Author(s)

Insert your Booktitle, Subtitle, Edition

SPIN Springer's internal project number, if known

– Monograph –

December 26, 2006

Springer

Please consider the following packages in Tex header:

```
\usepackage[latin1]{inputenc}
\usepackage{amsmath,bm}
\usepackage{tabularx}
\usepackage{multirow}
```

References are in BibTex format, no bibliographystyle has been given, the `splncs` style is used to generate the document.

Contents

1

Structure Extraction in Printed Documents Using Neural Approaches

Abdel Belaïd and Yves Rangoni

University Nancy 2 - LORIA, Campus Scientifique, 615 rue du Jardin Botanique, 54600 Villers-Lès-Nancy, France
{abelaid,rangoni}@loria.fr

1.1 Abstract

This paper addresses the problem of layout and logical structure extraction from image documents by using neural networks (NN). Two approaches are considered: data-driven and model-driven. In the latter, some specific approaches like rule-based or formal grammar are studied, while in the former: approaches rather oriented artificial network are discussed. Our comprehension of these techniques let us think that a hybrid model can be a more appropriate solution to deal with this kind of problem. Based on this standpoint, we proposed a perceptive NN based approach which is even of static topology possesses the characteristics of recurrent and dynamic NN. This model has been applied in document structure extraction and its results exceed those of a static multilayer perceptron.

1.2 Introduction

Automatic structure extraction remains a very challenging problem due to the inherent complexity of the documents. Starting at the pixel level, the gap between physical and logical structure is huge. In fact, it is difficult to model the intermediate steps and the relationships between the extreme structures, to maintain the coherence between steps in the recognition process, and to face to the layout variation and to noise during the processing.

In spite of the numerous researches done in this way, the research investigation is prudent:

1. the recognition has been limited to few structures (less than 10, let say 5 in average), because focused more on editorial documents (i.e. books, articles, reports, etc.) having more standard architecture, often accompanied by a DTD (Document Type Definition) and making the recognition more stereotyped;
2. the recognition methodology was limited to translate the DTD knowledge and its application on the document. We saw appearing some methods such as context-free grammars, tree and graph comparisons, which revealed very limited to face to complex situations.

Certainly, the literature is plentiful of approaches but their application on document analysis is not straightforward and their advantages often equal their drawbacks. Two main types of approaches categorize these approaches: information manipulation aspect and data perception aspect.

Considering the information manipulation aspect, two sub-categories exist:

- model driven systems where the rule-based approaches can be assigned. They use and formalize very well the knowledge, are precise and fast but are dependent on the expert to guide their action, not generic and reputed sensitive to variation and noise;
- data driven systems, starting from real information. Their classes should represent very well the structure elements, but data description is not easy and the convergence is not assured. However, contrary to the former, they remain very general and flexible as their adaptation to new documents is easier.

Considering the perception aspect, here also we can consider two points of view:

- global to local which is often assimilated to top-down approach. The process is based on a segmentation refinement: here the progress seems to be made continuously and safely but if an error is introduced in the beginning, it remains during all the process.
- local to global or bottom-up approaches. These labeling-based methods go from fine to coarse building progressively the context. Here a lot of useless things have to be extracted and the more things are added the more their management has to be done.

As it is indicated below, all the methods investigated in the literature present some limitations. Hence, the solution that seems to be appropriate for document structure analysis is a hybrid approach in the sense where it mixes both aspects: information and perceptual points of view.

This paper is organized as follows: section **??** will discuss about NN in DAR and specially in physical tasks. The section **??** will be more focused on NN based solution for structural pattern recognition such as logical structure

extraction. Finnaly, section **??** will present some outlines about NN in logical structure analysis.

1.3 Neural networks in document analysis and recognition

1.3.1 Physical or geometrical layout analysis

In Document Analysis and Recognition (DAR), NN have been devoted mainly to preprocessing operations or to global recognition of small patterns as isolated characters. As detailed by Marinai et al. in [**?**], NN operations in DAR include binarization, noise reduction, skew detection, and character thinning. The MLP, for example, is used by [**?**] to binarize image for character segmentation. After a histogram based segmentation phase, the authors feed MLP with pixel and statistical values on a 5×5 moving window to improve the segmentation. In [**?**], a Self Organizing Map and an MLP is applied on the image to extract its most representative grey levels or colors. Another use of NN is noise elimination such as in [**?**] by operating a morphological filtering, filling operations, Kalman filtering, and line following methods.

For character recognition, various NN models dealing with printed or handwritten have been experimented by the researchers. The majority of these models proceed directly on the images. Their inputs are often composed of the image pixel values but can also be high level features. LeCun et al. [**?**] propose a survey of various NN models dealing with handwritten words. Convolutional NN are used by [**?**] for handwritten digits recognition. In addition, the authors propose a network topology adaptation driven by a SOM to handle all the possible deformation of the rejected patterns. Garris et al. [**?**] use an enhanced MLP for the same problem, enhancements are focused on neuron activations functions, regularization and Boltzmann pruning.

Hence, we can already show across these examples that the NN are capable, in spite of their rigid topology, of a certain suppleness to apprehend the local aspect of recognition.

1.3.2 Logical structure analysis

There are few works on logical structure recognition using NN. Indeed most of the approaches are model-driven. The model contains the description of the physical elements of the document and the labels associated. The recognition procedure consists in locating the same physical entities in the document mentioned by the models and to assign them the same labels.

Usually, these models are either trees or grammar rules. In both cases, a syntactical analysis procedure is employed to perform this labeling [**?**]. For example, Brugger et al. [**?**] use a generalized n-gram (with n=3) to represent geometrical relationship between the text blocs, then an optimization method

to match the current input with a global model or a sub-tree of this model. Hu et al. [**?**] use dynamic parsing and fuzzy logic to be more flexible when analyzing the logical structure. Niyogi et al. [**?**] use a rule-based system with a top-down backward-chaining strategy. Their system "Delos" handles about 160 rules in three levels for classification, reading order and logical structure analysis.

Although this methodology seems natural as it transcribes a known structure hierarchy of the document and works very well for simple documents, its application on more complex document becomes quickly difficult and source of a lot of errors. In fact, the use of deterministic models fails because of absence of suppleness in the application of the rules. Furthermore, these models are often given by hand, leading the operator to select and tune manually a lot of parameters. This explains the limits of such models when applied on real images. The structure is often complex and does not fit very well with the general model and where the inherent noise of the input image can somewhere introduce some handicaps in the interpretation of the elements or their frontiers.

To face this problem, a data-driven oriented method seems more appropriated. NN based solution will prevent drawbacks of structural based method but the knowledge must be integrated. Indeed as mentioned in [**?**], the classical use of MLP is not sufficient to tackle the problem. Most of the contributions do not work on the model but on the use fashion of the MLP to resolve the problem. The idea is to use a model which is not based on MLP but which can integrate the structural aspect of the problem. The solution seems coming from NN in general because one learns from examples, and NN are robust faced to noise and have a generalization capacity.

Two types of NN can be considered:

1. static NN (i.e. with MLP configurations) can adapt to structured patterns by cleverly integrating the structure in the topology as made by [**?**];
2. dynamic NN by transforming the temporal chain in structured version as in [**?**, **?**].

These two structures will be described in the following.

1.4 Neural networks for structured patterns

Neural networks are suitable to handle classification problems with static information. For several applications including logical structure analysis, the patterns to deal with are in a structured domain. NN are rather designed to classify unstructured patterns and cannot deal directly with tree or graph structures. However we can find models which can take into account the structured patterns either in a dynamic or a static version.

1.4.1 Static networks

The most known from them is the MLP because it is the easiest to perform and its training algorithm is well known and experimented with success on different kinds of data.

As mentioned in section **??**, its use is generally devoted to physical element recognition where there is no or few structure to interpret. All researches done on this kind of networks do not work on the model topology but more on the way MLP is applied to the specific task.

1.4.2 Dynamic networks

In order to take into account the temporal dimension of some real-world problems, dynamic networks can be a good alternative to static one.
The Time Delay NN (TDNN) is a straightforward solution that unfolds the time sequence onto several static models through a Tapped Delay Line (TDL) [**?**]. The same approach can be done with the RBFN (Radial Basis Function Network) to take into account the temporal dimension [**?**].
Feedback dynamic methods as recurrent networks integrate feedback contrary to feedforward systems. The learning is recursive and consequently more complex to undertake. The output Feedback based systems use the network outputs in a second TDL besides the classical one's as in TDNN [**?**].
State feedback methods use feedback connections between neurons are introduced: each neuron contributes to all components of the state vector.
Time Hopfield Networks (THN) [**?**] are mono-layer networks in which all the possible interconnections are used. The Continuous THN (CTHN) is well known as it can handle oscillations or even chaotic phenomenon. The Discrete THN (DTHN) is similar to the previous one's but here the activation function is hard-limiter and not a sigmoid.
Continuous Time Recurrent Neural Network (TRNN) [**?**] is quite similar to CTHN: there is one layer of fully connected neurons, the difference is in the differential equation managing the dynamic process. The same analogy is done for the Discrete Time Recurrent Neural Networks (DTRNN) with its hard-limiter function [**?**]. The DTRNN has the property to simulate deterministic finite automata. In such NN, the training stage is more complicated and two main solutions can be seen in the literature. The first totally converts the network on a feedforward version by unfolding the entire network over the time. The second method consists in the use recursive versions of the gradient descend.

These dynamic networks have been developed to process sequences of patterns but adaptations to structured patterns can be found in the literature

Küchler and Goller [**?**] propose an approach to classify structured patterns. The patterns considered are those possibly represented by a Direct Acyclic Graph (DAG) or by a Rooted LDAG (i.e. a graph with only one root node, i.e. one node with indegree zero). The NN topology, in a static view, corresponds

to the folding of the DAG in a feedforward MLP. The first layers compute the folding part (i.e. inputs through DAG representation) and the following layers constitute the transformation part (Fig.**??**).

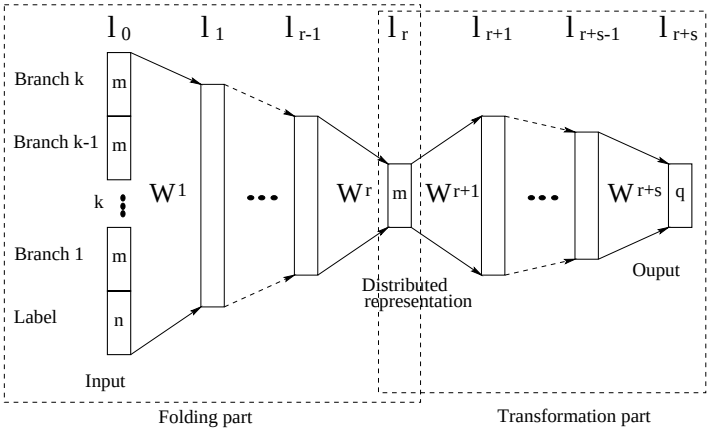

Fig. 1.1. Kuchler et al. generic folding architecture

The input contains the vertex labels for distributed representation of DAG. The last layer corresponds to the task specific output. The network dynamics are defined as follows:

$$o_j^{(l+1)}(t) = f\left(\sum_i o_i^{(l)}(t) w_{ij}^{(l+1)} + \theta_j^{(l+1)}\right) \tag{1.1}$$

where $o_i^{(l)}(t)$ is the output of neuron i in the layer l at recursion stage t, $\theta_i^{(l)}$ is the bias associated with neuron i at layer l, $w_{ij}^{(l+1)}$ the weight of the connection between neuron i in layer l and neuron j in layer $l+1$ and f the sigmoid function.

The authors use a modified version of the Back-Propagation Through Time algorithm which the structure of a labeled DAG is incorporated in the error measurement

$$E = \sum_{i=1}^{p} \sum_{j=0}^{q-1} \frac{1}{2} \left([t_i]_j - o_j^{(r+s)}(root(s_i))\right)^2 \tag{1.2}$$

where *root* denotes the function mapping structures to their root nodes, s_i are in the general symbolic domain and t_i define by $\Xi(s_i) = t_i$ with Ξ the function to be approximated.

Thanks to a special gradient descent technique: Back-Propagation Through Structure (BPTS), the network can be trained. The experimentation has been

done also on 2-classes classification problems on logical terms. The results are very promising: 99% for the training and 98% for the test.

Sperduti et al. [**?**] propose another dynamic NN extended to structural patterns. The main idea is to generalize a recurrent neuron in a "Generalized Recursive Neuron" (GNR). The approach is different from the standard fashion which focuses on the tree-structure encoding in a fixed input vector. The GRN considers the outputs of the unit for all the vertices which are pointed by the current input vertex.

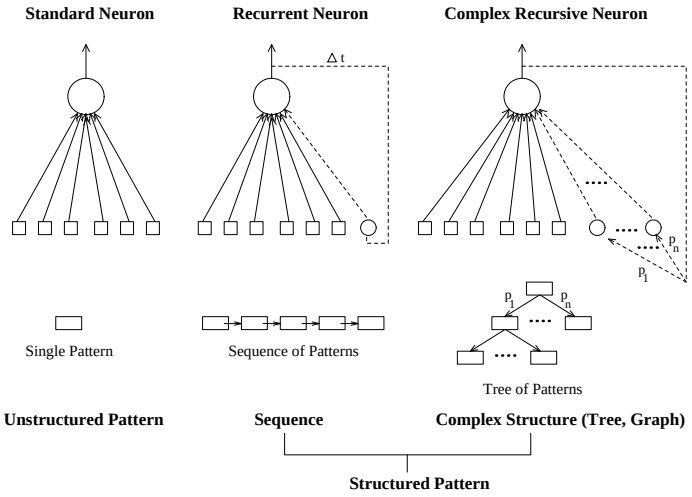

Fig. 1.2. Neurons models for different input domains

Figure **??** shows the standard models for unstructured and sequence of patterns, on the right side, the presented configuration can represent any graph or tree structure thanks to the proposed GRN.

Usually, in a standard neuron the output is given by:

$$o^{(s)} = f\left(\sum_i w_i I_i\right) \tag{1.3}$$

where f is non-linear function such as the sigmoid, I the input vector and w the weight vector.

In the recurrent version, the output depends on time:

$$o^{(r)}(t) = f\left(\sum_i w_i I_i(t) + w_s o^{(r)}(t-1))\right) \tag{1.4}$$

where $o^{(r)}(t-1)$ is the previous output at time $t-1$ that is weighted by w_s and added to activation formulae. In the GRN the output $o^{(g)}(x)$ depends on

a vertex in the graph and computed recursively on the output performed for all the vertices pointed by it. The output is given by:

$$o^{(g)}(x) = f\left(\sum_{i}^{N_L} w_i l_i + \sum_{j=1}^{\text{out\_degree}_X(x)} \hat{w}_j o^{(g)}(\text{out}_X(x,j))\right) \tag{1.5}$$

where x is a vertex of a graph X, N_L the unit number encoding the label l attached to the current input x, $\hat{w}_j$ the weights on the recursive connections and $out_X(x,j)$ the out nodes of the graph X attached to the node x.

The graph is encoded to fit with the GRN representation as illustrated in figure **??**.

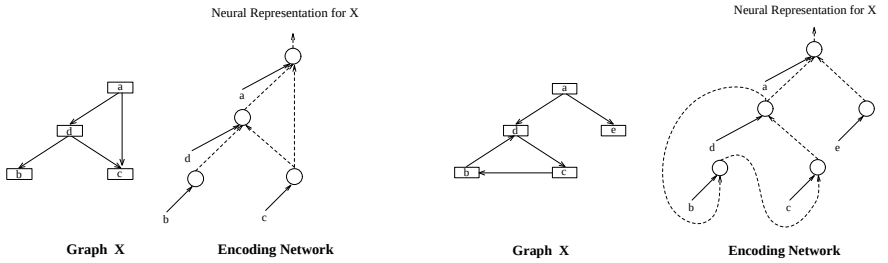

Fig. 1.3. On the left side, encoding for acyclic graph. On the right side, the encoding network for a cyclic graph

The authors have extended five supervised algorithms for NN to handle the GRN: backpropagation through structure, real-time recurrent learning, LRAAM-based networks and simple recurrent networks, cascade-correlation for structures, and neural trees.

For example the Backpropagation Through Structure (BPTS) is simply as in [**?**] an expression of the backpropagation through time. The trick consists in unfolding through the time the recurrent network in an equivalent and fully feedforward network. As a consequence, the transformed network can be trained with backpropagation algorithm. For the GRN, the network is decomposed into two parts: an encoding function Ψ and a classification function Φ such as

$$o(X) = \Phi(\Psi(X)) \tag{1.6}$$

Using standard backpropagation learning, the weights are modified unsing **??** and **??**:

$$\Delta \mathrm{W}_{\Phi} = -\eta \frac{\partial \mathrm{Error}(\Phi(y))}{\partial \mathrm{W}\Phi} \tag{1.7}$$

$$\Delta \mathrm{W}_{\Psi} = -\eta \frac{\partial \mathrm{Error}(\Psi(y))}{\partial \mathrm{y}} \frac{\partial y}{\partial \mathrm{W}_{\Psi}} \tag{1.8}$$

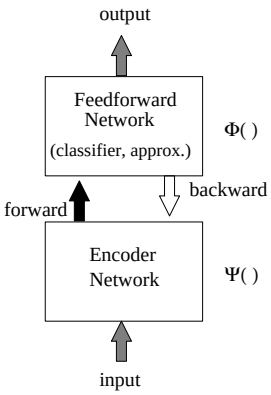

Fig. 1.4. Encoding part forward the structures to the classifier, the classifier returns the deltas used by the encoder to adapt its weights

Two cases must be treated separately in the case of a DAG and graphs with cycles. With DAG, Küchler et al. [**?**] algorithm can be used. The training is computed by backpropagation of the error from the feedforward network through the encoding network of each structure. For cyclic graphs, the recurrent backpropagation must be considered.

Real-Time Recurrent Learning and can also be extended. For DAG the extension does not present particular problems, the cyclic graphs are more difficult to extend and require different situation according to global cycle presence. Thanks to the Strongly Connected Component and Component Graph notion, the cyclic graphs can be considered as many acyclic graphs and solved more easily.

Labeling Recursive Auto Associative Memory (LRAAM) [**?**], another model to represent labeled structures, is trained by a combination of a supervised method and unsupervised one. For the structured pattern recognition, Sperduti uses this LRAAM to produce a compressed representation of the structure then he uses an MLP to carry out the classification.

GRN can be also extended to the cascade-correlation algorithm developed by Fahlmane and Lebiere [**?**]. This model grows a standard NN by using an incremental approach for classification of unstructured patterns. The starting network $\mathcal{N}_0$ is a network with no hidden nodes trained by using LMS. If $\mathcal{N}_0$ cannot resolve the problem, a hidden unit u_1 is added so that the correlation between the output of the unit and the residual error of the network $\mathcal{N}_0$ is maximized. The weights of u1 are frozen and the remaining weights are retained. If the retained network $\mathcal{N}_1$ cannot solve the problem, the network is further grown by new hidden units which are connected (with frozen weights) with all the inputs and ancient hidden units. The resulting network is a cascade of nodes. Sperduti et al. extend the output of the k^{th} to GRN using:

$$
\begin{aligned}
o^{(k)}(x) &= f(\alpha + \beta + \gamma) \\
\alpha &= \sum_{i}^{N_L} w_i^{(k)} l_i \\
\beta &= \sum_{v=1}^{k} \sum_{j=1}^{\text{out\_degree}_X(x)} \hat{w}_{(v,j)}^{(k)} o^{(v)}(\text{out}_X(x,j)) \\
\gamma &= \sum_{q=1}^{k-1} \bar{w}_q^{(k)} o^{(q)}(x)
\end{aligned}
\tag{1.9}
$$

where $w_{(v,j)}$ is the weight of the k^{th} hidden unit associated with the output of the v^{th} hidden unit computed on the j^{th} component pointed by x. $\bar{w}_q^{(k)}$ is the weight of the connection from q^{th} hidden unit and the k^{th} hidden unit. Learning is performed as in standard cascade-correlation with the difference that the equations are recurrent on the structures.

GRN can also be adapted to neural tree. The advantage of this kind of model is to build the structure on the fly and not linked to a static structure as in a feedforward NN. New classes are learnt incrementally with a supervised or unsupervised training. The extension of this network to a structured version is done by analogy: each discriminator associated with each node of the tree is replaced by a generalized recursive discriminator.

The experiments about GRN have been made on several classification tasks. The data are randomly generated. For small size structures (tree depth between 3 and 6) the results obtained on classification problems are nearly perfects for the training (near 100%) and very good for the test (average of 95% and sometimes 100% with a good choice of hidden units and learning parameters).

There is more and more works about dynamic networks, although they are not oriented directly towards logical structure extraction, it seems that the previous contributions can be easily extended to this kind of application. However, these recurrent techniques present some drawbacks compared to static NN:

- they are time and memory consuming;
- the convergence is more difficult to reach as there are more local minima;
- the convergence is slower, decreasing the training step make the training more and more slow;
- there are more numerical errors that have serious repercussion on the error;
- the gradient explosion occurs quickly on long sequences. More the sequence is long and more the error can be huge.

On top of that, the presented dynamic neuronal methods can deal with logical structure recognition but are not sufficient. In addition to the inherent limitations structures to be performed need to be known and fixed through-

out the training and recognition that it is not necessarily true in real world applications.

1.4.3 Perceptive structured neural network

As seen in previous section, dynamic NN can be extended to deal with structured patterns. The well known static NN such as MLP can be also improved to handle structured patterns.

In [**?**], Côté et al. propose a perceptual model for handwritten words recognition. The proposed method is based on McClelland and Rumelhart reading mode [**?**]. Two questions are explored: what kinds of features are detected and how the information concerning the meaning of a word is accessed. The key of the author works is to integrate a knowledge representation in NN. Indeed, trying to use a standard network with distributed representation, such as the MLP, seems to cannot deal correctly with handwritten recognition. That is why in [**?**] a network with local representation is opted being the kernel of their approach. The Interactive Activation Model of [**?**] is a neural network with local knowledge representation, parallel processing of information, and gradual propagation of activation between adjacent levels of neurons. Original activation is given by

$$\begin{aligned} A_i(t+\delta t) &= A_i(t) - \theta_i(A_i(t) - r_i) + E_i(t) \\ E_i(t) &= \begin{cases} n_i(t)(M - A_i(t)) & \text{if } n_i(t) > 0 \\ n_i(t)(A_i(t) - m) & \text{if } n_i(t) < 0 \end{cases} \\ n_i(t) &= \sum_j (\alpha_{ij} - \beta_{ij}) a_j(t) \end{aligned} \tag{1.10}$$

where θ_i is a decreasing constant, r_i activation threshold, $E_i(t)$ the neighborhood contribution, M and m superior and lower activation bounds, α_{ij} and β_{ij} the positive and negative stimulation from j to i, and $a_j(t)$ the activation of node j

The recognition is performed trough several bottom-up and top-down processes. The physical features extracted from the image are specific to the problem: primary (e.g. ascender, descender) secondary (e.g. loop, bar) and face-up/face-down valley (e.g. connected components of the background between the lower and upper contours of the word. The architecture of the system is enough general to handle hierarchical organized interpretation. The authors have chosen three levels of neurons: feature, letter, and word (Fig.**??**).

The connections between adjacent levels are excitatory and bi-directional. The connections are only bottom-up between the feature letter and the letter level. The weights are determined according to a priori knowledge. Thanks to an active and passive neuron system, the network can reach the solution after several cycles (until saturation) of bottom-up and top-down processes called perceptual cycles. The system generate hypothesis, validate them and

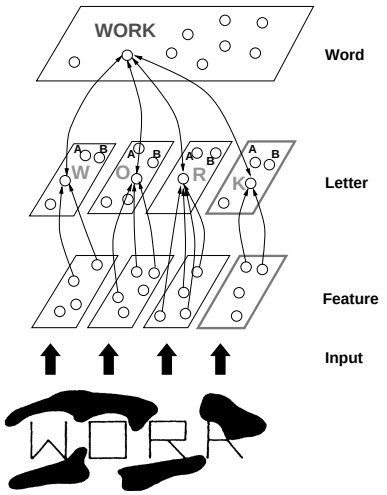

Fig. 1.5. Côté et al. hierarchically organized NN model

possibly insert letter candidate in the right place thanks to already validated letters (Fig.**??**).

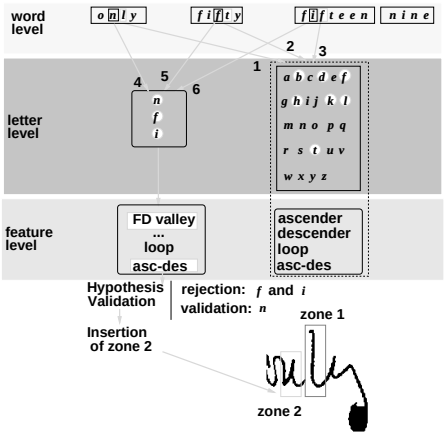

Fig. 1.6. Top-Down process: feedback and insertion

Experiments have been made on CENPARMI database (French and English handwritten cheques), 184 pattern for training and 2929 for testing achieve form 85.3% for word length 3 up to 100% with word length 9.

In [**?**], Maddouri et al. propose an extension of the Perceptro model [**?**]. They use a geometrical correction method to improve the performances of

their Arabic handwritten word recognition system. The recognition is proceeded by cycles of global and local observations. The global observations try to detect apparent features of the words. They create hypothesis on the word label. To carry out the recognition from different kind of information, a normalization stage is done on the word edges to improve the local observations. Indeed, contrary to printed words or characters, the handwritten text needs a powerful normalization stage to handle the variability in position, size, rotation, slant, and distortion. The authors have chosen a Fourrier based solution to eliminate this variability. The whole recognition process is summarized in Figure **??**.

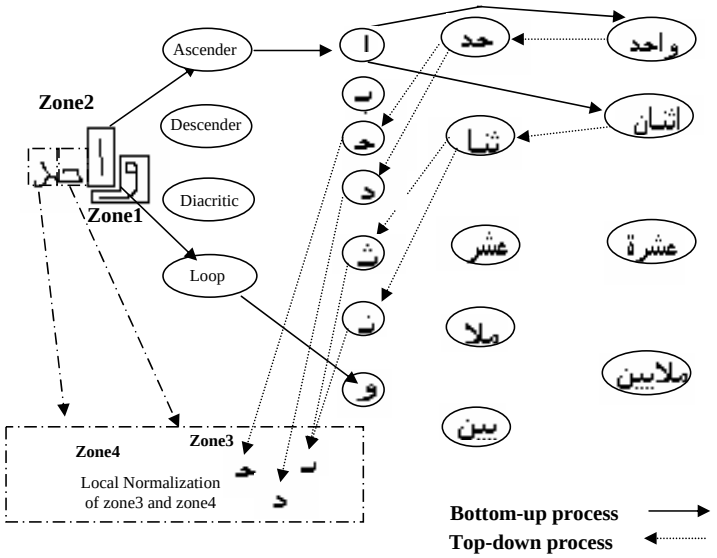

Fig. 1.7. Local normalization and global recognition

The top-down and bottom-up cycles are proceeded thanks to the TNN model (right part of the schema), the local observation comes from the normalization of feature such as ascender, descender, diacritic, and loop (left part of Fig.**??**).

The normalization is performed on the boundary of the word: a detection of the contour is done firstly, then a Freeman chain code is generated. The next step consists in computing Fourrier coefficient of the chain-encoded contour and finally, the coefficients are normalized to cope with variability. To obtain the final normalized character, a reverse Fourrier transformation is applied to the latest normalized coefficient. The reader can refer to [**?**] to see how the boundary normalization is carried out. When the word is normalized, a metric distance is used to evaluate the difference between the current word and printed references.

In [**?**], Rangoni et al. propose a quite similar NN for logical structure recognition in document images. The hierarchical organized interpretation is kept and transposed to handle editorial documents. Each neuron corresponds to an interpretable concept and is attached to an element of the logical structure. Excluding the first layer composed of input physical features, the followingfne7 layers unfold the interpretation by introducing fine concepts in the first layers and general concepts in the latest layers (Fig.**??**). If a DTD is present, it can be helpful to set the neurons: the hierarchy included in the DTD can be unfolded to form the layers and the neurons. Contrary to other models [**?**, **?**] the network is fully connected and the neurons can be inhibitors. As the relations between the layers are not straightforward, a training phase similar to MLP is proceeded to set all the weights.

In the backpropagation algorithm, the error $E_p(w)$ between the desired output d_q and the computed output $o_{L,q}$ is minimized for each pattern p

$$\begin{aligned} E_p(w) &= \frac{1}{2}\sum_{q=1}^{N_L}(o_{L,q}(x_p) - d_q(x_p))^2 \\ o_{l,j} &= f\left(\sum_{i=0}^{N_{l-1}} w_{l,j,i}o_{l-1,i}\right) \end{aligned} \tag{1.11}$$

As a consequence, the weight between the unit i in layer l and unit j in layer $l+1$ is modified as follows

$$w_{l,i,j} \rightarrow w_{l,i,j} - \mu\sum_{p=1}^{P}\frac{\partial E_p(w)}{\partial o_{l,j}}f'\left(\sum_{m=0}^{N_{l-1}} w_{l,j,m}o_{l-1,m}\right)o_{l-1,i} \tag{1.12}$$

In case of [**?**], all the neurons carry interpretable concepts and the desired output is known for all the unit. So, the partial term is given by:

$$\forall l, \frac{\partial E_p(w)}{\partial o_{l,j}} = o_{l,j}(x_p) - d_j(x_p) \tag{1.13}$$

and the network can be trained as a cascade of mono-layer perceptrons.

The model is on the one hand data-driven thanks to the training stage and the other hand the network is model-driven due to the integration of knowledge inside the topology. This kind of NN is called Transparent Neural Network (TNN) in opposition with the "blackbox" aspect of MLP. For document logical layout analysis, it will be named Perceptive Structured NN (PSNN).

The aim of the latest layers is to bring context during the perceptive cycles as the previous authors used these to simulate the word superiority on letters. As the network is feedforward, the learning of the network is the same as an MLP but here the training is done separately between each consecutive pair of layers because all the desired outputs are known.

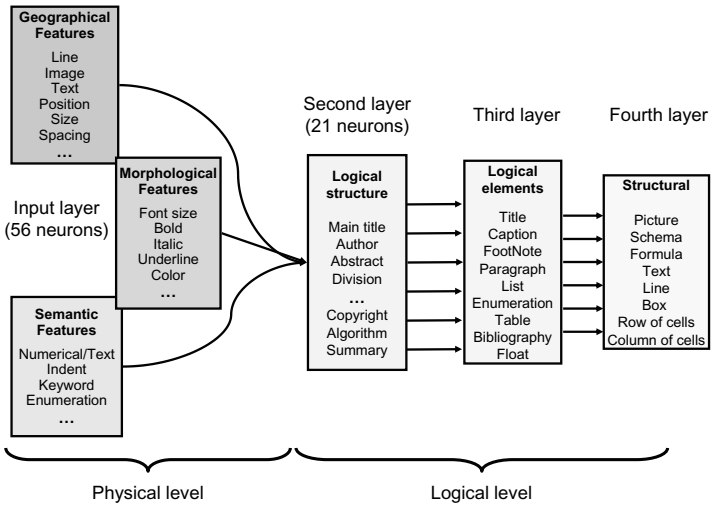

Fig. 1.8. Topology for Scientific Articles

During the recognition step, the network is used as an MLP but after each propagation, the outputs are analyzed. If the output vector is close to a basis vector (**??** & **??**) the pattern is considered classified, otherwise the following layers are taking into account to bring context. $M(O)$ give a vector with at least one component with high value, $\Gamma(O)$ give a vector where one component has a value very high compared to other components.

$$M(O) = \|O\|_\infty > \varepsilon \qquad \text{with} \qquad 0 \ll \varepsilon < 1 \tag{1.14}$$

$$\Gamma(O) = \frac{n((\sum O_i)^2 - \sum O_i^2)}{(n-1)(\sum O_i)^2} < \eta \quad \text{with} \quad 0 < \eta \ll 1 \tag{1.15}$$

As these layers contain more global information, they are more robust and accurate. They are used to generate hypothesis on the pattern. The context manages the correction of the input feature. Once a label is supposed to be the good one, the input vector is corrected according to this hypothesis and according to the knowledge extracted from the training database. Indeed, several representative samples are extracted from database and are matched with the current input. The input is corrected to be close to a representative sample and another perceptive cycle is completed and so on until no ambiguities persist (Fig.**??**).

There are several methods to determine these representative samples, unfortunately there is no exact solution. Some approaches have been investigated. Methods using optimization produce mathematically perfect sample but they do not correspond to real-world interpretable solutions. Methods that are more straightforward can produce appropriate samples: mean sample

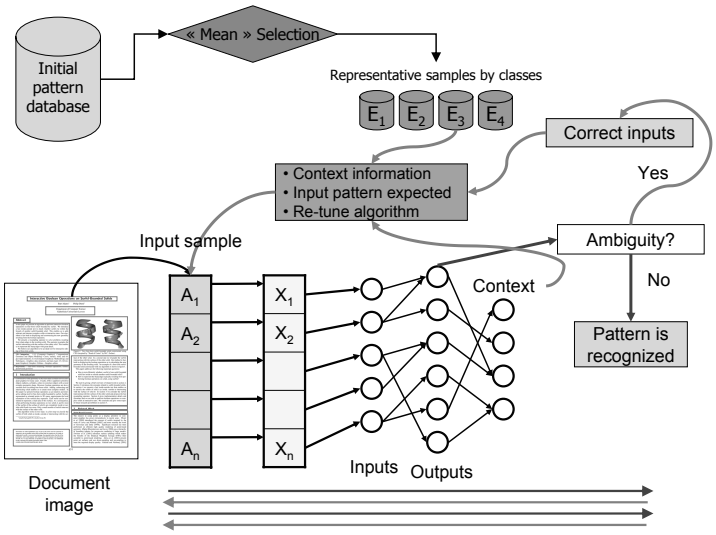

Fig. 1.9. Recognition in Perceptive Structured NN

for only one representative or a k-NN for select several samples per class. Others methods can be performed during the training stage: In [**?**], a new learning method is presented which can produce from a very few sample number of the global database an MLP almost as efficient as trained on the whole database. The subset arisen from the algorithm will provide the representative samples.

The perceptive cycles in this PSNN allow bottom-up and top-down resolution and refine the recognition. However, if too cycles have to be done, the task could be very time consuming because a lot of physical extraction must be completed. On top of that, some of the inputs are high-level and slow down the logical structure recognition. In order to face this problem, a human based approach has been used to trim down the extractions. To simulate global and local vision, the input features are partitioned in cluster thanks to a data categorization. Instead of feeding the network with the whole features for each cycle, the features are given progressively during the recognition and only if the pattern is too ambiguous (Fig.**??**).

Subsets of feature are computed according to their extraction time and their predictive capacities. The first criterion is trivial as the extraction can be timed by experiments or by analyzing the algorithm complexity. Evaluating the predictive power to make group of feature is more complicated as there is no optimal solution to do this. The literature proposes two main family approaches: filter based method and wrapper method [**?**]. The filter methods only use sample database to score the feature, they are fast to evaluate the feature separately but do not produce good groups. On the other side, wrapper methods consider always the variables but they need the classifier to produce the groups. The method exposed in [**?**] is based on a filter approach but can

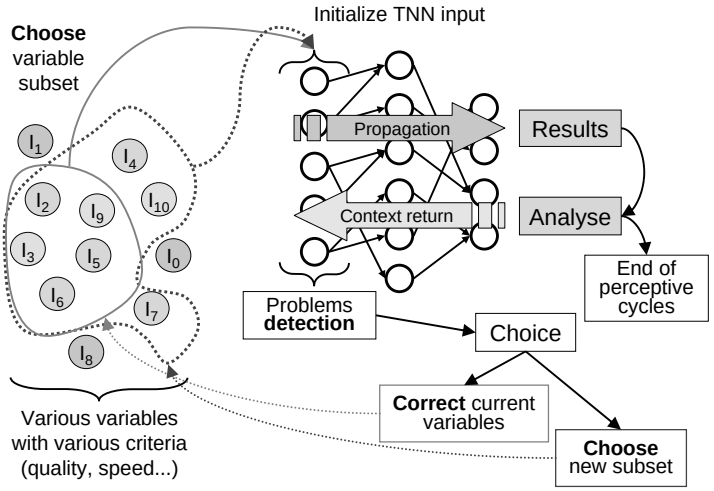

Fig. 1.10. Perceptive cycles: propagation, analyze, context return, correction, input feature selection

compute groups at the same time with ordered predictive power and with the less redundancy inside each group (Fig.**??**).

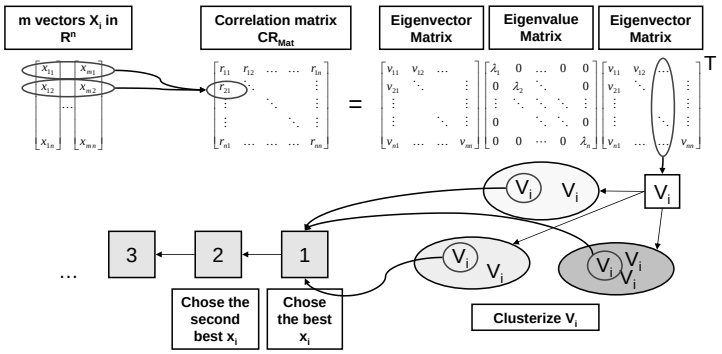

Fig. 1.11. Input feature clustering

By combining the input feature correction and selection, the PSNN is able to adapt the computation amount according to pattern complexity without adding too much processing time.

The system has been testing on scientific articles. After four perceptive cycles, the recognition rate raise 91.7% which is 10 points better than a classical MLP (Tab.**??**).

Classes	MLP	PSNN			
		C_1	C_2	C_3	C_4
Whole	81.6%	45.2	78.9	90.2	91.7%
Best	86.9%	66.7	85.3	85.3	99.3%
Worst	0.0%	0.0	0.0	4.0	28.6%
Time	1	0.7	1.45	1.85	2.40

Table 1.1. Logical structure classification for MLP and for PSNN

1.5 Perspectives

Although able to deal with structured patterns, dynamic NN are not still used for logical layout analysis. On the other hand static networks have very few contribution compared to pure model-driven approaches. All the works presented in the section 4 show how to extend classical models to deal with such a problem. The neuronal approach is accessible and can be as competitive as grammar or rule based systems. It is obvious that, as mentioned in Nagy et al. [**?**], domain specific knowledge appears essential for document interpretation.

The PSNN can be improved in different way: the data-driven can be more developed by introducing hidden layers between each layer of interpretable concepts. The "transparency" property will be lost but the system will be definitely more accurate and greater generalization capacities.

Another way could be integrate transparency in a dynamic network or adding dynamic properties to PSNN. A simply output feedback based PSNN will have more feedback information when bringing the context. On top of that, the context will be taking into account not only during the recognition but also during the training stage.