ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 325-330

Recursive Networks for Processing Graphs with Labelled Edges

M. Bianchini, M. Maggini, L. Sarti, and F. Scarselli
Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Siena
Via Roma, 56 — 53100 Siena (ITALY)
Email:{monica,maggini,sarti,franco}@dii.unisi.it

Abstract. In this paper, we propose a new recursive neural network model, able to process directed acyclic graphs with labelled edges. The model is based on a different definition of the state transition function, which is independent both from the number and the order of the children of each node. In fact, the particular contribution of each child is encoded in the label attached to the corresponding edge. The computational capabilities of the new recursive architecture are also assessed.

1 Introduction

Recursive neural networks are a connectionist model [1, 2, 3] particularly suited to process Directed Positional Acyclic Graphs (DPAGs). In the case of DPAGs, each arc starting from a node has an assigned position, and any rearrangement of the children of a node produces a different graph. While such an assumption is useful in some applications, it sometimes introduces unnecessary constraints on the representation. For example, this hypothesis is not suitable when representing a chemical compound or a protein structure, and might not be adequate in several pattern recognition problems.

The concept of invariance or symmetry has emerged as one of the key ideas in many different areas, such as physics, mathematics, psychology, and engineering [4]. From a theoretical point of view, seminal contributions in permutation invariant algebras can be found in [5, 6]. Significant applications are in computer vision [4, 7], in signal theory [8], and in pattern recognition [9, 10]. Even in the neural network research field, there have been some attempts to construct application–specific architectures, permutation invariant w.r.t. the inputs. Invariance is often realized with preprocessing [11, 12], or via the extraction of features which are invariant under input transformations [13]. Otherwise, *ad hoc* network models are used, to preserve the output under particular input transformations [14, 15].

In this paper, we present a novel recursive neural network model, able to process Directed Acyclic Graphs with Labelled Edges (DAGs–LE). At each

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 325-330

node of the structure to be learnt, the state transition function is independent both from the number and the order of the children. In fact, the particular contribution of each child is encoded in the label attached to the corresponding edge. The paper is organized as follows. In the next section, some notation is introduced whereas, in Section 3, the novel recursive model is described. In Section 4 some theoretical results are shown in order to assess the computational power of the model. Finally, conclusions are drawn in Section 5.

2 Notation

Let $G = (V, E, \mathcal{L})$ be a directed graph, where V is the set of nodes, $E \subseteq V \times V$ represents the set of arcs, and $\mathcal{L} : V \rightarrow L_v$ is a labelling function, being $L_v \subset \mathbf{R}^m$ a finite set of labels. Given any node $v \in V$, pa$[v]$ is the set of the *parents* of v, while ch$[v]$ represents the set of its *children*. The *outdegree* of v, od$[v]$, is the cardinality of ch$[v]$, and $o = \max_v \text{od}[v]$ is the maximum outdegree. Each node stores a set of domain variables into a *label*. The presence of an edge (v, w) in a labelled graph stands for the existence of a causal link between the labels of v and w. Moreover, for recursive processing, G should have a *supersource*, i.e. a node $s \in V$ with no incoming edges, and from which any other node in V can be reached. The supersource s may eventually be added following the algorithm in [3].

In this paper, we consider the class of Directed Acyclic Graphs (DAGs), where a partial ordering can be defined on V, such that $v \prec w$ if v is connected to w by a direct path. Directed Positional Acyclic Graphs (DPAGs), for which recursive networks were originally defined, are a subclass of DAGs, where an injective function $o_v : \text{ch}[v] \rightarrow \{1, \ldots, o\}$ assigns a position $o_v(c)$ to each child c of a node v. Therefore, a DPAG is represented by the tuple $(V, E, \mathcal{L}, \mathcal{O})$, where $\mathcal{O} = \{o_1, \ldots, o_{|V|}\}$ is the set of functions defining the position of the children for each node. Finally, the definition of the class of DAGs with Labelled Edges (DAGs–LE) requires the introduction of an additional edge labelling function $\mathcal{E}$, such that $G = (V, E, \mathcal{L}, \mathcal{E})$, where $\mathcal{E} : E \rightarrow L_e$, and L_e is a finite subset of $\mathbf{R}^k$. The presence of an edge label introduces some semantical contents into the link between two nodes (e.g., when the nodes represent two adjacent regions in an image, the label can collect some information on their respective positions [16]) .

3 Processing DAGs with labelled edges

Recursive neural network (RNN) models proposed so far have been devised to deal with DPAGs. In this case, the state transition function f takes into account the order of the children of each node, because the state of each child occupies a particular position in the list of the arguments of f. In [14], a weight sharing approach was proposed, to relax the order constraint, which allows to assess that RNNs are universal approximators also for DAGs (with a bounded

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 325-330

outdegree). Unfortunately, such method cannot be applied to DAGs with a large outdegree o, due to the factorial growth in the network parameters w.r.t. o. Even if the maximum outdegree can be bounded during a preprocessing phase, for instance by pruning those connections that are heuristically classified as less informative, nevertheless some important information may be discarded in this way. Such drawbacks can be removed when considering DAGs–LE.

For DAGs–LE, a state transition function $\tilde{f}$ can be defined which has not a predefined number of arguments and that does not depend on their order. The different contribution of each child depends on the label attached to the corresponding edge. At each internal (not a leaf) node v, the total contribution $\overline{\mathbf{X}}(\mathrm{ch}[v]) \in \mathbf{R}^p$ of the state of its children is computed as

$$\overline{\mathbf{X}}(\mathrm{ch}[v]) = \frac{1}{|\mathrm{ch}[v]|}\left(\sum_{i=1}^{|\mathrm{ch}[v]|}\left(\sum_{j=1}^{k}\mathbf{H}_j\mathbf{L}^{(j)}_{(v,\mathrm{ch}_i[v])}\right)\mathbf{X}_{\mathrm{ch}_i[v]}\right) \tag{1}$$

where $\mathbf{L}_{(v,\mathrm{ch}_i[v])} \in \mathbf{R}^k$ is the label attached to the edge $(v, \mathrm{ch}_i[v])$, and $\mathbf{H} \in \mathbf{R}^{p,n,k}$ is the weight matrix. In particular, $\mathbf{H}_j \in \mathbf{R}^{p,n}$ is the j–th layer of matrix $\mathbf{H}$ and $\mathbf{L}^{(j)}_{(v,\mathrm{ch}_i[v])}$ is the j–th component of the edge label. Finally, the state at a generic internal node v is computed by a two–layer perceptron with linear outputs, as

$$\mathbf{X}_v = \tilde{f}(\overline{\mathbf{X}}(\mathrm{ch}[v]), \mathbf{U}_v, \theta_{\tilde{f}}) = \mathbf{V}\vec{\sigma}(\mathbf{A}\overline{\mathbf{X}}(\mathrm{ch}[v]) + \mathbf{B}\mathbf{U}_v + \mathbf{C}) + \mathbf{D} \tag{2}$$

where σ is a sigmoidal function, $\theta_{\tilde{f}}$ collects $\mathbf{A} \in \mathbf{R}^{q,p}$, $\mathbf{B} \in \mathbf{R}^{q,m}$, $\mathbf{C} \in \mathbf{R}^q$, $\mathbf{D} \in \mathbf{R}^n$, and $\mathbf{V} \in \mathbf{R}^{n,q}$, being q the number of hidden units. The state of the leaves is conventionally fixed to the frontier state X_F. At the supersource, also an *output* function g is evaluated by a feedforward network,

$$\mathbf{Y}_s = g(\mathbf{X}_s, \theta_s) = \mathbf{W}\vec{\sigma}(\mathbf{E}\mathbf{X}_s + \mathbf{F}) + \mathbf{G}$$

with $\mathbf{E} \in \mathbf{R}^{q',n}$, $\mathbf{F} \in \mathbf{R}^{q'}$, $\mathbf{G} \in \mathbf{R}^r$, and $\mathbf{W} \in \mathbf{R}^{r,q'}$.

Remark 1. In the more general case, $\overline{\mathbf{X}}(\mathrm{ch}[v])$ may be computed using a nonlinear function $\phi : \mathbf{R}^{(n+k)} \to \mathbf{R}^p$, depending on a set of parameters θ_ϕ:

$$\overline{\mathbf{X}}(\mathrm{ch}[v]) = \frac{1}{|\mathrm{ch}[v]|}\left(\sum_{i=1}^{|\mathrm{ch}[v]|}\phi(\mathbf{X}_{\mathrm{ch}_i[v]}, \mathbf{L}_{(v,\mathrm{ch}_i[v])}, \theta_\phi)\right)$$

4 Theoretical results

In Section 3, eqs. (1)–(2) describe how to compute the state of a node v, starting from the contribution of its children and independently of their number and order. In the following lemma the feedforward architecture which computes the transition function $\tilde{f}$ is defined.

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 325-330

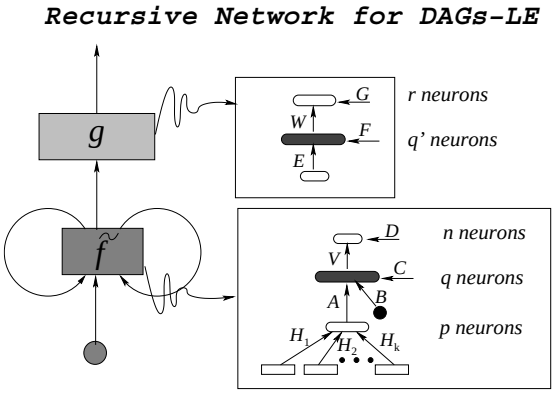

Figure 1: An MLP implementation of the recursive network. Grey layers are sigmoidal, whereas white layers are linear in both the feedforward networks.

Lemma 1. *At each node v of the structure to be learnt, the feedforward network which computes the state transition function $\tilde{f}$ can be realized by a three–layer perceptron, having $\mathbf{H}_j$, $j = 1, \ldots, k$, as input–to–hidden weight matrices.*

Proof. Eq. (1) may be rewritten as

$$\overline{\mathbf{X}}(\mathrm{ch}[v]) = \frac{1}{|\mathrm{ch}[v]|} \left(\sum_{j=1}^{k} \mathbf{H}_j \left(\sum_{i=1}^{|\mathrm{ch}[v]|} \mathbf{L}^{(j)}_{(v,\mathrm{ch}_i[v])} \mathbf{X}_{\mathrm{ch}_i[v]} \right) \right)$$

which shows how the contribution of the children of each node to its state can be computed using a three–layer perceptron with k inputs $\sum_{i=1}^{|\mathrm{ch}[v]|} \mathbf{L}^{(j)}_{(v,\mathrm{ch}_i[v])} \mathbf{X}_{\mathrm{ch}_i[v]}$, $j = 1, \ldots, k$, and with $\mathbf{H}_j$ as input–to–hidden weight matrices. □

The recursive neural network for processing graphs with labelled edges (RNN–LE) is depicted in Fig. 1.

Even if the edge labels increase the semantical content attached to the links, it is worth studying how such labels can be used to codify the order relationship.

Theorem 1. *Each DPAG can be represented by a DAG–LE.*

Proof. The assertion follows straightforwardly by observing that the position of the i–th child can be encoded by an integer label attached to the link. In particular, for each node in the DPAG, a corresponding node exists in the DAG–LE, and the edge labels must have dimension o (being o the maximum outdegree). Moreover, for each node of the DPAG the corresponding node in the DAG–LE has o outcoming edges, with labels having all but one zero entries. An entry equal to one stands for the presence of the child in a particular position (in this case, the labels form the Euclidean basis of $\mathbf{R}^o$). The absence of a child in a node of the DPAG implies the use of a predefined frontier state, $\mathbf{X}_F$, as the state of the corresponding child in the DAG–LE. □

ESANN'2004 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 28-30 April 2004, d-side publi., ISBN 2-930307-04-8, pp. 325-330

Theorem 2. *For any DPAG G and any standard recursive neural network RNN with a transition function f, there exists a DAG–LE L and an RNN–LE, with a transition function $\tilde{f}$ such that $f(G) = \tilde{f}(L)$.*

Proof. Let us suppose to consider the standard RNN model in which, at each node v, the state transition function f is computed using a two–layer perceptron, with sigmoidal hidden neurons and linear output neurons, in which the matrices $\mathbf{A}_k$, $k = 1, \ldots, o$ weighs the input–to–hidden contribution of the state of the children [14]. Then, the proof follows directly by choosing $\mathbf{A} \cdot \mathbf{H}_k = \mathbf{A}_k$ in the RNN–LE. □

Finally, the following theorem assess the computational capabilities of the RNN–LE architecture.

Theorem 3. *Given a function $t : DPAGs \rightarrow \mathbf{R}^r$, a probability measure P on DPAGs and any real ε, there is a function $\tilde{h}$, realized by an RNN–LE, such that $P(|\tilde{h}(L) - t(G)| \geq \varepsilon) < \varepsilon$.*

Proof. The proof follows straightforwardly from the results in [17, 18, 19]. □

Remark 2. Since any DAG can be represented with a DPAG, by assigning an arbitrary position to each child, the above results can be directly extended to the class of DAGs.

5 Conclusions

In this paper, a novel neural network model, capable of processing DPAGs with labelled edges, was introduced. The feedforward neural network that computes the transition function has a three–layer architecture. The computational capabilities of the recursive model were also discussed, assessing that it works as an universal approximator with respect to the classes of DPAGs and DAGs.

References

[1] P. Frasconi, M. Gori, and A. Sperduti, "A general framework for adaptive processing of data structures," *IEEE Transactions on Neural Networks*, vol. 9, pp. 768–786, September 1998.

[2] A. Küchler and C. Goller, "Inductive learning in symbolic domains using structure–driven recurrent neural networks," in *Advances in Artificial Intelligence* (G. Görz and S. Hölldobler, eds.), pp. 183–197, Springer, 1996.

[3] A. Sperduti and A. Starita, "Supervised neural networks for the classification of structures," *IEEE Transactions on Neural Networks*, vol. 8, pp. 429–459, 1997.

[4] R. Lenz, "Group theoretical transforms in image processing," *Current Topics in Pattern Recognition Res.*, vol. 1, pp. 83–106, 1994.

[5] C. Curtis and L. Reiner, *Representation Theory of Finite Groups and Associative Algebras*. New York: Wiley, 1988.

[6] W. Fulton and J. Harris, "Representation theory," in *Graduate Texts in Mathematics*, New York: Springer–Verlag, 1991.

[7] J. Mundy, A. Zissermann, and D. Forsyth, "Applications of invariance in computer vision," in *Lecture Notes in Computer Science*, New York: Springer–Verlag, 1994.

[8] W. Schempp, "Harmonic analysis of the Heisenberg nilpotent Lie group, with applications to signal theory," in *Pitman Research Notes in Mathematics*, Harlow, Essex (U.K.): Longman, 1986.

[9] R. Lenz, "Optimal filters for the detection of linear patterns in 2–D and higher dimensional images," *Pattern Recognition*, vol. 20, no. 2, pp. 163–172, 1987.

[10] R. Lenz, "A group theoretical approach to filter design," in *International Conference on Acoustic, Speech, and Signal Processing*, 1989.

[11] J. Lahtinen, T. Martinsen, and J. Lampinen, "Improved rotational invariance for statistical inverse in electrical impendance tomography," in *Int. Joint Conf. on Neural Networks*, (Como (ITALY)), pp. 154–158, 2000.

[12] H. Chiu and D. Tseng, "Invariant handwritten Chinese character recognition using fuzzy min–max neural networks," *Physical Review Letters*, vol. 18, no. 5, pp. 481–491, 1997.

[13] E. Nordström, O. Gällmo, L. Asplund, M. Gustafsson, and B. Eriksson, "Neural networks for admission control in an ATM network," in *Connectionism in a Broad Perspective* (L. Niklasson and M. Bóden, eds.), pp. 239–250, Ellis Horwood, 1994.

[14] M. Bianchini, M. Gori, and F. Scarselli, "Processing directed acyclic graphs with recursive neural networks," *IEEE Transactions on Neural Networks*, vol. 12, no. 6, pp. 1464–1470, 2001.

[15] Y. le Cun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, pp. 541–551, 1989.

[16] M. Gori, M. Maggini, and L. Sarti, "A recursive neural network model for processing directed acyclic graphs with labeled edges," in *Int. Joint Conf. on Neural Networks*, vol. 2, (Portland), pp. 1351–1355, IEEE, New York, 2003.

[17] B. Hammer and V. Sperschneider, "Neural networks can approximate mappings on structured objects," in *Int. Conf. on Computational Intelligence and Neural Networks* (P. P. Wang, ed.), pp. 211–214, 1997.

[18] B. Hammer, "Approximation capabilities of folding networks," in *ESANN '99*, (Bruges, (Belgium)), pp. 33–38, April 1999.

[19] B. Hammer, *Learning with Recurrent Neural Networks*. PhD thesis, Fachbereich Mathematik/Informatik — Universität Osnabrück, 1999.