LOCAL PROPAGATION IN NEURAL NETWORK LEARNING BY ARCHITECTURAL CONSTRAINTS

Ph.D Thesis in Information Engineering and Science

Matteo Tiezzi

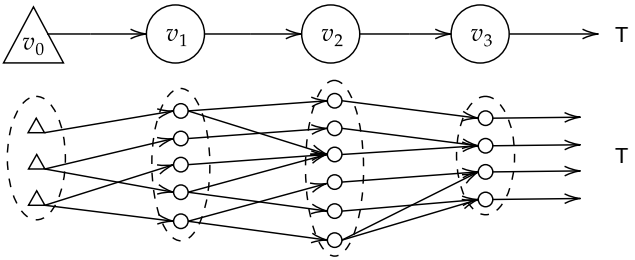
Supervisor: Prof. Marco Maggini

UNIVERSITÀ DI SIENA
1240

SAILab
Siena Artificial Intelligence Lab

ARTIFICIAL NEURAL NETS AS DATA-FLOW COMPUTATIONAL MODELS

- The processing scheme characterizing Artificial Neural Networks (ANNs) can be conveniently described as a data-flow through a computational graph $\mathcal{G} = (V, E)$.
  - Nodes $v_i \in V$: input or intermediate variable (neural activations of neurons residing in the same layer)
  - Edges $e_i \in E$: elementary operations (non-linearities) applied on originating nodes - parametrized by learnable synaptical weights $\mathcal{W}$



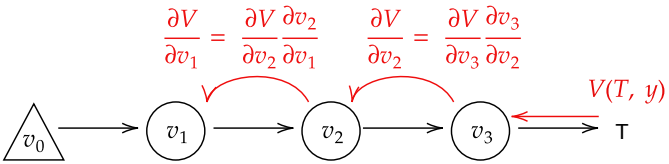
ANNs are trained optimizing an error function of the network predictions with respect to some target values, $V(\cdot)$, in a two-phase process:

1. Evaluate the derivatives of the error function (Automatic Differentiation - BackPropagation) with respect to the learnable parameters $\mathcal{W}$

$$\nabla_{\mathcal{W}} V(\cdot) \tag{1}$$

2. Adjust the parameters towards values which guarantee an improvement of measured performances (Gradient Descent)

$$\Delta \mathcal{W}^{t+1} = \mathcal{W}^{t} - \eta \nabla_{\mathcal{W}^{t}} V(\cdot) \tag{2}$$

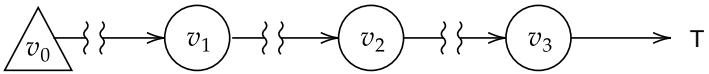


- Forward phase from the roots (input layer) up to the prediction computation, on the leaves (output layer)
- Backward Phase message passing scheme to backpropagate the errors, differentiating through the computational graph (gradients are computed via the chain rule).
  - Pros Computationally efficient
  - Cons
    - High memory consumption to store intermediate values
    - Mandatory sequential nature of computation (Non-locality, hardship in achieving parallelization)

INTUITION: DESCRIBE THE NEURAL ARCHITECTURE BY A MATHEMATICAL SUPERSTRUCTURE

- Learning from Constrains[1] Line of research aimed at expressing and injecting external knowledge onto the domain of the task-at-hand.
  - An agent lives and interact with an environment which imposes the fulfillment of constraints
- Contribution Exploit constraints to force internal knowledge onto the structure of the neural models.

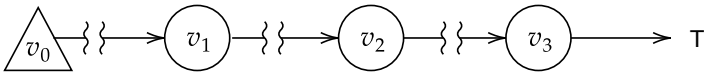
- Decompose neural architectures into local components
- Leverage the mathematical notion of constraint to put into communication such local substructures



[1] Giorgio Gnecco et al. "Foundations of support constraint machines". In: *Neural computation* 27.2 (2015), pp. 388–480.

INTUITION: FLOW OF INFORMATION AS CONSTRAINT SATISFACTION

- Enrich the learning problem with auxiliary local variables (Lifted Networks)
- Constrain their evolution to follow the neural computational scheme
- Neural architectures will be treated as a collection of local submodules, whose interconnection and processing scheme is defined by constraints



This choice allows to setup an optimization procedure that is “local”, i.e. does not require:

1. to query the whole network
2. to accomplish a sequential diffusion of the information
3. to bufferize data streamed over time in order to be able to compute gradients.

constraints describe the dependencies in the neural computation. Hence, the proposed technique can be summarized by the definition Learning by Constraints

The thesis investigates three different learning settings that are instances of the aforementioned scheme:

1. constraints among layers in feed-forward neural networks[2]
2. constraints among the states of neighboring nodes in Graph Neural Networks[3]
3. constraints among predictions over time[4].

[2] G. Marra et al. "Local Propagation in Constraint-based Neural Networks". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. 2020, pp. 1–8. DOI: *10.1109/IJCNN48605.2020.9207043*.

[3] Matteo Tiezzi et al. "A Lagrangian Approach to Information Propagation in Graph Neural Networks". In: vol. 325. Giacomo, Giuseppe De. IOS Press, 2020, pp. 1539–1546. URL: *https://doi.org/10.3233/FAIA200262*.

[4] Matteo Tiezzi et al. "Focus of Attention Improves Information Transfer in Visual Features". In: *Advances in Neural Information Processing Systems* 33 (2020).

BP has become the de-facto algorithm for training neural networks. The sequential nature of the performed computation hinders parallelizations capabilities and causes a high memory consumption.

Is it possible to devise a novel computational method for a generic Directed Acyclic Graph that gets inspiration and advantages from principles of locality?

1 – Local Propagation in Constraint-based Neural Networks

We consider the supervised learning problem defined by

- N example pairs: (x_i^0, y_i), $i = 1 \dots N$;
- An MLP with H hidden layers, denoted by $f(W, \cdot)$;
- We denote with x_i^ℓ the outputs on layer ℓ relative to example i;
  - The computational rule associated with the MLP, for the i-th pattern

$$x_i^\ell = \sigma(W_{\ell-1} x_i^{\ell-1}) \tag{3}$$

- An optimization problem on the loss function $V(\cdot)$.

$$\min_W \sum_{i=1}^N V(f(W, x_i^0), y_i),$$

that is usually solved via

$$\text{GD} + \text{Backprop}$$

- Cast the learning problem as constrained optimization[5].
- Add free variables x_i (to be optimized) corresponding to the neural outputs.

Problem

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} V(f(W, x_i^0), y_i); \\ \text{subject to} \quad & \mathcal{G}(x_i^\ell - \sigma(W_{\ell-1} x_i^{\ell-1})) = 0, \quad i = 1, \dots N, \quad \ell = 1, \dots, H. \end{aligned}$$

With $\mathcal{G}(0) = 0$ (tolerance to noise, improve generalization, stabilize learning).

Enforce the constraint satisfaction: describe the computational scheme and the learning mechanism – Local Propagation (LP).

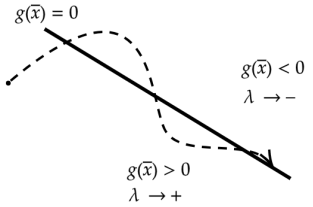
[5] Yann LeCun et al. "A theoretical framework for back-propagation". In: *Proceedings of the 1988 connectionist models summer school*. Vol. 1. CMU, Pittsburgh, Pa: Morgan Kaufmann. 1988, pp. 21–28.

We tackle the constrained problem introducing the Lagrangian

$$L(W, x, \lambda) := \sum_{i=1}^{N} \left(V(x_i^{H+1}, y_i) + \sum_{\ell=1}^{H} \lambda_\ell^i \mathcal{G}(x_i^\ell - \sigma(W_{\ell-1} x_i^{\ell-1})) \right)$$

and then looking for saddle point of this function, in a gradient ascent-descent scheme[6].

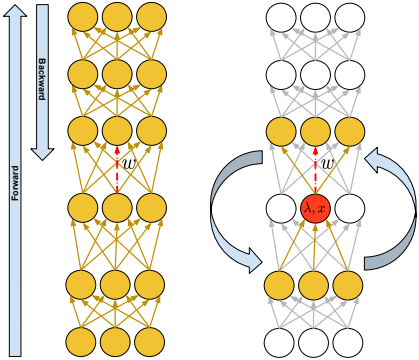
$$\min_{W,X} \max_{\Lambda} \mathcal{L}(W, X, \Lambda)$$



[6] John C Platt and Alan H Barr. "Constrained differential optimization". In: *Neural Information Processing Systems*. 1988, pp. 612–621.

PROPERTIES OF LP

- The gradients of L that appear in the updates rules for W, x and λ can be explicitly calculated and depend only on local quantities.
- It is parallelizable over the neurons;
- It reproduces GD with backprop when $\partial L/\partial x = 0$, $\partial L/\partial \lambda = 0$ and $\mathcal{G} = \mathrm{id}$.



- Prove the feasibility and that generalization skills are in-line with BP.
- Show that locality and parallelization do not correspond to a loss in performance w.r.t. BP even if the problem is lifted with X and Λ.

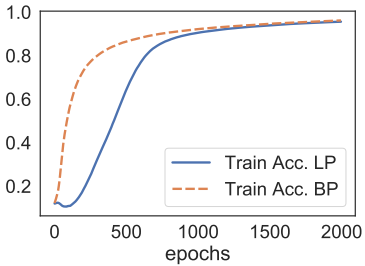
Dataset	Examples	Dimensions	Classes
Adult	48842	14	2
Ionosphere	351	33	2
Letter	20000	16	26
Pima	768	8	2
Wine	179	13	3
Ozone	2536	72	2
Dermatology	366	34	6
MNIST	70000	784	10

Table 1: Experiments on 7 benchmarks from the UCI repository[7] and on the MNIST dataset.

[7] Dheeru Dua and Efi Karra Taniskidou. *UCI Machine Learning Repository*. 2017. URL: `http://archive.ics.uci.edu/ml`.

Table 2: Performances of the same architectures optimized with BP and LP. Left: $H = 1$ hidden layer (100 units); right: $H = 3$ hidden layers (30 units each). Largest average accuracies are in bold.

	BP ($H = 1$)	LP ($H = 1$)	BP ($H = 3$)	LP ($H = 3$)
Adult	84.66 ±0.00	**85.43** ±0.00	84.91 ±0.00	**85.34** ±0.00
Iono.	91.48 ±0.57	91.48 ±2.95	92.61 ±0.57	**94.60** ±1.86
Letter	94.20 ±0.31	**94.94** ±0.05	**92.27** ±0.19	90.42 ±0.78
Pima	76.17 ±1.62	**77.21** ±2.79	**76.56** ±2.42	75.91 ±1.54
Wine	97.16 ±1.88	**98.86** ±1.14	97.73 ±2.78	**98.86** ±1.97
Ozone	97.04 ±0.26	**97.12** ±0.13	**97.28** ±0.13	97.20 ±0.17
Derma.	95.60 ±1.74	**96.70** ±1.74	97.53 ±1.20	**98.63** ±0.48



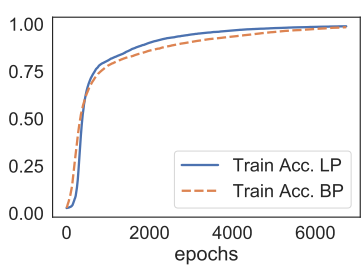


Figure 1: Convergence speed of BP and LP in MNIST (left) and Letter (right).

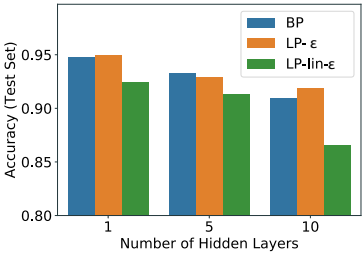
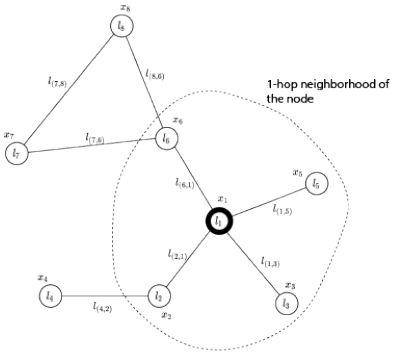


Figure 2: Accuracies of BP and LP (with ε and lin-ε) on the MNIST data.

2 – Lagrangian Propagation Graph Neural Networks



- Graph $G = (V, E)$, where V is a finite set of *nodes* and $E \subseteq V \times V$ collects the *arcs*
- l_i node i features, $l_{(i,j)}$ arc (i, j) features (both optional)
- Structures that allow to represent relationships
- GOAL Learn a mapping $f : V \to \mathcal{Y}$ predicting some graph property (at node/graph level)

Describe several models under a common framework - MPNN[8]

- Minimal requirements to design a message passing node aggregation function:
  - Permutation invariance
  - Independent on neighborhood size (1-hop)
  - Exploit same aggregation function among the nodes (gain generalization)
  - Linear complexity on the Edges
- Layerwise feature extraction fosters message propagation.

MPNN

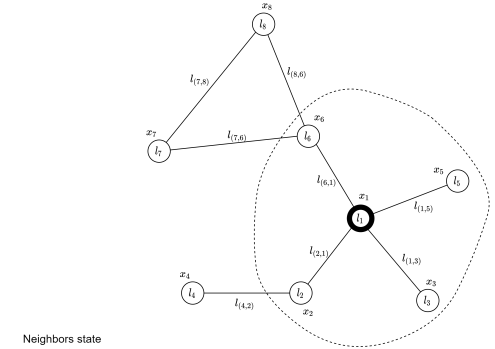
$$m_{i\leftarrow j}^{(\ell)} = \mathrm{MSG}_\ell\left(x_i^{(\ell-1)}, x_j^{(\ell-1)}, l_i,\ l_j, a_{i\leftarrow j}\right)$$
$$x_i^{(\ell)} = \mathrm{UP}_\ell\Big(\sum_{v_j\in\mathcal{N}_i^*} m_{i\leftarrow j}^{(\ell)}\Big)$$

[8] Justin Gilmer et al. "Neural message passing for quantum chemistry". In: *arXiv preprint arXiv:1704.01212* (2017).

- GNN*[9] apply a two-phase computation on each graph $G = (V, E)$
- Encoding (aggregate) phase compute a state vector x_v for each node in V by iteratively applying the state transition function f_a
  - yields a diffusion mechanism
  - executed until convergence of the state representation, i.e. until $x_n^{(t)} \simeq x_n^{(t-1)}, v \in V$.
  - Corresponds to the computation of the *fixed point* of f_a on the input graph.
  - garanteed if f_a is a contraction map - **Banach Fixed Point Theorem**
- Output (readout) phase exploits the final latent representations encoded by the states stored in each node to compute the model output with the output function f_r

[9] Franco Scarselli et al. "Graph neural networks for ranking web pages". In: *The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*. IEEE. 2005, pp. 666–672.

BACKGROUND: GRAPH NEURAL NETWORKS[10]



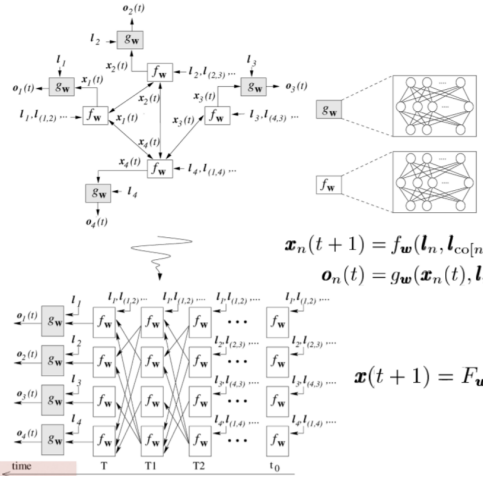
$$f_{a,v}^{(\text{SUM})} = \sum_{u \in ne[v]} h(x_u, l_u, l_v, l_{(u,v)} \mid \theta_h)$$

$$f_{a,v}^{(\text{AVG})} = \frac{1}{|ne[v]|} \sum_{u \in ne[v]} h(x_u, l_u, l_v, l_{(u,v)} \mid \theta_h)$$

[10] Franco Scarselli et al. “The graph neural network model”. In: *IEEE Transactions on Neural Networks* 20.1 (2008), pp. 61–80.

How we get the equilibrium points?

Relaxation to an equilibrium



$$\boldsymbol{x}_n(t+1) = f_{\boldsymbol{w}}(\boldsymbol{l}_n, \boldsymbol{l}_{\mathrm{co}[n]}, \boldsymbol{x}_{\mathrm{ne}[n]}(t), \boldsymbol{l}_{\mathrm{ne}[n]})$$

$$\boldsymbol{o}_n(t) = g_{\boldsymbol{w}}(\boldsymbol{x}_n(t), \boldsymbol{l}_n), \qquad n \in \boldsymbol{N}.$$

$$\boldsymbol{x}(t+1) = F_{\boldsymbol{w}}(\boldsymbol{x}(t), \boldsymbol{l})$$

Figure credit to Scarselli et al.

GNN*

$$x_v^{(t+1)} = f_a(x_{ne[v]}^{(t)}, l_{ne[v]}, l_v, a_{v \leftarrow ne[v]})$$
$$y_v = f_r(x_v^{(T)})$$

- Node update is repeated until convergence of the state representation, i.e. until $x_v^{(T)} \simeq x_v^{(T-1)}, v \in V$.
- Pros – Diffusion mechanism involving all the graph, not only a k-hop neighborhood (k layers).
- Cons – Epoch wise ad-hoc iterative convergence and BackProp.
- Hence, f_a reaches its fixed point, satisfying the constraint:

$$\forall v \in V, x_v = f_{a,v} \ .$$

The seminal Graph Neural Network[11] *model (GNN*) leverage an iterative convergence mechanism to compute the fixed-point of the state transition function, in order to allow the information diffusion among long-range neighborhoods of a graph.*

Is it possible to devise a local constraint-based scheme to avoid such costly procedure, maintaining these powerful aggregation capabilities?

[11] Scarselli et al., "Graph neural networks for ranking web pages".

- Avoid the explicit iterative computation of the fixed point for each epoch.
- Cast the learning problem as constrained optimization.
- Add free variables x_v (to be optimized) corresponding to the node states.

Problem

$$\min_{\Theta_{f_a}, \theta_{f_r}, X} \quad \sum_{v \in S} L(f_r(x_v), y_v)$$

$$\text{subject to} \quad \mathcal{G}(x_v - f_{a,v}) = 0, \quad \forall\, v \in V$$

With $\mathcal{G}(0) = 0$.

Enforce the constraint satisfaction in order to achieve:

- a new learning mechanism - search for node state representation that fulfill the constraints.
- Diffusion of information inside the graph.

Model evolution:

- Introduce a set of K states for each node $v \in V$, organized into K layers, $\{x_{v,k}, k = 0, \ldots, K-1\}$.
- Node states as additional input to the upper layer state transition function f_a^k.

$$\mathcal{G}(x_{v,k} - f_{a,v}^k) = 0, \quad \forall\, v \in V,\ \forall\, k \in [0, K-1]$$

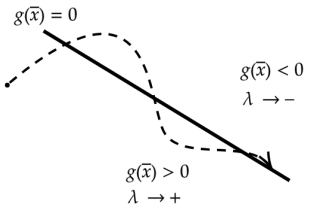
- Layered structure – increase the representational power (ConvGNN-like)
- At each and every layer, enforce the diffusion through the constraint satisfaction mechanism

We tackle the constrained problem introducing the Lagrangian

$$\mathcal{L}(\theta_{f_a}, \theta_{f_r}, X, \Lambda) = \sum_{v \in S} \left[L(f_r(x_v), y_v) + \sum_{k=0}^{K-1} \lambda_v^k \mathcal{G}\left(x_v - f_{a,v}\right) \right]$$

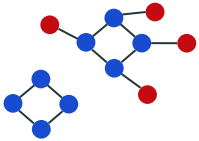
and then looking for saddle points of this function, in a gradient ascent-descent scheme[12].

$$\min_{\Theta_{f_a}, \theta_{f_r}, X} \max_{\Lambda} \mathcal{L}(\Theta_{f_a}, \theta_{f_r}, X, \Lambda)$$



[12] Platt and Barr, "Constrained differential optimization".

- Jointly optimize the model weights and the state representations without the need of separate ad-hoc optimization stages.
  - No more need of the epoch-wise convergence procedure
- Trade-off w.r.t. Local Propagation (LP) – f_a and f_r are BP-trainable models, the only additional variable are the node states x_v.
- Diffuse information layerwise by gradually enforcing the convergence of the state transition function to a fixed point (by virtue of the constraints).
- LP-GNNs strictly split deep feature extraction from the diffusion mechanism.
- Our scheme can be plugged into all SOTA models, leveraging powerfull aggregation functions empowered by diffusion over the graph.



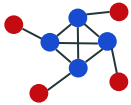


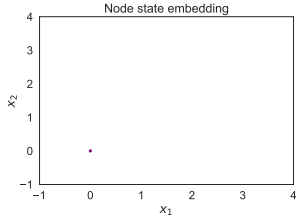
Table 3: Accuracies on the artificial datasets, for the proposed model (Lagrangian Propagation GNN - LP-GNN) and the standard GNN model for different settings.

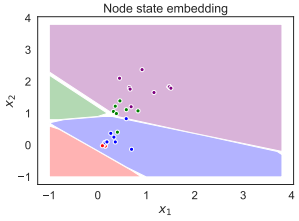
Model			Subgraph		Clique	
	$\mathcal{G}$	ϵ	*Acc(avg)*	*Acc(std)*	*Acc(avg)*	*Acc(std)*
LP-GNN	*abs*	0.00	96.25	0.96	88.80	4.82
		0.01	**96.30**	0.87	88.75	5.03
		0.10	95.80	0.85	85.88	4.13
	lin	0.00	95.94	0.91	84.61	2.49
		0.01	95.94	0.91	85.21	0.54
		0.10	95.80	0.85	85.14	2.17
	squared	-	96.17	1.01	**93.07**	2.18
GNN	-	-	95.86	0.64	91.86	1.12

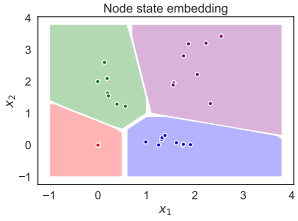
Table 4: Average and standard deviation of the classification accuracy on the graph classification benchmarks, evaluated on the test set, for different GNN models.

Datasets	IMDB-B	IMDB-M	MUTAG	PROT.	PTC	NCI1
# graphs	1000	1500	188	1113	344	4110
# classes	2	3	2	2	2	2
Avg # nodes	19.8	13.0	17.9	39.1	25.5	29.8
DCNN	49.1	33.5	67.0	61.3	56.6	62.6
PATCHYSAN	71.0 ± 2.2	45.2 ± 2.8	**92.6** ± 4.2	75.9 ± 2.8	60.0 ± 4.8	78.6 ± 1.9
DGCNN	70.0	47.8	85.8	75.5	58.6	74.4
AWE	74.5 ± 5.9	51.5 ± 3.6	87.9 ± 9.8	–	–	–
GRAPHSAGE	72.3 ± 5.3	50.9 ± 2.2	85.1 ± 7.6	75.9 ± 3.2	63.9 ± 7.7	77.7 ± 1.5
GIN	75.1 ± 5.1	**52.3** ± 2.8	89.4 ± 5.6	76.2 ± 2.8	64.6 ± 7.0	**82.7** ± 1.7
GNN	60.9 ± 5.7	41.1 ± 3.8	88.8 ± 11.5	76.4 ± 4.4	61.2 ± 8.5	51.5 ± 2.6
LP-GNN-SINGLE	71.2 ± 4.7	46.6 ± 3.7	90.5 ± 7.0	77.1 ± 4.3	64.4 ± 5.9	68.4 ± 2.1
LP-GNN-MULTI	**76.2** ± 3.2	51.1 ± 2.1	**92.2** ± 5.6	**77.5** ± 5.2	**67.9** ± 7.2	74.9 ± 2.4

- Completely removed node-attached features from the Karate Club dataset, in order to exploit only topological properties.
- No dependence on node features (l_v^0), the states are continuous representations of topological features of the nodes in the graph.







- GCN-like models need to stack multiple layers to achieve information diffusion.
- Some tasks suffice a shallow representation of the nodes, but still need a diffusion process to take place.
- LP-GNN naturally model this diffusion, without the need of deep architectures: the diffusion process is independent of the depth of the network.

Table 5: Average test accuracy on the IMDB-B dataset for LP-GNN and GIN model with state layers $K \in [1, 5]$.

Model	Number of State Layers			
	1	2	3	5
GIN[13]	52	72.6	72.7	75.1
LP-GNN	**71.2**	**73.7**	**73.9**	**76.2**

[13] Keyulu Xu et al. "How Powerful are Graph Neural Networks?" In: *International Conference on Learning Representations*. 2018.

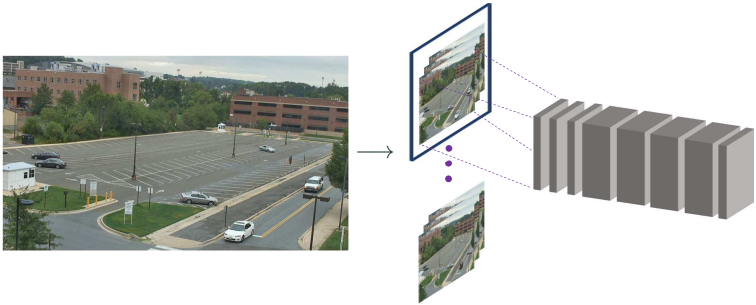
- Extend the approach to a wide variety of aggregation functions available in literature
- Adapted the proposed constraint-based scheme to use the aggregation functions of two popular models, i.e., GIN and GCN.

Table 6: LP diffusion in other GNN models. Average test accuracy on the IMDB-B dataset.

Evaluation	Model	Number of State Layers			
		1	2	3	5
Absolute	LP-GNN	65.3	73.7	73.9	76.2
	LP-GIN	71.3	73.2	73.0	73.6
	LP-GCN	73.4	73.5	73.8	73.7

3 – Constraint-based Mutual Information Computation in Salient Areas of Video Streams

A challenging problem that requires to go beyond the classic batch-mode setting.



Video Stream

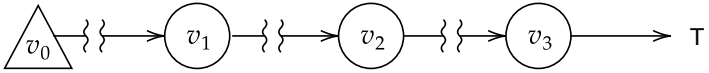
Pixelwise Deep-CNN

- InfoMax principle feature learning technique based on the maximization of the transferred information, in terms of Mutual Information (MI)
- Online Learning Avoid catastrofic forgetting in lifelong visual streams

Unsupervised learning from continuous visual streams is a challenging problem that cannot be naturally and efficiently managed in the classic batch-mode setting of computation. Hence, the task of transferring visual information in a truly online setting is hard.

Is it possible to overcome this issues by devising a local temporal method that forces consistency among predictions over time?

- MI maximization over time regard entropy approximations yielded at each time instant as components of the same temporal computational model.
  - Temporally local entropy approximations are subcomponents of the overall architecture, put into relation by soft-constraints
  - Enforce a temporal estimate that is not limited to the current frame
  - No more need to bufferize data over time



Process a stream of video frames $t \mapsto u(t)$ by a neural network

- weights and biases at time t are represented by the vector variable $w(t)$.

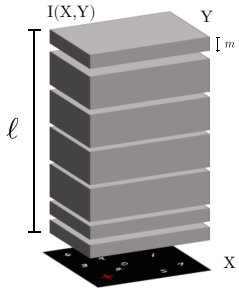
Learning as a variational problem - find a stationary point of the functional $w \mapsto \Gamma(w)$

$$\Gamma(w) := \int_0^T L(t, w(t), \dot{w}(t), \ddot{w}(t))\, dt = \int_0^T e^{\theta t}\big(K(\ddot{w}, \dot{w}) + U_{\mu_a}(w(t), u(t))\big)\, dt \tag{4}$$

- Kinetic term K enforces temporal regularization;
- Potential term U describes the temporal interaction with the environment (frame $u(t)$)
  - Unsupervised Learning – U is a temporally local and causal estimation of the Mutual Information.
  - Look for trajectories of the weights $t \mapsto w(t) \in \mathbf{R}^n$ leading to configurations with small potential energy, $U\big(w(t), u(t)\big) \approx 0$.

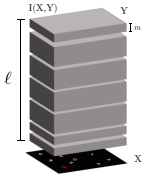
[14] Alessandro Betti, Marco Gori, and Stefano Melacci. "Learning visual features under motion invariance". In: *Neural Networks* 126 (2020), pp. 275–299.

- Neural architecture processes each frame and yields m pixel-wise predictions $(y_1, y_2, ..., y_m)$
  - Output y_i represent the probability of attaching to that pixel the i-th symbol
  - Discrete vocabulary composed by m symbols
  - Probabilities of the output symbol are conditioned on the actual weight configuration, the processed pixel and the frame.



Maximize the information transfer between the two random variables X and Y:

- X – associated with the input spatio-temporal probability distribution, with a realization given by ($x :=$ coordinate in the Retina(R), $t :=$ temporal instant, $u(t) :=$ frame)
- Y – associated with the probability distribution over the output symbols.



MI index over the video portion $[t_1, t_2]$,

$$I(X, Y; \omega; t_1, t_2) = H(Y; \omega; t_1, t_2) - H(Y|X; \omega; t_1, t_2)$$

We can obtain the entropy of the output symbols between time instants t_1 and t_2,

$$H(Y;\omega;t_1,t_2) = -\sum_{j=1}^{m} P_j(\omega,t_1,t_2)\log P_j(\omega,t_1,t_2) \tag{5}$$

being P the average output activation on the video portion between time instants t_1 and t_2,

$$P(\omega,t_1,t_2) \equiv \int_{t_1}^{t_2} \overline{P}(\omega,t)dt := \int_{t_1}^{t_2}\int_{R} p(\omega,x,u(t))\mu(x,t)dxdt, \tag{6}$$

- $\mu(x,t)$ is a *spatio-temporal* density (commonly assumed to be uniform in time and space)
- R is the aforementioned Retina.
- $\overline{P}(\omega,t)$ instantaneous probability of the output symbols computed via the total probability rule
  - An exact computation of the probability P would require to bufferize data

Online process that yields the frame probability predictions as a temporal computational model.

- additional auxiliary local variable $s(t)$, that is used to replace P – a temporal estimate which is not limited to the current frame.
- L is augmented with the differential soft-constraint

$$|\dot{s}(t) - \overline{P}(w(t), t)|^2 \tag{7}$$

- $s(t)$ enforced to approximate a probability estimate which is not limited to the current frame.

TEMPORAL DISTRIBUTION

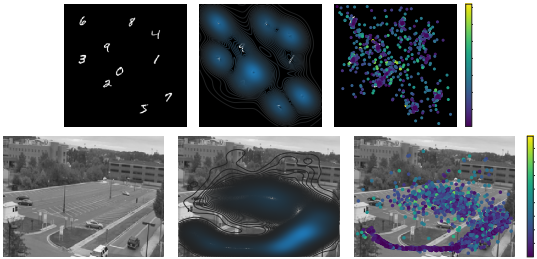
- VAR auxiliary variable to softly enforce a causal (depending only on previous-in-time quantities) approximation up to time t
- PLA temporal delta distribution peaked at time t – potentially leading to poorly informed updates
- AVG plain temporal Moving average.

$$\overline{P}(\omega, t) := \int_R p(\omega, x, u(t))\mu(x, t)dxdt, \tag{8}$$

Are all the input pixels equally important?

- ☐ Common assumption: *uniform probability density*.
- \+ Humans *discard lot of information* when observing a video!

G-Eymol[15] – the attended scanpath trajectory $t \mapsto a(t)$ emerges as a gravitational process.



Distribution $\mu_a(x, t) := g(x - a(t))$ with different implementations of g:

- Uniform on the whole frame UNI
- Restricted to the focus of attention (FOA or FOAW)

[15] Dario Zanca, Stefano Melacci, and Marco Gori. "Gravitational laws of focus of attention". In: *IEEE transactions on pattern analysis and machine intelligence* 42.12 (2019), pp. 2983–2995.

Stream	Test	UNI	FOA	FOAW
SparseMNIST	UNI	0.004	**0.144**	0.020
	FOA	0.103	**0.431**	0.229
	FOAW	0.144	**0.255**	0.157
Carpark	UNI	0.653	0.556	**0.745**
	FOA	0.678	0.639	**0.768**
	FOAW	0.653	0.601	**0.721**
Call	UNI	0.339	**0.556**	0.350
	FOA	0.430	**0.582**	0.492
	FOAW	0.442	**0.566**	0.457

Training over one of the spatial distributions UNI, FOA, FOAW, tested measuring the MI index in all the three density cases UNI, FOA, FOAW (best performing temporal distribution). *Filtering on the FOA trajectory improves the information transfer in all conditions.*

Stream	Test	PLA	AVG	VAR
SparseMNIST	UNI	0.006	0.054	**0.144**
	FOA	0.149	0.321	**0.431**
	FOAW	0.146	0.184	**0.255**
Carpark	UNI	0.422	**0.556**	0.315
	FOA	0.458	**0.639**	0.326
	FOAW	0.489	**0.601**	0.389
Call	UNI	0.259	**0.556**	0.369
	FOA	0.328	**0.582**	0.459
	FOAW	0.368	**0.566**	0.443

Learning using different temporal distributions, keeping fixed the spatial setting FOA. *Preserving the temporal information (AVG, VAR) leads to the best performances.*

Human-like FOA trajectory has a positive impact transferring information with respect to a random scanpath (RND), considering different DeepCNNs (S—3 layers, D—7 layers, DL—7 layers and more features).

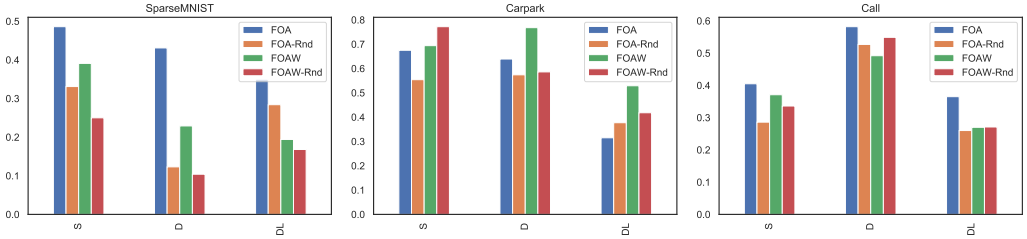


Figure 3: Comparison between models trained on a regular trajectory of the attention and on a random trajectory (suffix -RND), for architectures S, D, DL. Each bar is about a different training probability density, and the height of the bar is the test MI index along the regular FOA trajectory.

We devise a framework capable to decompose neural architectures into local components.

- Auxiliary variables and the unifying mathematical notion of constraint are leveraged to force internal knowledge onto the structure of the neural models.

Three different learning settings:

1. constraints among layers in feed-forward neural networks[16]
2. constraints among the states of neighboring nodes in Graph Neural Networks[17]
3. constraints among predictions over time[18].

[16] Marra et al., "Local Propagation in Constraint-based Neural Networks".
[17] Tiezzi et al., "A Lagrangian Approach to Information Propagation in Graph Neural Networks".
[18] Tiezzi et al., "Focus of Attention Improves Information Transfer in Visual Features".

LP and LP-GNN:

- Issue Memory complexity (Lifted Networks)
- Solution Reduce the lifted variables or predict the Multipliers via an ANN, using the same differential learning scheme
- Future works Investigate the parallelization capabilities (LP), alternative optimization schemes

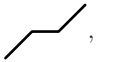
Online MI prediction:

- Issue Is MI a good criteria in order to develop informative features?
- Future works Test the learned features in downstream tasks, immerse the agent in a lifelong simulated environment

Thank you for listening!

Local Propagation in Neural Network Learning
by Architectural Constraints

Matteo Tiezzi

$$\mathcal{G}(a) = \textit{lin-}\varepsilon(a) := \max\{a, \varepsilon\} - \max\{-a, \varepsilon\} =$$  ,

$$\mathcal{G}(a) = \textit{abs-}\varepsilon(a) := \max\{|a| - \varepsilon, 0\} =$$  .

Table 7: The considered variants of the $\mathcal{G}$ function. By introducing ε-insensitive constraint satisfaction, we can inject into our hard-optimization scheme a controlled amount (i.e. ε) of unsatisfaction tolerance.

	lin	*lin-*ε	*abs*	*abs-*ε	*squared*
$\mathcal{G}(a)$	a	$\max(a, \varepsilon) - \max(-a, \varepsilon)$	$\lvert a \rvert$	$\max(\lvert a \rvert - \varepsilon, 0)$	a^2
Unilateral	×	×	✓	✓	✓
ε-insensitive	×	✓	×	✓	×

- Stabilize the learning process
- Improved generalization, injection and tolerance to noise.

$$\frac{\partial \mathcal{L}}{\partial W_\ell} = -\sum_{i=1}^{N} \left(\lambda_{\ell+1,i} \odot \mathcal{G}'_{\ell+1,i} \odot \sigma'(W_\ell x_{\ell,i})\right) x_{\ell,i}^T \tag{9}$$

$$\frac{\partial \mathcal{L}}{\partial x_{\ell,i}} = \lambda_{\ell,i} \odot \mathcal{G}'_{\ell,i} - W_\ell^T\left(\lambda_{\ell+1,i} \odot \mathcal{G}'_{\ell+1,i} \odot \sigma'(W_\ell x_{\ell,i})\right) \tag{10}$$

$$\frac{\partial \mathcal{L}}{\partial x_{H,i}} = \lambda_{H,i} \odot \mathcal{G}'_{H,i} + W_H^T \left(V'(y_i, y'_i)\right) \tag{11}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_{\ell,i}} = \mathcal{G}_{\ell,i} \tag{12}$$

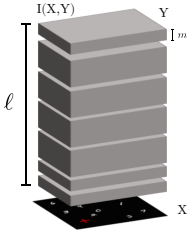
- RNN constraints:

$$\mathcal{G}\left(x_\ell^t - \sigma(W_{\ell-1}x_{\ell-1}^t + U_{\ell-1}x_\ell^{t-1})\right) = 0;$$

- Residual Network constraints:

$$\mathcal{G}\left(x_\ell - z(h(x_{\ell-1}) + f(W_{\ell-1}x_{\ell-1}))\right) = 0.$$

- Process a lifelong visual stream in a theoretically grounded online setting.
- Analyze the information transfer (Mutual Information index $I(X, Y)$) from the input X to the output Y of a Deep-CNN that performs pixel-wise predictions.



- Classical statistical machine learning - minimize an *empirical functional risk* based on random sampling

$$V(w) = \frac{1}{\ell} \sum_{k=1}^{\ell} v(w, x_k) \tag{13}$$

- Online learning – different intuition on the functional risk

$$V(w(t), t) = \sum_{t=1} v(w(t), x(t)) \tag{14}$$

- We assume there is a trajectory $t \mapsto u(t)$ in the pattern space that slides along a continuous temporal manifold - temporally coherent signal
- Parallel with Lagrangian mechanics, the online functional risk interpreted as a potential
- $V\big(w(t), u(t)\big)$ is referred to as the potential of the system defined by Lagrangian coordinates w
- We look for trajectories of the weights $t \mapsto w(t) \in \mathbf{R}^n$ that possibly lead to configurations with small potential energy, $V\big(w(t), u(t)\big) \approx 0$

We reformulated the problem in terms of the limit of the minima of a family of functionals $(\Gamma_\varepsilon)_{\varepsilon>0}$ (starting from Γ)[19].

- Update rules (ODEs) for the weights of the NN starting from fixed initial conditions on w^0, w^1 for the weights and their temporal derivatives, respectively:

$$\begin{cases} \alpha\ddot{w}(t) + \beta\dot{w}(t) + kw(t) + \nabla U_{\mu_a}(w(t), u(t)) = 0; \\ w(0) = w^0, \qquad \dot{w}(0) = w^1. \end{cases} \tag{15}$$

- Second order update rules instead of fourth order ODEs as we would have obtained from the stationarity condition on Γ.
- Full temporal causality – local computational model.

[19] Matthias Liero and Ulisse Stefanelli. "A new minimum principle for Lagrangian mechanics". In: *Journal of nonlinear science* 23.2 (2013), pp. 179–204.

- We exploited ready to use libraries (Tensorflow – Autograd for gradient computation)
- Sequential Forward adn Backward computations of gradient – GPUs benefit by the parallelization of matrix operation within each layer
- LP goes beyond that – local nature – easy to scale up with data parallelization strategies (many workers)[20]
- The ℓ-th computational unit needs to share the memory where some variables are stored with the $(\ell + 1)$-th and $(\ell - 1)$-th units.
- Parallelize computation both on the example dimension i and on the layers ℓ dimension.
- Need an ad-hoc parallel implementation
  - Central node computes the global loss function, update the weight and broadcast the updates to the nodes

[20] Gavin Taylor et al. "Training neural networks without gradients: A scalable admm approach". In: *International conference on machine learning*. 2016, pp. 2722–2731.

References

- Betti, Alessandro, Marco Gori, and Stefano Melacci. “Learning visual features under motion invariance”. In: *Neural Networks* 126 (2020), pp. 275–299.
- Dua, Dheeru and Efi Karra Taniskidou. *UCI Machine Learning Repository*. 2017. URL: `http://archive.ics.uci.edu/ml`.
- Gilmer, Justin et al. “Neural message passing for quantum chemistry”. In: *arXiv preprint arXiv:1704.01212* (2017).
- Gnecco, Giorgio et al. “Foundations of support constraint machines”. In: *Neural computation* 27.2 (2015), pp. 388–480.
- LeCun, Yann et al. “A theoretical framework for back-propagation”. In: *Proceedings of the 1988 connectionist models summer school*. Vol. 1. CMU, Pittsburgh, Pa: Morgan Kaufmann. 1988, pp. 21–28.
- Liero, Matthias and Ulisse Stefanelli. “A new minimum principle for Lagrangian mechanics”. In: *Journal of nonlinear science* 23.2 (2013), pp. 179–204.
- Marra, G. et al. “Local Propagation in Constraint-based Neural Networks”. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. 2020, pp. 1–8. DOI: `10.1109/IJCNN48605.2020.9207043`.
- Platt, John C and Alan H Barr. “Constrained differential optimization”. In: *Neural Information Processing Systems*. 1988, pp. 612–621.

- Scarselli, Franco et al. “Graph neural networks for ranking web pages”. In: *The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI’05)*. IEEE. 2005, pp. 666–672.
- Scarselli, Franco et al. “The graph neural network model”. In: *IEEE Transactions on Neural Networks* 20.1 (2008), pp. 61–80.
- Taylor, Gavin et al. “Training neural networks without gradients: A scalable admm approach”. In: *International conference on machine learning*. 2016, pp. 2722–2731.
- Tiezzi, Matteo et al. “A Lagrangian Approach to Information Propagation in Graph Neural Networks”. In: vol. 325. Giacomo, Giuseppe De. IOS Press, 2020, pp. 1539–1546. URL: *https://doi.org/10.3233/FAIA200262*.
- Tiezzi, Matteo et al. “Focus of Attention Improves Information Transfer in Visual Features”. In: *Advances in Neural Information Processing Systems* 33 (2020).
- Xu, Keyulu et al. “How Powerful are Graph Neural Networks?” In: *International Conference on Learning Representations*. 2018.
- Zanca, Dario, Stefano Melacci, and Marco Gori. “Gravitational laws of focus of attention”. In: *IEEE transactions on pattern analysis and machine intelligence* 42.12 (2019), pp. 2983–2995.