Learning Efficiently with Neural Networks: A Theoretical Comparison between Structured and Flat Representations

Marco Gori[1] and **Paolo Frasconi**[2] and **Alessandro Sperduti**[3]

Abstract. We are interested in the relationship between learning efficiency and representation in the case of supervised neural networks for pattern classification trained by continuous error minimization techniques, such as gradient descent. In particular, we focus our attention on a recently introduced architecture called recursive neural network (RNN) which is able to learn class membership of patterns represented as labeled directed ordered acyclic graphs (DOAG). RNNs offer several benefits compared to feedforward and recurrent networks for sequences. However, how RNNs compare to these models in terms of learning efficiency still needs investigation. In this paper we give a theoretical answer by giving a set of results concerning the shape of the error surface and critically discussing the implications of these results on the relative difficulty of learning with different data representations. The message of this paper is that, whenever structured representations are available, they should be preferred to "flat" (array based) representations because they are likely to simplify learning in terms of time complexity.

1 Introduction

Recursive neural networks [7, 15, 5] have been recently proposed as a novel connectionist approach for supervised learning in domains in which a significant amount of information is encoded by binary relationships between atomic entities. In these domains, instances are represented as labeled graphs, as opposite to vector-based (or sequence-based) data representations which are much more commonly used in conjunction with connectionist models such as feedforward networks. Interesting applications of these structured connectionist models been found in chemistry, software engineering, automated reasoning, and pattern recognition. There are several reasons for preferring structured to flat representations, the most obvious being the natural way of dealing with variable size patterns and the possibility of preserving information encoded in the relationship between basic entities.

Despite of their attractiveness, little theoretical results have been proved concerning the actual effectiveness of neural networks for data structures. In this paper we are specifically interested in the efficiency of learning. As it turns out, the computational efficiency of a learning process can be significantly affected by the representation used by the learner. There are well known simple examples of this phenomenon. For example, in the PAC learning model, learning 3-term DNF is intractable. However, if the learner is allowed to output a more expressive hypothesis (such as 3-CNF formulae) then the learning task becomes polynomial [11]. The question addressed in this paper is how structured representations affect learning efficiency and whether there is theoretical support to the claim that learning directly in the graphical domain is likely to outperform learning approaches in which the graph is reduced to a flat vector or to a sequence. We give a set of results concerning the shape of the error surface and we critically discuss the implications of these results on the relative difficulty of learning with different data representations. In order to develop such a discussion, in this paper we take the result published in [4] as a starting point and we add a considerable number of new results [4]

Our analyses are based on some assumptions discussed below. First, we relate the difficulty of the learning task (i.e., its computational complexity) to the presence of suboptimal stationary points in the error surface (also popularly but somewhat improperly called "local minima"). The loading problem is a well known intractable problem for feedforward neural nets [10, 2]. This clearly points out that under the constraint of a given representation (i.e., network and representation of data are chosen by an adversary) the learning problem is, in general, computationally hard. However (as testified by a quite large amount of successful applications) if the representation is carefully chosen, these computational limitations are not experienced in practice.

As a first approximation, we assume that tractability/intractability is tightly related to absence/presence of suboptimal stationary points.

To avoid local minima there are essentially two known possibilities. The first one is to relax the representational constraints, either by permitting a large enough expressive power (by increasing the number of hidden units) [14, 9, 16] or by increasing the dimensionality of the input space. As a major drawback, the generalization to new examples can be severely undermined in this way. The second one is to put generality aside, limiting the investigation to "easy" learning problems such as the classification of linearly separable patterns [8].

In this paper we claim that an interesting alternative for enlarging the class of "easy" problems be sought by a rather radical change of perspective in the representation used by the learner. Whenever representing instances by means of labeled graphs is an option, such representation should be used instead of a "flat" (i.e. vector-based) representation.

[1] University of Siena, Dept. of Information Engineering, Via Roma 56, I-53100 Siena, Italy, email: marco@ing.unisi.it

[2] University of Florence, Dept. of Systems and Computer Science, Via di Santa Marta 3, I-50139 Firenze, Italy, email: paolo@dsi.unifi.it

[3] University of Pisa, Dept. of Computer Science, Corso Italia 40, I-56125 Pisa, Italy, email: perso@di.unipi.it

[4] For obvious space limitations the proofs are not reported here, but can be found in [6].

2 Recursive neural networks for data structures

The input for a recursive neural network model is a labeled DOAG $\mathcal{U}$, where the label $\boldsymbol{u}_v$ attached to each node v is a vector in R^m, possibly encoding categorical attributes (and perhaps including a constant component to simulate biases). The DOAG is assumed to possess a supersource s and we also assume that its maximum outdegree is lesser or equal than a fixed integer o. The network operates according to the following scheme. For each node v, we define a state vector $\boldsymbol{x}_v \in R^n$ computed as follows:

$$\boldsymbol{x}_v = \sigma\left(\sum_{k=1}^{o} \boldsymbol{A}(k) q_k^{-1} \boldsymbol{x}_v + \boldsymbol{B}\boldsymbol{u}_v\right) \tag{1}$$

where σ is the usual logistic function, $q_k^{-1}\boldsymbol{x}_v$ is the state associated with the k-th child of node v (or the zero n-vector, if such child is missing) and $\boldsymbol{A}(k)$ and $\boldsymbol{B}$ are weight matrices. Note that using the same formalism it is also possible to deal with directed positional acyclic graphs (DPAG). Since there are no directed cycles, this computation is always well defined and can be serialized following any topological order of the graph, starting from the leaves and ending at the supersource s. The state vector at s, $\boldsymbol{x}_s$ has a special meaning. It is in fact an adaptive code for the whole labeled graph (a similar idea was exploited in [13] for unlabeled and labeled graphs, respectively). In the present approach, the graph is classified by mapping $\boldsymbol{x}_s$ to a real value using a single layered network:

$$y = \sigma(\boldsymbol{C}'\boldsymbol{x}_s) \tag{2}$$

where $\boldsymbol{C}$ is a weight vector and $y \in [-1, 1]$ can be interpreted as the conditional probability of the class given the input graph[5]. Each graph $\mathcal{U}_l$ in the training set has a corresponding target $d_l \in \{-1, 1\}$. The error function measures the mismatch between d_l and y_l.

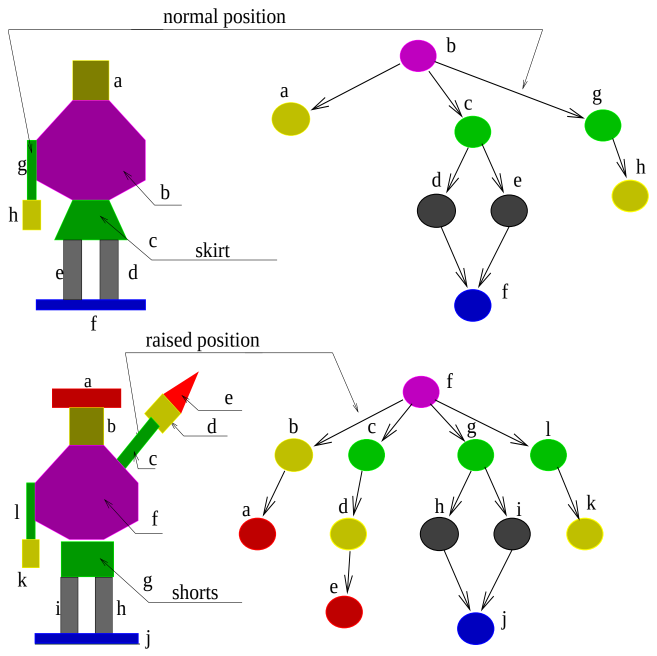

Figure 1. Cops and the corresponding DOAGs which can uniquely be created.

[5] For the sake of simplicity we have assumed a classification task with two classes only. 1 out of n classification is a straightforward generalization.

3 The cop learning task

In this section we introduce an artificial learning task useful for discussing the theoretical results reported in the remaining of the paper. This task is used for comparing the efficiency of learning when using different representations of the patterns. Cops are sketched using a set of blocks (nodes) connected each other (see Figure 1). A graphical description of these patterns is possible where the blocks are represented by nodes and the connection between blocks by edges. Each block is characterized by three attributes describing shape, color, and size. These three attributes (using a proper encoding) will form the label of each vertex of the graph. Although simple and perhaps coarse, this domain turns out to be quite interesting because pictures such as those shown in Figure 1 can be easily generated by an attributed plex grammar. If we assume that there is a starting block (e.g. the body) then these pictures can be given a unique representation by means of directed graphs. Note that, because of the construction, the graph turns out to be a directed ordered graph (DOAG), that is there is also an ordering in the children of each node. More precisely, since also the position of the children can be relevant in these graphs, they are sometimes more properly referred to as directed positional acyclic graphs (DPAGs) [3].

We may define a set of interesting target concepts on this domain. Here are some examples:

1. $\mathcal{P}(\texttt{size.color})$, e.g. only *large blue policemen* are positive instances.
2. $\mathcal{P}(\texttt{sex})$. We assume that policemen dress short, whereas policewomen dress skirt and the task consists of learning the sex.
3. $\mathcal{P}(\texttt{hat} \vee \texttt{skirt})$, an instance is positive iff either `hat` or `skirt` is present.
4. $\mathcal{P}(\texttt{gun} \wedge \texttt{large})$, positive examples must have a `gun` and be `large`.
5. $\mathcal{P}(\texttt{raised-arm})$, an instance is positive iff one or both arms are raised.
6. $\mathcal{P}(\texttt{halting1})$, and $\mathcal{P}(\texttt{halting2})$ - `halting` concepts
   Consider the problem of following the directions of a traffic policeman located at a cross-roads. The traffic policeman is supposed to provide direction by using the signal or by simple classical gestures as indicated in Figure 1, in which the stop is signalled by raising one or two hands. This loading problem is denoted $\mathcal{P}(\texttt{halting1})$, whereas $\mathcal{P}(\texttt{alting2})$ consists of recognizing those cops who have either one/two raised hands or the red sign.
7. $\mathcal{P}(\texttt{depth})$
   Classify a collection of DOAG's depending on their depth. Loading problem $\mathcal{P}(\texttt{depth})$ might be given nice interpretations in the cop world.
8. $\mathcal{P}(\texttt{out} - \texttt{degree})$
   The problem consists of classifying graphs depending on their maximum outdegree. This problem is also meaningful in the cop world.

In order to define these learning tasks completely, we must specify the representation of the patterns. When choosing pixel-based representations, depending on the assumptions we give in this artificial world, especially some of the defined loading problems can become very hard. The critical comments raised by Minsky in the epilogue of Perceptrons [12] hold particularly for structured learning tasks. In order to develop a fair comparison between static and dynamic representations, we make the assumption that these pictures can be processed in such a way to construct the corresponding directed graph.

This is made possible by the underlying hypothesis that we are dealing with patterns that are not severely corrupted by noise. A natural dynamic representation is the graph itself, whereas we take any equivalent sequential representation of the graph, properly padded with zeros so as to complete the coordinates up the the maximum (the maximum number of coordinates is in fact derived from the largest graph of the training set). For instance, the parenthesized representation or the representation based on the combination of the in-order and pre-order traversals are equivalent sequential representations for binary tree. In the remainder of the paper, we give theoretical results for supporting the conjecture that graphical representations of the input with the corresponding recursive networks simplify the loading problem with respect to the traditional feature-based representations (static data types). Before starting the comparison between array- and graphical-based representations we note that variable size patterns could also be represented directly by the equivalent list without padding with zero. This kind of representation makes it possible to adopt recurrent neural networks[6] for the classification. Such an approach, however, would suffer two major drawbacks. First, the sequences obtained from graph traversals are likely to be very long (for example, in a tree the number of nodes grows exponentially with the height) thus making learning very hard because of long-term dependencies [1]. The same problem does not hold for graphical representation (especially when they are reasonably balanced): the backpropagation through structure algorithm backpropagates the error along paths whose length is not greater than the depth of the graph. Second, mapping of data structures into sequences is likely to break some nice regularities inherently associated with the data structure, thus making generalization very hard. Of course, this problem holds also when using any feature-based representation.

4 Conditions for local minima free learning

In this section we give a set of sufficient conditions for the absence of stationary points. We do not consider the equivalent task based on recurrent networks for sequences for the reasons already discussed, whereas we focus attention on the comparison with respect to feedforward networks. In particular we are interested in comparing the loading problems deriving from the dynamic and the corresponding static representation. Given a set of training graphs $\mathcal{L}_s$ (with associated targets), let $\mathcal{L}_u$ be the set of training examples obtained from an equivalent sequential representation as described in the previous section. For the structured representation define the set $\mathcal{E}_s$ of all the error functions associated with $\mathcal{L}_s$ and the set of all the RNNs such that a zero-cost solution exists. Similarly, for the unstructured representation define the set $\mathcal{E}_u$ of the error functions associated with $\mathcal{L}_u$ and all the feedforward nets such that a zero cost solution exists. Note that in this way we have restricted our analysis to the set of networks having a sufficient expressive power to perfectly load all the examples.

4.1 Conditions related to the labels

Here we study global conditions based on node contents. Let us denote by $\boldsymbol{U}$ the matrix obtained by grouping into columns the label vectors attached to all the nodes of all the DOAGs in the training set.

[6] Recurrent networks for sequences are in fact a special case of recursive networks. In this paper we call *recursive* those networks that can deal with graphs, and *recurrent* those limited to sequences.

Now suppose to arrange the labels into two matrices, $\boldsymbol{U}^+$ and $\boldsymbol{U}^-$, collecting all the labels found in positive and negative examples, respectively.

Theorem 4.1 *If $\boldsymbol{U}^+$ and $\boldsymbol{U}^-$ are linearly separable then every function in $\mathcal{E}_s$ is unimodal.*

These theorems correspond to similar results for feedforward networks (see [8]) and thus, under conditions that guarantees $\mathcal{E}_u$ contain only unimodal functions, $\mathcal{E}_s$ is guaranteed to contain unimodal functions as well.

Example 4.1 Consider the concept $\mathcal{P}(\texttt{size.color})$ defined in Section 3. If the color is coded in the RGB (red-green-blue) space, then the features `size.color` can be given a real-valued representation. In this case, the linear separability required by Theorem 4.1 can hold in many interesting cases (for instance in the case in which the positive examples are *large blue cops*).

Note that in this example the graphical representation does not provide advantages with respect to the static counterpart in the framework of our comparison (absence of local minima).

4.2 Conditions related to the graph topology

In this section we give conditions that take into account both the topology and the labels of the training graphs. The main idea behind these results is that, because of the topology and because of the backpropagation through structure gradient computation scheme, there are training examples that do not contribute some gradient components.

Definition 4.1 *Let $\mathcal{N}$ be a recursive network which is fed by graph $\mathcal{U}_l \in U^{\#}$ (set of graphs of the training set). The* gradient contribution *$\hat{G}_{B,i,j,l}$ of $\boldsymbol{U}_l$ is defined as*

$$\hat{G}_{B,i,j,l} \doteq \begin{cases} 1 & if \;\; \exists\, \omega \in \Omega : \; G_{B,i,j,l}(\omega) \neq 0, \\ 0 & otherwise. \end{cases} \qquad (3)$$

where Ω is the weight space corresponding to matrix $\boldsymbol{B}$. Note that $\hat{G}_{B,i,j,l}$ is non-zero if DOAG $\mathcal{U}_l$ contributes, for at least a weight b_{ij}, to the gradient element $G_{B,i,j,l}(\omega)$. A special interesting case arises when the decoupling is due to a bias term b_j.

Definition 4.2 *For each element b_{ij} we define the* DOAG contribution set *w.r.t. the learning environment $\mathcal{L}^{\#}$ as follows:*

$$\lambda_b(i,j|\mathcal{L}^{\#}) \doteq \quad \{l \in \triangleright\mathcal{L}^{\#} \;:\; \hat{G}_{B,i,j,l} = 1\}. \qquad (4)$$

where $\triangleright$ is the pointer operator; hence $\triangleright\mathcal{L}^{\#}$ is the set of indices of $\mathcal{L}^{\#}$. Note that $\lambda_b(i,j|\mathcal{L}^{\#})$ collects all DOAG's that contribute to the corresponding element (i,j) of $\boldsymbol{G}_B$.

Definition 4.3 *Let $\mathcal{C}_+^{\#}$ and $\mathcal{C}_-^{\#}$ be the partition of $\mathcal{L}^{\#}$ based on the positive and negative examples, that is $\mathcal{L}^{\#} = \mathcal{C}_+^{\#} \bigcup \mathcal{C}_-^{\#}$ and $\mathcal{C}_+^{\#} \bigcap \mathcal{C}_-^{\#} = \emptyset$. Weight b_{ij} is* decoupled *w.r.t. $\Gamma \subseteq \mathcal{C}^{\#}$ (either $\mathcal{C}_+^{\#}$ or $\mathcal{C}_+^{\#}$) provided that*

$$\lambda_b(i,j|\mathcal{L}^{\#}) \cap \triangleright\Gamma = \emptyset \qquad (5)$$

Note that if $b_{\hat{i},\hat{j}}$ is decoupled w.r.t. $\mathcal{C}_+^{\#}$ ($\mathcal{C}_-^{\#}$) then $\lambda_b(\hat{i},\hat{j}|\mathcal{L}^{\#}) \subseteq \triangleright\mathcal{C}_-^{\#}$ ($\lambda_b(\hat{i},\hat{j}|\mathcal{L}^{\#}) \subseteq \triangleright\mathcal{C}_+^{\#}$).

Definition 4.4 *Classes $\mathcal{C}_{+}^{\#}$ and $\mathcal{C}_{-}^{\#}$ are* decoupled *w.r.t. weight $b_{\hat{i},\hat{j}}$ provided that:*

$$\lambda_b(\hat{i},\hat{j}|\mathcal{L}^{\#}) = \triangleright\mathcal{C}_{+}^{\#} \text{ or } \lambda_b(\hat{i},\hat{j}|\mathcal{L}^{\#}) = \triangleright\mathcal{C}_{-}^{\#}. \quad (6)$$

Remembering that the gradient is a linear operator (i.e., $\partial E/\partial w = \sum_l \partial E_l/\partial w$) the absence of decoupling can be shown to be one of the main reasons for the presence of stationary points: gradients vanishes because the contributions from positive and negative examples cancel out. The following result hold:

Theorem 4.2 (label decoupling)
The error function E is unimodal provided there exists a weight b_{ij} of matrix $\boldsymbol{B}$ such that classes $\mathcal{C}_{+}^{\#}$ and $\mathcal{C}_{-}^{\#}$ are decoupled *w.r.t. weight $b_{i,j}$.*

Theorem 4.3 (pointer decoupling)
The error function E is unimodal provided there exists a pointer k such that at least for one element a_{ijk} of matrix $\boldsymbol{A}(k)$ classes $\mathcal{C}_{+}^{\#}$ and $\mathcal{C}_{-}^{\#}$ are decoupled.

From these two theorems it is easy to show the following consequence.

Corollary 4.1 *Suppose that training graphs are labeled by categorical attributes encoded using one-hot coding. Also suppose there is at least one node that occurs in positive examples only or in negative examples only. Then every function in $\mathcal{E}_s$ is unimodal.*

Example 4.2 Let us consider the concepts $\mathcal{P}(\texttt{sex})$ and $\mathcal{P}(\texttt{hat})$. Clearly every training set for this concept matches the conditions of Corollary 4.1, thus giving rise to a unimodal loading problem. Basically, the presence of a *discriminant block* (`skirt` or `shorts`) for $\mathcal{P}(\texttt{sex})$ or the presence of the block `hat` for $\mathcal{P}(\texttt{hat})$ makes it possible to conclude that there are no suboptimal local minima in the error function. Let us consider the static counterparts of $\mathcal{P}(\texttt{sex})$ and $\mathcal{P}(\texttt{hat})$. Because of the assumptions made in the creation of graphs from pictures (see Figure 1), the block `hat` occupies a fixed position in the equivalent sequential representation. Hence, the corresponding loading problem is linearly separable [8]. On the opposite, when converting graphs to sequential representations for loading problem $\mathcal{P}(\texttt{sex})$), we can easily see that the position of the blocks `skirt` and `short` is *not* in a fixed position, but can also considerably be shifted because of the lack of some blocks in the pictures. It can easily be shown that in this case the static counterpart of the given problem may not be linearly-separable. Hence, $\mathcal{P}(\texttt{sex})$ is *easy when using structured representation of the pictures, but can be very hard when choosing equivalent static representations.*

Theorem 4.2 and the correspondent Corollary 4.1 can be extended to the case of logical conditions associated to the presence/absence of nodes. Let us define the class of *discriminant OR problems*, denoted $\mathcal{P}_{\vee}$, as the class of loading problems for which there exists a set V_d of discriminant nodes such that a training graph is positive iff it contains at least one node in V_d.

Corollary 4.2 *For every training set $\mathcal{L}_s$ that originates from a discriminant OR problem, the associated set of error functions $\mathcal{E}_s$ only contains unimodal functions.*

The class of *discriminant AND problems* can be defined in a similar way and an analogous result holds true:

Corollary 4.3 *For every training set $\mathcal{L}_s$ that originates from a discriminant AND problem, the associated set of error functions $\mathcal{E}_s$ only contains unimodal functions.*

Example 4.3 Problem $\mathcal{P}(\texttt{hat} \vee \texttt{skirt})$ as defined in Section 3 is clearly a discriminant OR problem (discriminant nodes are `hat` and `skirt`) and, by Corollary 4.2, it has associated unimodal error functions. One could regard $\mathcal{P}(\texttt{hat} \vee \texttt{skirt})$ as the loading problem $\mathcal{P}(\texttt{woman})$ associated with the recognition of police-women.

Example 4.4 The problem $\mathcal{P}(\texttt{gun} \wedge \texttt{large})$ is a discriminant AND problem and, by Corollary 4.3, it has associated unimodal error functions. Note that `large` is regarded as a global feature. One could regard $\mathcal{P}(\texttt{gun} \wedge \texttt{large})$ as the loading problem $\mathcal{P}(\texttt{man})$ for the recognition of policemen.

Like discriminant nodes, the concept of discriminant pointers turn out to be very useful for producing decoupling and, consequently, the absence of local minima.

Example 4.5 Consider the concept $\mathcal{P}(\texttt{raised-arm})$. If we use a DPAG for representing the pictures then from Theorem 4.3 we conclude that $\mathcal{P}(\texttt{raised-arm})$ has associated unimodal error functions. A raised arm is in fact associated with the corresponding pointer which connects the body to the arm (see Figure 1). Note that the same conclusion could not easily be derived when using DOAG-based representations of the policemen.

Interestingly enough, the more general decoupling conditions which either involve the nodes or the edges makes it possible to study also the `halting1` and `halting2` loading problem. The theorems reported in this section do not have a counterpart for flat representations. In particular there is no guarantee that instances in the training set $\mathcal{U}_u$ are linearly separable. This makes it evident that *there are learning problems that are easier using a structured representation.*

4.3 Topological indices

When labels are all categorical, the training set $\mathcal{L}_s$ can be compacted into a single DOAG [15]. This is done by introducing a topology-based equivalence relation on the nodes of the training graphs. Two nodes v and w are topologically equivalent iff the labeled subgraphs induced by their descendants are identical (the relationship is denoted by $\bowtie$). The equivalence can be explained by noting that running any RNNs on the graphs in which v and w appear, the associated hidden state representation are indistinguishable (i.e., $\boldsymbol{x}_v = \boldsymbol{x}_w$). A single DOAG $\mathcal{U}^*$ is then constructed by including only the representants of the equivalence classes[7]. The number of nodes of $\mathcal{U}^*$ is called *power pointer* of the training set, denoted $\Uparrow \mathcal{L}_s$. The power pointer only depends on the topology and the labeling of the training graphs. In [4] it is shown that $n \geq \Uparrow \mathcal{L}_s$ is a sufficient condition for the absence of stationary points (the result can be seen as a generalization of [16]). Another topological index can be introduced which makes it possible to determine in advance the number of hidden units required to avoid suboptimal local minima. The following new topological index in introduced:

Definition 4.5 *Given $\mathcal{L}^{\#}$ with categorical variables we call* brotherhood index *of $\mathcal{L}^{\#}$ the number*

$$\models \mathcal{L}^{\#} \doteq \max_{k \geq 1} \models \mathcal{L}_k^{\#} \doteq \max_{k \geq 1} |\, U^{\#}/_{\bowtie_k} \,| \,. \quad (7)$$

[7] The supervision scheme when learning with the single graph $\mathcal{U}^*$ needs to slightly changed since every node that appears as a supersource in the original training set has an associated target.

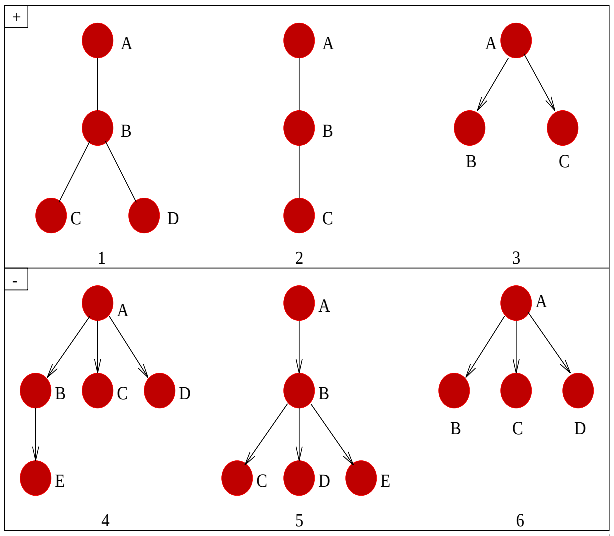

Figure 2. The problem of classifying DOAGs depending the their outdegree. Positive DOAGs are those with outdegree less than three.

Example 4.6 Consider loading problem $\mathcal{P}(\texttt{out} - \texttt{degree})$. In [4], it was shown that $\mathcal{D}_L \Uparrow = 8$ and that a network with 8 hidden units is sufficient in order to avoid local minima. Hence, $\Uparrow \mathcal{L}^{\#} = 8$. Likewise we can calculate $U_L^{\#}/\bowtie_k$:

$$\begin{aligned} U_L^{\#}/\bowtie_1 &= \{(C-1, C-2, B-3, E-4, C-5, B-6) \\ &(B-1), (B-2, B-4), (B-5)\}. \\ U_L^{\#}/\bowtie_2 &= \{D-1, C-3, C-4, D-5, C-6)\}. \\ U_L^{\#}/\bowtie_3 &= \{D-4, E-5, D-6)\}. \end{aligned}$$

Hence, $\models \mathcal{L}^{\#} = 4$. It can be proven that a recursive neural network with $n = \models -1 + \mathcal{L}^{\#} = 3$ hidden units, guarantees the absence of local minima.

5 Conclusions

In this paper we have considerably extended the results given in [4] by presenting sufficient conditions which make it possible to conclude that many interesting loading problems of data structures into recursive neural networks are *easy* (no suboptimal local minima populate the attached error function). We have introduced an artificial problem, the cop learning task, and have shown that graphical representations of the instances make learning easy, whereas when providing an equivalent static representation of the same data the loading problem may become very hard. The main conclusion is that whenever representing instances by means of labeled graphs is an option, such representation should be used instead of a "flat" (i.e. vector-based) representation.

REFERENCES

[1] Y. Bengio, P. Frasconi, and P. Simard, 'Learning long-term dependencies with gradient descent is difficult', *IEEE Trans. on Neural Networks*, **5**(2), 157–166, (1994).

[2] A. Blum and R. L. Rivest, 'Training a 3-node neural net is NP-complete', in *NIPS 1*, ed., D. S. Touretzky, pp. 494–501, San Mateo, CA, (1989). Morgan Kaufmann.

[3] P. Frasconi, 'An introduction to learning structured representations', in *Lecture Notes in Artificial Intelligence, 1387*, eds., M Gori and C.L. Giles, chapter 4, 99–120, Springer Verlag, (1998).

[4] P. Frasconi, M. Gori, and A. Sperduti, 'On the efficient classification of data structures by neural networks', in *Proc. Int. Joint Conf. on Artificial Intelligence*, (1997).

[5] P. Frasconi, M. Gori, and A. Sperduti, 'A general framework for adaptive processing of data structures', *IEEE Trans. on Neural Networks*, **9**(5), 768–786, (1998).

[6] P. Frasconi, M. Gori, and A. Sperduti, 'Learning efficiently with neural networks: A theoretical comparison between structured flat representations (extended version of this paper)', Technical report, University of Siena, (2000).

[7] C. Goller and A. Küchler, 'Learning task-dependent distributed structure-representations by backpropagation through structure', in *IEEE International Conference on Neural Networks*, pp. 347–352, (1996).

[8] M. Gori and A. Tesi, 'On the problem of local minima in backpropagation', *IEEE Trans. on Pattern Anal. and Mach. Intell.*, **14**(1), 76–86, (1992).

[9] L.G.C. Hamey, 'Comment on "can backpropagation error surface not have local minima?"', *IEEE Trans. on Neural Networks*, **5**(5), 844, (1994).

[10] J.S. Judd, *Neural Network Design and the Complexity of Learning*, The MIT Press, Cambridge, London, 1990.

[11] M. J. Kearns and U. V. Vazirani, *An Introduction to Computational Learning Theory*, MIT Press, 1994.

[12] M.L. Minsky and S.A. Papert, *Perceptrons - Expanded Edition*, MIT Press, Cambridge, 1988.

[13] J. B. Pollack, 'Recursive distributed representations', *Artificial Intelligence*, **46**(1-2), 77–106, (1990).

[14] T. Poston, C. Lee, Y. Choie, and Y. Kwon, 'Local minima and backpropagation', in *Proc. of the IEEE-IJCNN91*, pp. 173–176, Seattle, WA, (1991).

[15] A. Sperduti and A. Starita, 'Supervised neural networks for the classification of structures', *IEEE Trans. on Neural Networks*, **8**(3), 714–735, (1997).

[16] X.H. Yu and G.A. Chen, 'On the local minima free condition of backpropagation learning', *IEEE Trans. on Neural Networks*, **6**(6), 1300–1303, (1995).