Supplementary material of paper *Supervised learning of convex piecewise linear approximations of optimization problems*

Laurine Duchesne, Quentin Louveaux and Louis Wehenkel

University of Liege - Dept of EE&CS
Liege - Belgium

1 Introduction

This document provides a supplementary material for [1]. It is organized as follows.

In section 2, we provide the mathematical proof that the approximated problem

$$\min\{f(x)\} \text{ s.t. } x \in \tilde{\mathcal{D}}_\lambda,$$

with $f(x)$ piecewise linear and convex and $\tilde{\mathcal{D}}_\lambda$ a convex feasible set approximation built according to the method described in [1], can be written as a linear program, in the case of piecewise linear, convex, and non-decreasing activation functions.

In section 3, we describe how we adapted the loss function of the ICNN in order to improve the quality of the approximation in regions close to the optimum.

2 Proof: $\min_{x \in \tilde{\mathcal{D}}_\lambda} f(x)$ can be reduced to a linear program

We consider a convex ICNN classifier used to approximate the feasible set $\mathcal{D}$ of an optimization problem. Let us show that if the objective function $f(x)$ is piecewise linear and convex, and if all the activation functions g_i used in the ICNN are piecewise linear, convex, and non-decreasing functions (such as $ReLU(x) = \max(0, x)$, or $leaky-ReLU(x) = \max(0.01x, x)$), the resulting optimization problem

$$\min\{f(x)\} \text{ s.t. } x \in \tilde{\mathcal{D}}_\lambda,$$

reduces to a linear program. We first notice that if $f(x)$ is piecewise linear and convex, then

$$\min\{f(x)\} \text{ s.t. } x \in \tilde{\mathcal{D}}_\lambda$$

may also be rewritten as

$$\min\{z\} \text{ s.t. } z \geq a_j^T x + b_j, \forall j = 1, \dots, l; x \in \tilde{\mathcal{D}}_\lambda,$$

where the set of inequalities $z \geq a_j^T x + b_j, \forall j = 1, \dots, l$ represent the epigraph of $f(x)$. We thus need only to prove that $\tilde{\mathcal{D}}_\lambda$ may itself also be represented by a set of linear (in)equalities.

For simplicity, we prove this in the case of ReLU activation functions but the result can be extended to any other choice of piecewise linear, convex and non-decreasing activation functions.

First, let us consider the domain P which is the set of points $(x, z_0, \ldots, z_{k-1})$ such that

$$z_0 = 0, \tag{1}$$

and

$$\forall i = 0, \ldots k-2 : z_{i+1} = \max(W_i^z \times z_i + W_i^x \times x + b_i, 0) \tag{2}$$

and

$$(W_{k-1}^{z,1} - W_{k-1}^{z,0}) \times z_{k-1} + (W_{k-1}^{z,1} - W_{k-1}^{x,0}) \times x + b_{k-1}^1 - b_{k-1}^0 \leq \lambda. \tag{3}$$

Note that the projection on x of the domain P, $proj_x(P)$, corresponds to $\tilde{\mathcal{D}}_\lambda$. When the max function is replaced with a set of equations, the equivalent formulation of the constraints in P becomes

$$\begin{aligned}
&(W_{k-1}^{z,1} - W_{k-1}^{z,0}) \times z_{k-1} + (W_{k-1}^{z,1} - W_{k-1}^{x,0}) \times x + b_{k-1}^1 - b_{k-1}^0 \leq \lambda, && && (4)\\
&z_{i+1} = 0 + s_i^0 && \text{for } i = 0, \ldots, k-2 && (5)\\
&z_{i+1} = W_i^z \times z_i + W_i^x \times x + b_i + s_i^z && \text{for } i = 0, \ldots, k-2 && (6)\\
&s_i^0 s_i^z = 0 && \text{for } i = 0, \ldots, k-2 && (7)\\
&s_i^0, s_i^z \geq 0 && \text{for } i = 0, \ldots, k-2 && (8)\\
&z_0 = 0, && && (9)
\end{aligned}$$

where we introduce slack variables $s_i^{\cdot}$ to express that z_{i+1} is either equal to 0 or to $W_i^z \times z_i + W_i^x \times x + b_i$ for $i = 0, \ldots, k-2$.

All the constraints in P are linear, except equation (7). However, we can show that this nonlinear equation is not necessary regarding our purpose because the projection on x of a relaxed version of the domain P, that we call Q and for which all the constraints defining Q correspond to a set of linear equations, is also equal to $\tilde{\mathcal{D}}_\lambda$.

Lemma 1. *We are given the parameters of an ICNN using ReLU as activation functions and learnt to build a convex approximation $\tilde{\mathcal{D}}_\lambda$ of the feasible set $\mathcal{D}$. Consider the set P defined as*

$$\begin{aligned}
P = \big\{&(x, z_1, \ldots, z_{k-1}, s_0^0, \ldots, s_{k-2}^0, s_0^z, \ldots, s_{k-2}^z) | \\
&(W_{k-1}^{z,1} - W_{k-1}^{z,0}) \times z_{k-1} + (W_{k-1}^{z,1} - W_{k-1}^{x,0}) \times x + b_{k-1}^1 - b_{k-1}^0 \leq \lambda, && && (10)\\
&z_{i+1} = 0 + s_i^0 && \forall i = 0, \ldots, k-2, && (11)\\
&z_{i+1} = W_i^z \times z_i + W_i^x \times x + b_i + s_i^z && \forall i = 0, \ldots, k-2, && (12)\\
&s_i^0 s_i^z = 0 && \forall i = 0, \ldots, k-2, && (13)\\
&s_i^0, s_i^z \geq 0 && \forall i = 0, \ldots, k-2, && (14)\\
&z_0 = 0\big\}. && && (15)
\end{aligned}$$

The set Q, which is defined as P but where constraint (13) is removed, i.e. defined as

$$
\begin{aligned}
& (W^{z,1}_{k-1} - W^{z,0}_{k-1}) \times z_{k-1} + (W^{z,1}_{k-1} - W^{x,0}_{k-1}) \times x + b^1_{k-1} - b^0_{k-1} \leq \lambda, && (16)\\
& z_{i+1} = 0 + s^0_i \qquad \forall i = 0, \dots, k-2, && (17)\\
Q = & \{(x, z_1, \dots, z_{k-1}, s^0_0, \dots, s^0_{k-2}, s^z_0, \dots, s^z_{k-2}) | && (18)\\
& z_{i+1} = W^z_i \times z_i + W^x_i \times x + b_i + s^z_i \qquad \forall i = 0, \dots, k-2, && (19)\\
& s^0_i, s^z_i \geq 0 \qquad \forall i = 0, \dots, k-2, && (20)\\
& z_0 = 0\}, && (21)
\end{aligned}
$$

is therefore a relaxation of P. We now prove that the projection on x of the set P is equal to the projection on x of the set Q:

$$proj_x(P) = proj_x(Q). \qquad (22)$$

Proof.

1. $proj_x(P) \subseteq proj_x(Q)$ is obvious since Q is a relaxation of P.
2. We now prove $proj_x(P) \supseteq proj_x(Q)$. Consider $(x, z, s^0, s^z) \in Q$. If $(x, z, s^0, s^z) \in P$, the result follows. Assume now that $(x, z, s^0, s^z) \notin P$. It is always possible, by following Algorithm 1, to build from a set of points (x, z, s^0, s^z) in Q but not in P, a set of points $(x, \bar{z}, \bar{s}^0, \bar{s}^z)$ in P with the same x, which proves the result.

Algorithm 1: Update $(x, z, s^0, s^z) \in Q$ such that $(x, \bar{z}, \bar{s}^0, \bar{s}^z) \in P$.

```
Result: $(x, \bar{z}, \bar{s}^0, \bar{s}^z)$ in $P$
// Initialization
$(x, \bar{z}, \bar{s}^0, \bar{s}^z) = (x, z_1, \dots, z_{k-1}, s^0_0, \dots, s^0_{k-2}, s^z_0, \dots, s^z_{k-2})$ ;
for $j = 0 : k-2$ do
    if $\bar{s}^0_j > 0$ and $\bar{s}^z_j > 0$ then
        $\Delta := \min(\bar{s}^0_j, \bar{s}^z_j)$;
        $\hat{s}^0_j = \bar{s}^0_j - \Delta$;
        $\hat{s}^z_j = \bar{s}^z_j - \Delta$;
        $\hat{z}_{j+1} = 0 + \hat{s}^0_j$ ;
        // or $\hat{z}_{j+1} = W^z_j \times \bar{z}_j + W^x_j \times x + b_j + \hat{s}^z_j$
        if $j < k-2$ then
            $\hat{s}^z_{j+1} = \bar{s}^z_{j+1} + W^z_{j+1}(\bar{z}_{j+1} - \hat{z}_{j+1})$;
            $(\bar{z}_{j+1}, \bar{s}^0_j, \bar{s}^z_j, \bar{s}^z_{j+1}) = (\hat{z}_{j+1}, \hat{s}^0_j, \hat{s}^z_j, \hat{s}^z_{j+1})$;
        else
            $(\bar{z}_{j+1}, \bar{s}^0_j, \bar{s}^z_j) = (\hat{z}_{j+1}, \hat{s}^0_j, \hat{s}^z_j)$;
        end
    end
end
```

Let us show that $(x, \bar{z}, \bar{s}^0, \bar{s}^z)$, built from $(x, z, s^0, s^z) \in Q$ with Algorithm 1, belongs to P. For that we proceed iteratively and we show that at each iteration, the updated vector still belongs to Q. At the end of the iterations, since by construction $(x, \bar{z}, \bar{s}^0, \bar{s}^z)$ satisfies constraint (13), $(x, \bar{z}, \bar{s}^0, \bar{s}^z) \in P$.

Let j be the smallest index such that $s^0_j > 0$ and $s^z_j > 0$.

Consider the point $(x, \bar{z}, \bar{s}^0, \bar{s}^z)$, obtained after j iterations with Algorithm 1. We can readily see that this point belongs to Q. Indeed, compared to the point $(x, z, s^0, s^z) \in Q$, the only constraint with a different realization of the left-hand-side or right-hand-side is constraint (19) for $i = j$ and $i = j + 1$. The constraint is nevertheless still valid in both cases:

- $i = j$: By definition of $\bar{z}_{j+1}$, the constraint holds with equality.
- $i = j + 1$: We have that $\bar{z}_{j+1} < z_{j+1}$ since $\bar{s}^{\cdot}_j < s^{\cdot}_j$ by construction. Furthermore, given that $W^z_{j+1} > 0$, $W^z_{j+1} \times z_{j+1} > W^z_{j+1} \times \bar{z}_{j+1}$. Therefore,

  $$z_{j+2} \geq W^z_{j+1} \times z_{j+1} + W^x_{j+1} \times x + b_{j+1} > W^z_{j+1} \times \bar{z}_{j+1} + W^x_{j+1} \times x + b_{j+1}$$

  and the constraint holds.

Note that in case $j = k - 2$, the right-hand-side of constraint (16) is also impacted. However, since $(W^{z,1}_{k-1} - W^{z,0}_{k-1}) \geq 0$,

$$(W^{z,1}_{k-1} - W^{z,0}_{k-1})\bar{z}_{k-1} < (W^{z,1}_{k-1} - W^{z,0}_{k-1})z_{k-1} \leq \lambda,$$

where the last inequality holds because $(x, z, s^0, s^z) \in Q$. Therefore, $(x, \bar{z}, \bar{s}^0, \bar{s}^z) \in Q$.

Observe that, if $(x, \bar{z}, \bar{s}^0, \bar{s}^z) \notin P$, there exists $\bar{j} > j$ such that $s^0_{\bar{j}} > 0$ and $s^z_{\bar{j}} > 0$. We can again decrease the value of $z_{\bar{j}+1}$ in order to make one of the slacks tight. We obtain the result by applying the procedure repeatedly. Since there is a finite number of indices, the procedure can be applied at most $k - 1$ times until we obtain $(x, \bar{z}, \bar{s}^0, \bar{s}^z) \in P$.

We can thus state that $proj_x(P) = proj_x(Q)$. □

Note that this lemma can be extended to other activation functions, as long as they are convex, piecewise linear and non-decreasing. It can thus be applied to leaky-ReLU activation functions. It is also still valid at the limit, for an infinite number of pieces and so it is valid for any smooth convex and non-decreasing activation function.

Consequently to Lemma 1, given an objective function that only depends on x, we have the following result.

Corollary 2. *Given an ICNN using ReLU as activation functions and learnt to build a convex approximation $\tilde{\mathcal{D}}_\lambda$ of the domain $\mathcal{D}$,*

$$\begin{aligned}
&\min f(x) \\
s.t.\ &(W^{z,1}_{k-1} - W^{z,0}_{k-1}) \times z_{k-1} + (W^{z,1}_{k-1} - W^{x,0}_{k-1}) \times x + b^1_{k-1} - b^0_{k-1} \leq \lambda \\
&z_{i+1} = \max(W^z_i \times z_i + W^x_i \times x + b_i, 0) \qquad \textit{for } i = 0, \dots, k-2 \\
&z_0 = 0
\end{aligned} \tag{23}$$

is equivalent to

$$\begin{aligned}
&\min f(x) \\
s.t.\ &(W^{z,1}_{k-1} - W^{z,0}_{k-1}) \times z_{k-1} + (W^{z,1}_{k-1} - W^{x,0}_{k-1}) \times x + b^1_{k-1} - b^0_{k-1} \leq \lambda \\
&z_{i+1} \geq 0 \qquad \textit{for } i = 0, \dots, k-2 \\
&z_{i+1} \geq W^z_i \times z_i + W^x_i \times x + b_i \qquad \textit{for } i = 0, \dots, k-2 \\
&z_0 = 0.
\end{aligned} \tag{24}$$

Indeed, the domain P is equivalent to the feasible set of problem (23) while Q is equivalent to the feasible set of problem (24). Since the projection of these two sets on the x space is equivalent, *i.e.* $proj_x(P) = proj_x(Q)$, then the feasible set of solutions regarding x and thus the optimal solution x^* of the two optimization problems are the same.

Therefore, $\min_{x \in \tilde{\mathcal{D}}_\lambda} f(x)$ reduces to a linear program if the objective function $f(x)$ is (piecewise) linear (and convex), and if all the activation functions g_i used in the ICNN are piecewise linear, convex, and non-decreasing functions.

3 Experiments: considering an objective function values when learning the ICNN

We detail in this section how the training-loss function can be adapted to consider the value of the optimization objective function when training the ICNN classifier, in order to improve the quality of the approximation $\tilde{\mathcal{D}}_0$ in regions close to the optimum. Notice that in order to implement these methods, we assume that for each element i of the training set, we also know the value $f_i = f(x_i)$ of the objective function (in addition to the input x_i and the output y_i indicating constraint satisfaction of x_i w.r.t. $\mathcal{D}$).

In order to force the learnt approximation $\tilde{\mathcal{D}}_0$ to be better in regions where the constrained optimum might be located, we give a higher weight in the loss function to input-output samples with smaller associated values of f. Thus, the loss function for an observation x of class y and corresponding to an objective value f would be given by:

$$\text{loss}(\theta, x, y, f) = w_{y,f} \times \left[-\log \left(\frac{\exp(h_y(\theta, x))}{\exp(h_0(\theta, x)) + \exp(h_1(\theta, x))} \right) \right],$$

where $w_{y,f}$ would depend both on the true class of the sample and on the value of the objective function for this sample.

We tested two methods to compute the weights $w_{y,f}$.

The first method is such that the weights vary linearly between two bounds with the objective function:

$$w_{y,f} = w_y^{\min} + \frac{f_y^{\max} - f}{f_y^{\max} - f_y^{\min}}(w_y^{\max} - w_y^{\min}),$$

where $f_y^{\max}$ and $f_y^{\min}$ are respectively the maximum and minimum values of the objective function among the training samples with label y, and $w_y^{\min}$ and $w_y^{\max}$ for $y \in \{0,1\}$ are four hyper-parameters that can be tuned.

The other method that we tested is such that the weights $w_{y,f}$ take only two values per class y, a high value when the (x, y, f) tuple corresponds to a "small enough" value of the objective function $f(x)$, and a lower value otherwise:

$$w_{y,f} = \mathbb{1}_{A_y}(f) w_y^{\max} + (1 - \mathbb{1}_{A_y}(f)) w_y^{\min},$$

where $w_y^{\max}$ and $w_y^{\min}$ are the two values the weights of the training samples with label y can take and $\mathbb{1}_{A_y}(\cdot)$ is the indicator function indicating if the sample corresponds to a "small enough" value of f for class y. On can imagine various possibilities to define each one of the two sets A_y so that it focuses on the "small enough" f-values of the corresponding class y.

In the paper, only the results of the second method are shown. In these computations, the set A_0 (of elements of class 0 with "small enough" f) was defined so as to contain the 10% training elements of class 0 ($x \in \mathcal{D}$) showing the smallest values of f; this comes together with a weight $w_0^{\max} = 1$ and $w_0^{\min} = 0.2$. On the other hand, for class 1, we used $w_1^{\max} = w_1^{\min} = 1$ so that the choice of the set A_1 has no impact in this case.

References

[1] Laurine Duchesne, Quentin Louveaux, and Louis Wehenkel. Supervised learning of convex piecewise linear approximations of optimization problems. 2021. Submitted for publication.