8 Comments June 22, 2024

(Akbar: Behrooz, could you add a detailed version of the construction with master node. The previous works have considered algorithmic alignment for “local” algorithms such as Bellman-Ford. It is interesting to show, by introducing the master node, we still have algorithmic alignment for “NOT local” algorithms such as the greedy algorithm.)

9 Comments June 28, 2024

We discussed set function extensions and the possibility of optimizing the extension using neural networks.

10 Decomposition into fixed cardinality sets

Suppose we are given a $\mathbf{x} \in [0,1]^n$ with $||x||_1 = k$. We can rewrite $\mathbf{x}$ as the result of a recursively constructed distribution with sets S such that $|S| = k$. More specifically at timestep t we have

$$\mathbf{x}_t = p_{\mathbf{x}_t}(S)\mathbf{1}_S + (1 - p_{\mathbf{x}_t}(S))\mathbf{x}_{t+1}. \tag{2}$$

In order to define the composition we need a way to determine the coefficients $p_{\mathbf{x}_t}(S)$. A crucial requirement is that $p_{\mathbf{x}_t}(S) = g(\mathbf{x}_t)$ where g is differentiable with respect to $\mathbf{x}$.

Another important consideration for the soundness of the procedure is the need to ensure that $\mathbf{x}_{t+1}$ is also contained within the hypercube. Since any $\mathbf{1}_S$ is a corner of the hypercube and (2) is a convex combination of points, then that would guarantee that the iterative process produces points that remain in the hypercube.

From (2) we express the point at the next iteration as

$$\mathbf{x}_{t+1} = \frac{\mathbf{x}_t - p_{\mathbf{x}_t}(S)\mathbf{1}_S}{(1 - p_{\mathbf{x}_t}(S))}$$

Let x_i be the i-th entry of $\mathbf{x}_t$. To guarantee that $\mathbf{x}_{t+1}$ is in the hypercube we require that

$$x_i - p_{\mathbf{x}_t}(S) \geq 0 \text{ for } i \in S,$$
$$\frac{x_i}{(1 - p_{\mathbf{x}_t}(S))} \leq 1 \text{ for } i \notin S.$$

The first condition focuses on the coordinates of $\mathbf{x}_{t+1}$ that correspond to the set S that we have picked to subtract from $\mathbf{x}_t$. It requires that the corresponding coordinates remain in the hypercube after subtracting, i.e., we don’t subtract too large of a number. The second condition looks at the coordinates that are not affected by the subtraction (i.e. do not correspond to coordinates of the set S). Since those will be rescaled, we need to ensure that the division we do to those coordinates does not make them escape the hypercube, i.e., that we are not dividing by too large of a number.

These two conditions imply the following

$$p_{\mathbf{x}_t}(S) \leq \min_{i \in S} x_i, \tag{3}$$
$$p_{\mathbf{x}_t}(S) \leq 1 - \max_{i \notin S} x_i. \tag{4}$$

Therefore, we can set

$$p_{\mathbf{x}_t}(S) = \min(\min_{i \in S} x_i, 1 - \max_{i \notin S} x_i), \tag{5}$$

For the fixed cardinality decomposition to work, we may sort the vector $\mathbf{x}$ and pick S according to its top k ranking coordinates at each step (Akbar: what if $\mathbf{x}_t$, for some t, has less than k non-zero entries?). At each step, there will be one more 0 or 1 by construction (we either kill a coordinate when we subtract or we scale one to 1 when we divide). (Akbar: How sets S_t and S_{t+1} changes ? are they similar to each other?)

Warning: To ensure Lipschitzness of the extension we need to ensure that the mapping from $\mathbf{x}$ to the vector of probabilities as described in 5 is Lipschitz.

Question 10.1. Suppose $\mathbf{x}$ is decomposed according to the decomposition and we have $\mathbf{x} = \sum_t p_{\mathbf{x}_t}(S_t)\mathbf{1}_{S_t}$. Let $F(\mathbf{x}) = \sum_t p_{\mathbf{x}_t}(S_t) f(S_t)$. What is

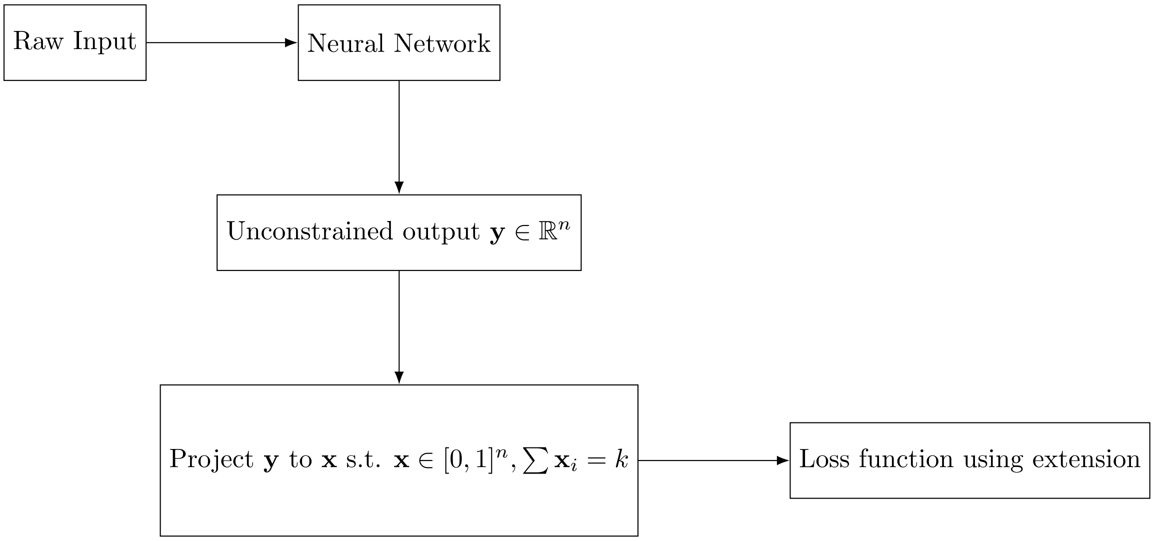
$$\frac{\partial F(\mathbf{x})}{\partial \mathbf{x}(e)} = ?$$

Question 10.2. The above extension, for a given $\mathbf{x}$, gives rise to a probability distribution over subsets of size k. Let R be a random subset sampled according to this probability distribution then what is

$$P(a \in R)$$

for an element a in the ground set? (Akbar: In general it could be zero, but maybe for "good" elemnts it is not zero and we have a lower bound.)(Akbar: It seems we need to answer this in order to follow the proofs of Lemma 1 in [1].)

11 Pipeline without algorithmic alignment [August 15]


Raw Input
Neural Network
Unconstrained output $\mathbf{y} \in \mathbb{R}^n$
Project $\mathbf{y}$ to $\mathbf{x}$ s.t. $\mathbf{x} \in [0,1]^n, \sum \mathbf{x}_i = k$
Loss function using extension

11.1 Projection into feasible set

We follow the same idea as [7]. Let $\mathbf{y} \in \mathbb{R}^n$ be the output of a neural network. We project $\mathbf{y}$ into $\mathbf{x}$ such that $\mathbf{x} \in [0,1]^n$ and $\sum_{i=1}^n \mathbf{x}_i = k$. Define matrices $\mathbf{S}, \mathbf{W}$ to be

$$\mathbf{W} = \begin{bmatrix} y_1 & y_2 & \dots & y_n & \beta \\ \beta & \beta & \dots & \beta & \beta \end{bmatrix} \quad \text{where } \mathbf{W} \in \mathbb{R}^{2\times(n+1)}, \beta \in \mathbb{R} \tag{6}$$

$$\mathbf{S} = \exp\left(\frac{\mathbf{W}}{\tau}\right) \quad \text{where } \mathbf{S} \in \mathbb{R}^{2\times(n+1)} \tag{7}$$

Now the the cardinality constraint is captured by

$$\mathbf{u}_e = \begin{bmatrix} 1 & 1 & \dots & 1 & 0 \end{bmatrix} \in \mathbb{R}^{n+1}, \tag{8}$$

$$\mathbf{v}_e = \begin{bmatrix} k & n-k \end{bmatrix} \in \mathbb{R}^2. \tag{9}$$

Question 11.1. Does this approach work for matroids ? In a matroid polytope the number of constraints is exponential.(or only the base sets are required?)

$$\mathcal{P}(\mathcal{M}) = \{\mathbf{x} \geq \mathbf{0} : \mathbf{x}(S) \leq r_{\mathcal{M}}(S);\ \forall S \subseteq E\}.$$