Divide and Conquer Networks (DiCoNet)

Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Alex Nowak   David Folqué   Joan Bruna
Center for Data Science, Courant Institute of Mathematical Sciences, NYU

SetUp

We consider tasks consisting in a mapping $\mathcal{T}$ between a variable-sized input set $X = \{x_1, \dots, x_n\}, x_j \in \mathcal{X}$ into an ordered set $Y = \{y_1, \dots, y_{m(n)}\}, y_j \in \mathcal{Y}$.

We are interested in tasks that are self-similar across scales, meaning that $\mathcal{T}$ can be decomposed as $\forall\, n\,, \forall X\,, |X| = n$:,

$$\mathcal{T}(X) = \mathcal{M}(\mathcal{T}(\mathcal{S}_1(X)), \dots, \mathcal{T}(\mathcal{S}_s(X)))\,,$$

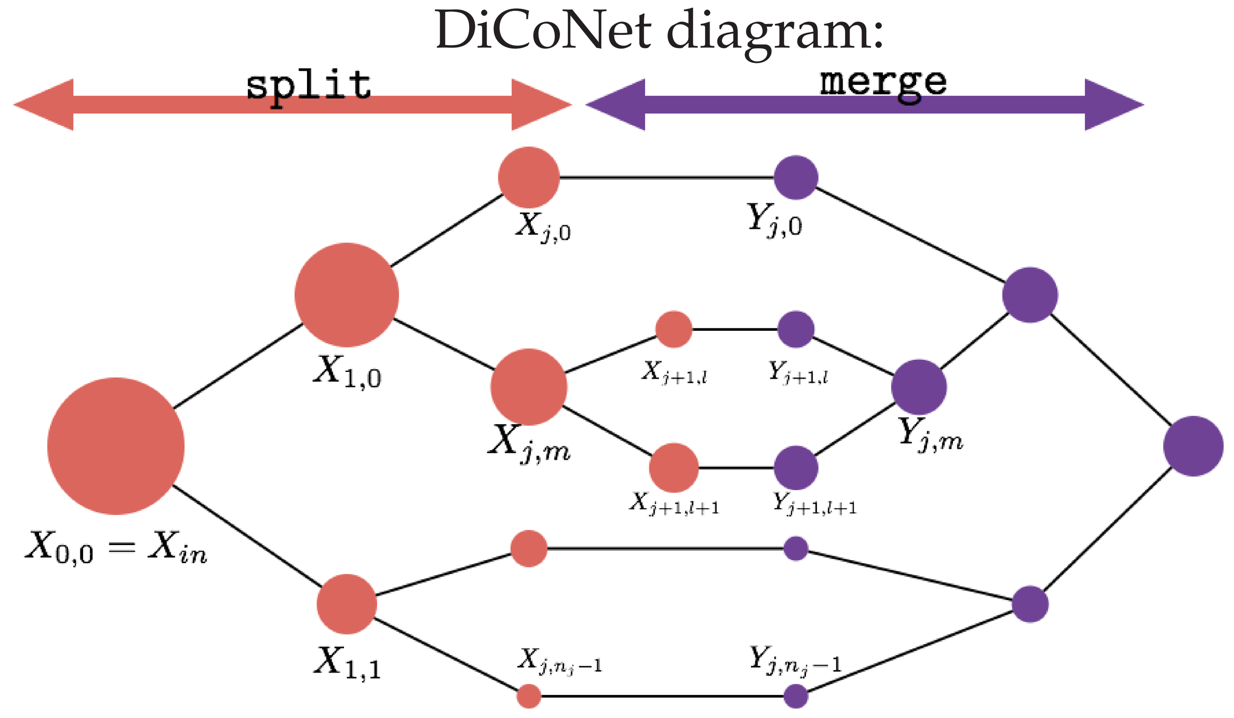
$$|\mathcal{S}_j(X)| < n\,,\ \cup_{j\leq s} \mathcal{S}_j(X) = X$$

where both $\mathcal{M}$ and $\mathcal{S} = (\mathcal{S}_1, \dots, \mathcal{S}_s)$ are *independent of* n.

Contributions

1. We introduce a new dynamic architecture that incorporates the inductive bias from recursive tasks.
2. We show that it **can be trained end-to-end with weak supervision**, and whose average computational **complexity can be optimized with gradient descent.**
3. We provide empirical evidence that the dynamic programming principle can be efficiently learnt on tasks such as planar convex-hull, hierarchical clustering, knapsack problem.

DiCoNet model



The DiCoNet is composed by two atomic blocks, namely **split** $\mathcal{S}_\theta$ and **merge** $\mathcal{M}_\phi$.

- *Split*: Splits the input X recursively by sampling from binary probabilities $p(z \mid X)$. It is modeled with a *Set2Set* or *Graph Neural-Net*. The recursive stochastic procedure results in a probability distribution over hierarchical partitions of X $\mathcal{P}(X) \sim \mathbf{S}_\theta(X)$
- *Merge*: Merges the input recursively traversing upwards the tree associated to $\mathcal{P}(X)$. It can be modeled with a *PtrNet* [2].

Training

Given a training set of pairs $\{(X^l, Y^l)\}_{l\leq L}$, the DiCoNet optimizes the following loss:

$$\mathcal{L}(\theta, \phi) = \frac{1}{L}\sum_{l\leq L} \mathbb{E}_{\mathcal{P}(X)\sim \mathbf{S}_\theta(X)} \log p_\phi(Y^l \mid \mathcal{P}(X^l))$$

with $p_\phi(Y \mid \mathcal{P}(X)) = \mathbf{M}_\phi(\mathcal{P}(X))$

- *Merge gradients*: As a vanilla PtrNet. The output stochastic matrix over indexes is replaced by the product of all the output stochastic matrices across scales (composing the permutations).

$$\nabla_\phi \mathcal{L}(\theta, \phi) = \frac{1}{L}\sum_{l\leq L} \mathbb{E}_{\mathcal{P}(X)\sim \mathbf{S}_\theta(X)} \nabla_\phi \log p_\phi(Y^l \mid \mathcal{P}(X^l))$$

- *Split gradients*: Approximated by samples using REINFORCE. Merge loss is used as a cost (or minus reward) for the split phase.

$$\nabla_\theta \mathbb{E}_{\mathcal{P}(X)\sim \mathbf{S}_\theta(X)} F(\mathcal{P}(X)) = \mathbb{E}_{\mathcal{P}(X)\sim \mathbf{S}_\theta(X)} F(\mathcal{P}(X)) \nabla_\theta \log f_\theta(\mathcal{P}(X))$$

where $F(\mathcal{P}(X)) = \log p_\phi(Y^l \mid \mathcal{P}(X^l))$ and

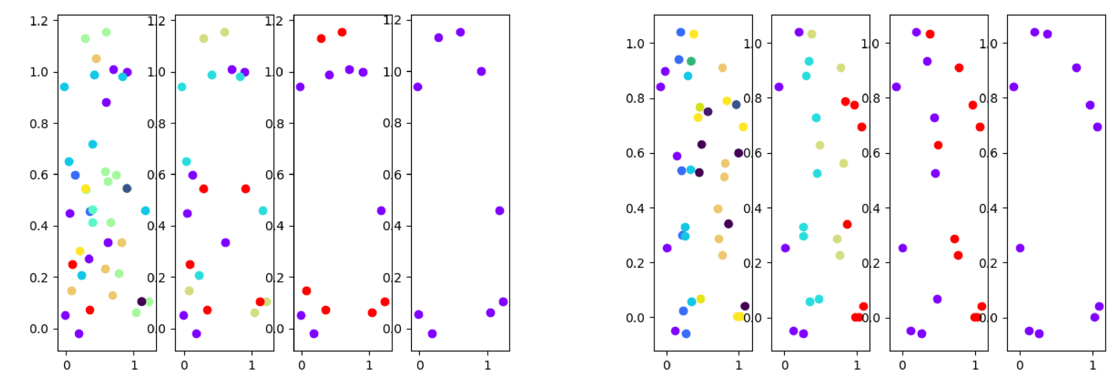
$$\log f_\theta(\mathcal{P}(X)) = \sum_{j=1}^{J} \sum_{k\leq n_j} \sum_{m\leq |X_{j,k}|} \log p_\theta(z_{m,j,k} \mid X_{j-1,k/2})\,.$$

Results

- PLANAR CONVEX HULL

Given a set of n points in the plane, find the ordered sequence of extremal points of the convex hull.

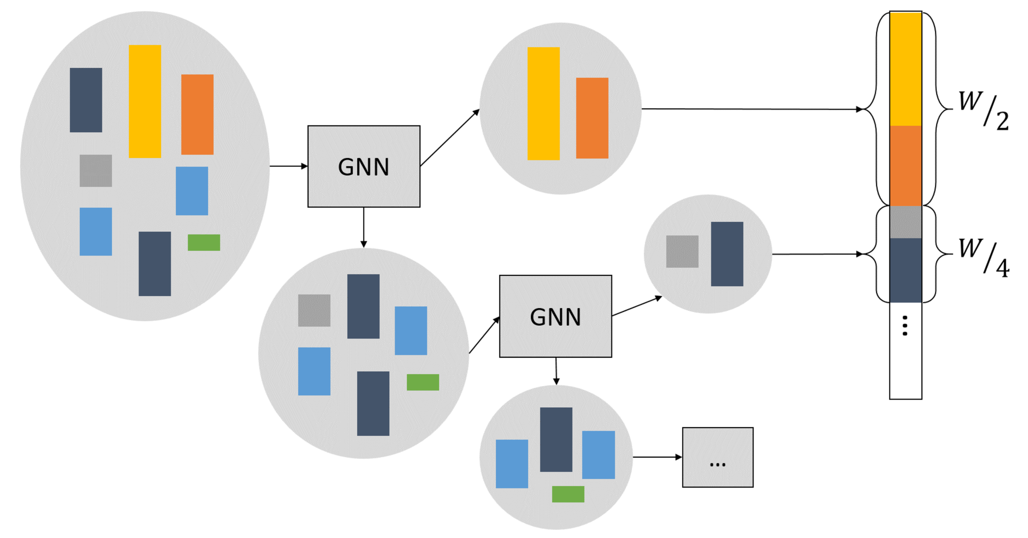
	n=25	n=50	n=100	n=200
Baseline	81.3	65.6	41.5	13.5
DiCoNet + Random Split	59.8	37.0	23.5	10.29
DiCoNet	88.1	83.7	73.7	52.0
DiCoNet + Split Reg	**89.8**	**87.0**	**80.0**	**67.2**



- CLUSTERING

Group n elements into k clusters. The DiCoNet will work well for problems with hierarchical structure.

	Gaussian (d=2)			Gaussian (d=10)			CIFAR-10 patches		
	k=4	k=8	k=16	k=4	k=8	k=16	k=4	k=8	k=16
Baseline / Lloyd	**1.8**	3.1	3.5	**1.14**	**5.7**	12.5	**1.02**	1.07	1.41
/ Lloyd	2.3	**2.1**	**2.1**	1.6	6.3	**8.5**	1.04	**1.05**	**1.2**
Baseline / Rec. Lloyd	**0.7**	1.5	1.7	**0.15**	**0.65**	1.25	**1.01**	1.04	1.21
/ Rec. Lloyd	0.9	**1.01**	**1.02**	0.21	0.72	**0.85**	1.02	**1.02**	**1.07**



- KNAPSACK

Given a set of n items, each with weight $w_i \geq 0$ and value $v_i \in R$, the 0-1 Knapsack problem consists in selecting the subset of the input set that maximizes the total value, so that the total weight does not exceed a given limit:

$$\begin{array}{ll} maximize_{x_i} & \sum_i x_i v_i \\ subject\ to & x_i \in \{0,1\},\ \sum_i x_i w_i \leq W\ . \end{array}$$

	n=50			n=100			n=200		
	cost	ratio	splits	cost	ratio	splits	cost	ratio	splits
Baseline	19.82	1.0063	0	38.79	1.0435	0	74.71	1.0962	0
DiCoNet	**19.85**	**1.0052**	3	**40.23**	**1.0048**	5	81.09	1.0046	7
Greedy	19.73	1.0110	-	40.19	1.0057	-	**81.19**	**1.0028**	-
Optimum	*19.95*	*1*	-	*40.42*	*1*	-	*81.41*	*1*	-

References

[1] Vinyals, Oriol and Bengio, Samy and Kudlur, Manjunath Order matters: Sequence to sequence for sets arXiv preprint arXiv:1511.06391

[2] Vinyals, Oriol and Fortunato, Meire and Jaitly, Navdeep Pointer networks Advances in Neural Information Processing Systems

Conclusions

We have presented a novel neural architecture that can discover and exploit scale invariance in discrete algorithmic tasks, and can be trained with weak supervision. Our model learns how to split large inputs recursively, then learns how to solve each subproblem and finally how to merge partial solutions.

Source Code

The source code to reproduce the experiments will be available soon at: https://github.com/alexnowakvila/DiCoNet