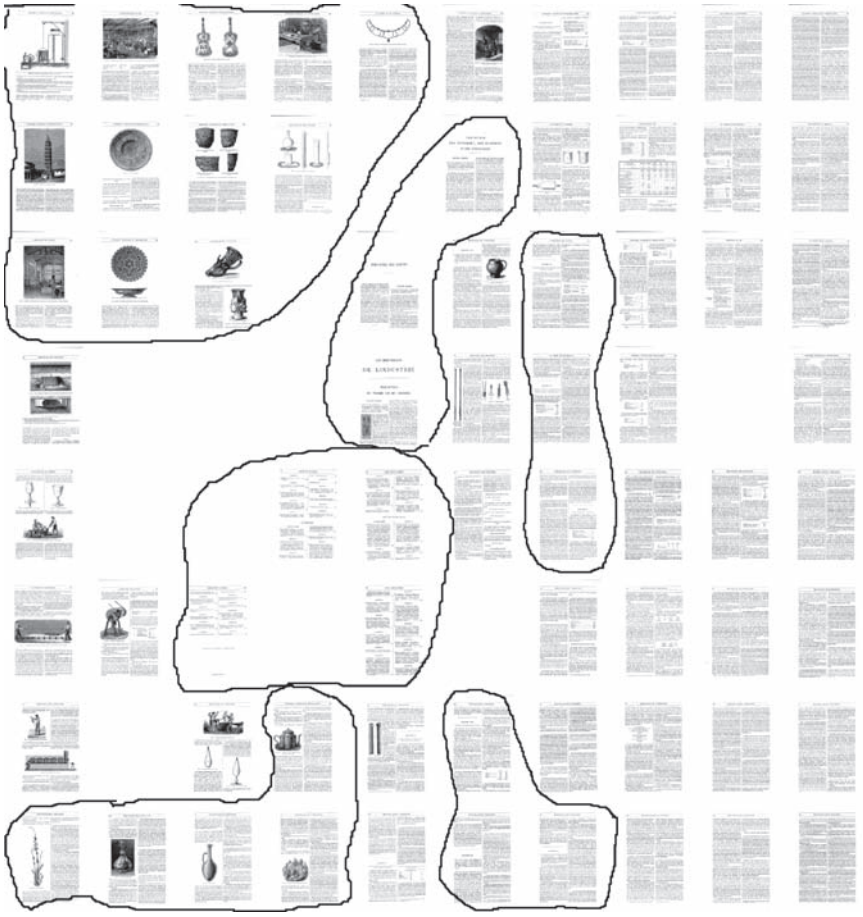

Fig. 9. A graphical representation of the layout SOM computed for the Encyclopedia data-set. Some regions containing similar pages are shown on the map

Table 3, the recognition rate for each class when using the largest map. We can observe that the tangent map provides better results for all the classes with the exception of the classes **Text2Image** and **Text2**. As a matter of fact pages of the two classes are sometimes confused. The reason for this behavior is probably due to the fact that the **Text2** class is the most populated one and has a very simple layout (text on two columns). In this case the use of tangent vectors is not useful since several training patterns already exist, and the addition of local distortions can be confusing for the clustering.

The experiments performed on the invoice data-set have been made with the aim of evaluating the effectiveness of the proposes SOM_TD model when dealing with a large collection of patterns that are labeled only in a small percentage.

	SOM	SOM_TD
ImageText2	81.33	82.14
Text2Image	69.08	67.05
SecM2	36.30	40.84
Image	96.72	96.72
Issue2	84.21	87.50
ITLPa	28.00	32.00
Text2	96.03	93.65

Table 3. Encyclopedia Data Set: average recognition rate of SOM and SOM_TD for the main classes

	12x10	15x10	18x10
SOM	59%	52%	59%
SOM_TD	58%	66%	61%

Table 4. Invoice Data Set: average recognition rate for various sizes of the SOM

The SOM training and labeling has been performed in two steps: first we trained a map with the SOM_TD algorithm considering all the patterns belonging to the training set. In the second step we labeled some neurons in the map on the basis of a majority voting computed with 64 pages belonging to 15 classes. The classification is obtained, as before, assigning each unknown page to the class of the closest neuron. Since a reduced number of labeled samples is available we made the experiments with a leave-one-out approach. From Table 4, containing the average recognition rate for the invoice data-set, we can notice that the best results are obtained with the 15x10 map when using the SOM_TD training algorithm. In this case we needed larger maps with respect to the Encyclopedia data-set since there are more classes anf the pages not belonging to the 15 labeled classes are more than a half of the whole data-set.

6 Conclusions

This chapter addresses the use of unsupervised learning in DIA applications with a special emphasis on SOM-based clustering. Clustering techniques are particularly appropriate in applications where a large number of labeled training samples is not available, or when there is no need to assign patterns into a pre-defined number of classes (for instance in document image retrieval applications). A well known technique for enhancing classifiers trained with a reduced number of training patterns is the tangent distance that we use, in this chapter, for SOM training.

The application of SOM clustering is discussed at three main processing levels: character, word, and layout clustering. In particular, the proposed

SOM_TD approach is evaluated on two applications: the word retrieval based on character clustering and the layout classification based on page clustering. In both cases the experimental results confirm the hint that the tangent distance is particularly suited when dealing with data sets having a small number of labeled training patterns.

References

1. Kohonen, T.: Self-organizing maps. Springer Series in Information Sciences (2001)
2. Marinai, S., Gori, M., Soda, G.: Artificial neural networks for document analysis and recognition. IEEE Transactions on PAMI **27**(1) (2005) 23–35
3. Poggio, T., Girosi, F.: Networks for approximation and learning. Proceedings of the IEEE **78**(9) (1990) 1481–1497
4. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks **2** (1989) 359–366
5. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In Rumelhart, D.E., McClelland, J.L., eds.: Parallel Distributed Processing. Volume 1. MIT Press, Cambridge (1986) 318–362
6. Kohonen, T.: The self-organizing map. Proceedings of the IEEE **78**(9) (1990) 1464–1480
7. Bernard, E., Casasent, D.: Invariance and neural nets. IEEE Transactions on Neural Networks **2**(5) (1991) 498–508
8. Avi-Itzhak, H., Diep, T., Garland, H.: High accuracy optical character recognition using neural networks with centroid dithering. IEEE Transaction on PAMI **17**(2) (1995) 218–224
9. Oliveira, L., Britto, A.S., Sabourin, R.: A synthetic database to assess segmentation algorithms. In: Int'l Conference on Document Analysis and Recognition. (2005) 207–211
10. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11) (1998) 2278–2324
11. Simard, P.Y., LeCun, Y., Denker, J.S.: Memory-based character recognition using a transformation invariant metric. In: Int'l Conference on Pattern Recognition. (1994) 262–267
12. Schwenk, H., Milgram, M.: Transformation invariant autoassociation with application to handwritten character recognition. In: Proc. NIPS. (1996) 991–998
13. Rumelhart, D.E., McClelland, J.L., the PDP Research Group: Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1. MIT Press, Cambridge (1986)
14. Duda, R.O., Hart, P., Stork, D.G.: Pattern Classification. John Wiley & sons (2001)
15. Ahmed, P.: A neural network based dedicated thinning method. PRL **16**(6) (1995) 585 – 590
16. Datta, A., Parui, S.K., Chaudhuri, B.B.: Skeletonization by a topology-adaptive self-organizing neural network. Pattern Recognition **34** (2001) 617–629
17. Sasamura, H., Saito, T.: A simple learning algorithm for growing self-organizing maps and its application to skeletonization. In: Int'l Joint Conference on Neural Networks. Volume 1. (2003) 787–790

18. Palenichka, R.M., Zaremba, M.B.: Multi-scale model-based skeletonization of object shapes using self-organizing maps. In: Int'l Conference on Pattern Recognition. (2002) 143–146
19. Zhou, J., Lopresti, D.: Extracting text from WWW images. In: Int'l Conference on Document Analysis and Recognition. (1997) 248–252
20. Park, S., Yun, I., Lee, S.: Color image segmentation based on 3-D clustering: Morphological approach. Pattern Recognition **31**(8) (1998) 1061–1076
21. Worring, M., Todoran, L.: Segmentation of color documents by line oriented clustering using spatial information. In: Int'l Conference on Document Analysis and Recognition. (1999) 67–70
22. Hu, J., Kashi, R., Wilfong, G.: Document image layout comparison and classification. In: Int'l Conference on Document Analysis and Recognition. (1999) 285–288
23. Suzuki, M., Tamari, F., Fukuda, R., Uchida, S., Kanahori, T.: INFTY: An integrated OCR system for mathematical documents. In: Document Engineering. (2003) 95–104
24. Lu, Y., Tan, C.: Information retrieval in document image databases. IEEE Transactions on Knowledge and Data Discovery **16**(11) (2004) 1398–1410
25. Haffner, P., Bottou, L., Howard, P.G., LeCun, Y.: DjVu: analyzing and compressing scanned documents for Internet distribution. In: Int'l Conference on Document Analysis and Recognition. (1999) 625–628
26. Witten, I.H., Moffat, A., Bell, T.C.: Managing gigabytes: compressing and indexing documents and images. Academic Press (1999)
27. Hobby, J.D., Ho, T.K.: Enhancing degraded document images via bitmap clustering and averaging. In: Int'l Conference on Document Analysis and Recognition. (1997) 394–400
28. Chiang, J.H., Gader, P.: Recognition of handprinted numerals in VISA card application form. MVA **10**(3) (1997) 144–149
29. Reddy, N.S., Nagabhushan, P.: A three-dimensional neural network model for unconstrained handwritten numeral recognition: a new approach. Pattern Recognition **31**(5) (1998) 511–516
30. Dehghan, M., Faez, K., Ahmadi, M.: A hybrid handwritten word recognition using self-organizing feature map, discrete HMM, and evolutionary programming. In: Int'l Joint Conference on Neural Networks. (2000) 515–520
31. Liou, C.Y., Yang, H.C.: Handprinted character recognition based on spatial topology distance measurement. IEEE Transaction on PAMI **18**(9) (1996) 941–945
32. Rahman, A., Fairhurst, M.: A new hybrid approach in combining multiple experts to recognise handwritten numerals. PRL **18**(8) (1997) 781–790
33. Teo, R.Y.M., Shingal, R.: A hybrid classifier for recognizing handwritten numerals. In: Int'l Conference on Document Analysis and Recognition. (1997) 283–287
34. Wang, J., Jean, J.: Resolving multifont character confusion with neural networks. Pattern Recognition **26**(1) (1993) 175–188
35. Su, H., Wang, W., Li, X., Xia, S.: Hierarchical neural network for recognizing hand-written characters in engineering drawings. In: Int'l Conference on Document Analysis and Recognition. (1995) 46–49
36. Song, H.H., Lee, S.W.: A self-organizing neural tree for large-set pattern classification. In: Int'l Conference on Document Analysis and Recognition. (1995) 1111–1114

37. Cho, S.B.: Neural-network classifiers for recognizing totally unconstrained handwritten numerals. IEEE Transactions on Neural Networks **8**(1) (1997) 43–53
38. Kohonen, T., Kaski, S., Lagus, K., Salojarvi, J., Honkela, J., Paatero, V., Saarela, A.: Self organization of a massive document collection. IEEE Transactions on Neural Networks **11**(3) (2000) 574–585
39. O'Neil, P.: An incremental approach to text representation, categorization, and retrieval. In: Int'l Conference on Document Analysis and Recognition. (1997) 714–717
40. Merkl, D.: Text classification with self-organizing maps: some lessons learned. Neurocomputing **21**(1–3) (1998) 61–78
41. Ménier, G., Lorette, G.: Lexical analyzer based on a self-organizing feature map. In: Int'l Conference on Document Analysis and Recognition. (1997) 1067–1071
42. Konig, A.: Interactive visualization and analysis of hierarchical neural projections for data mining. IEEE Transactions on Neural Networks **11**(3) (2000) 615–624
43. Marukawa, K., Hu, T., Fujisawa, H., Shima, Y.: Document retrieval tolerating character recognition errors - evaluation and application. Pattern Recognition **30**(8) (1997) 1361–1371
44. Taghva, K., Borsack, J., Condit, A.: Evaluation of model-based retrieval effectiveness with OCR text. ACM TOIS **14**(1) (1996) 64–93
45. Lopresti, D.P.: Robust retrieval of noisy text. In: Proc. of ADL'96. (1996) 76–85
46. Madhvanath, S., Govindaraju, V.: The role of holistic paradigms in handwritten word recognition. IEEE Transactions on PAMI **23**(2) (2001) 149–164
47. Cesarini, F., Gori, M., Marinai, S., Soda, G.: INFORMys: A flexible invoice-like form reader system. IEEE Transactions on PAMI **20**(7) (1998) 730–745
48. Williams, W., Zalubas, E., Hero, A.: Word spotting in bitmapped fax documents. Information Retrieval **2**(2/3) (2000) 207–226
49. Rath, T.M., Manmatha, R., Lavrenko, V.: A search engine for historical manuscript images. In: ACM SIGIR 04. (2004) 369–376
50. Marinai, S., Marino, E., Soda, G.: Font adaptive word indexing of modern printed documents. IEEE Transactions on PAMI **28**(8) (2006) 1187–1199
51. Marinai, S., Faini, S., Marino, E., Soda, G.: Efficient word retrieval by means of SOM clustering and PCA. In: DAS 2006, Springer Verlag- LNCS 3872 (2006) 336–347
52. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley (1999)
53. J.Hu, R.Kashi, G.Wilfong: Comparision and classification of documents based on layout similarity. Information Retrieval **2**(2/3) (2000) 227–243
54. Tzacheva, A., El-Sonbaty, Y., El-Kwae, E.A.: Document image matching using a maximal grid approach. In: Proceedings of the SPIE Document Recognition and Retrieval IX. (2002) 121–128
55. Marinai, S., Marino, E., Soda, G.: Tree clustering for layout-based document image retrieval. In: Proc. Second Int. Workshop on Document Image Analysis for Libraries. (2006) 243–251