ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

Fast and Stable Interval Bounds Propagation for Training Verifiably Robust Models

Paweł Morawiecki
Institute of Computer Science, Polish Academy of Sciences,
pawel.morawiecki@gmail.com
Przemysław Spurek
Marek Śmieja
Jacek Tabor
Institute of Computer Science and Computational Mathematics,
Jagiellonian University, Poland

Abstract. We present an efficient technique to train classification networks which are verifiably robust against norm-bounded adversarial attacks. This framework is built upon interval bounds propagation (IBP), which applies the interval arithmetic to bound the activations at each layer and keeps the prediction invariant to the input perturbation. To speed up and stabilize training of IBP, we supply its cost function with an additional term, which encourages the model to keep the interval bounds at hidden layers small. Experimental results demonstrate that the training of our model is faster, more stable and less sensitive to the exact specification of the training process than original IBP.[1]

1 Introduction

Although deep learning models achieve impressive performance on various tasks, they are also vulnerable to adversarial attacks [1]. Adversarial attacks rely on creating such input data points, which are visually indistinguishable from 'normal' examples, but drastically change the prediction of the model. In recent years, a lot of effort has been put on understanding deep learning models and making them more robust [2, 3].

One remedy is to construct adversarial examples and add them to the training set. While such models become robust to many adversarial attacks, there are no guarantees that another adversarial scheme exists. To formally verify the robustness of the model against norm-bounded perturbations, one can find the outer bound on the so-called 'adversarial polytope'. As an alternative, [4] adapted the framework of 'abstract transformers' to compute an approximation to the adversarial polytope using the stochastic gradient descent training. While these methods guarantee that no adversary within a given norm can change the class label, these techniques are computationally demanding and do not scale well to large networks.

In this paper, we consider the framework of interval bounds propagation (IBP) proposed by Gowal et al. [5] for constructing provably robust classifiers. IBP uses the interval arithmetic to propagate axis-aligned bounding box from layer to layer and minimizes the upper bound on the maximum difference between any

[1]The source code is available at: https://github.com/pawelmorawiecki/Fast-and-stable-IBP

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

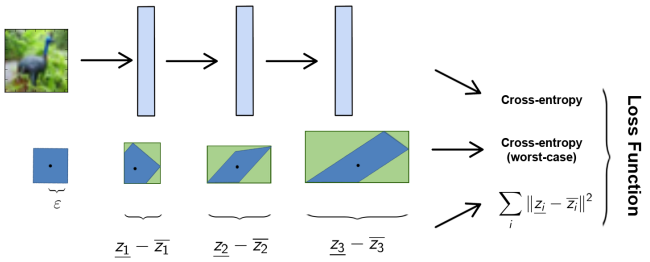

Fig. 1: The scheme of the proposed method. The original IBP loss is supplied with an additional term controlling the errors across layers.

pair of logits when the input is perturbed within the norm-bounded ball. While IBP is computationally appealing, it requires careful tuning of hyper-parameters to provide tight bounds on the verification network. In this contribution, we show that the training procedure of IBP can be significantly simplified, which results in more stable training and faster convergence. Our key idea relies on combining the IBP loss with an additional term, which controls the size of adversarial polytope across layers, see Figure 1 for the illustration. As a result, our model is less sensitive to the change of the aforementioned IBP hyper-parameters, which makes it easier to use in practice.

2 Interval bounds propagation

Training robust classifiers. We consider a feed-forward neural network $f : \mathbb{R}^D \to \mathbb{R}^N$ designed for a classification task. The network is composed of K layers given by K transformations:

$$z_k = h_k(z_{k-1}), \text{ for } k = 1, \ldots, K.$$

In practice, h_k is either an affine transformation or nonlinear monotonic function such as ReLU or Sigmoid. In the training stage, we feed the network with pairs of input vector $z_0 = x$ and its correct class label y_{true} and minimize the cross-entropy with softmax applied to the output logits z_K.

In the adversarial attack, any test vector x can be perturbed by some Δ with l_∞ norm-bounded by ϵ, for a small fixed $\epsilon > 0$. Thus the input to the network can by any point in D-dimensional hyper-cube:

$$I_\epsilon(x) = I(x - \epsilon, x + \epsilon) = [x_1 - \epsilon, x_1 + \epsilon] \times \ldots \times [x_D - \epsilon, x_D + \epsilon].$$

centered at x with side length 2ϵ. This set is transformed by a neural network f into some convex set called adversarial polytope:

$$\mathcal{Z}_\epsilon(x) = \{f(z) : z \in I_\epsilon(x)\}.$$

To design provable defense against adversarial attack, we have to ensure that class label y_{true} does not change for any output $z_K \in \mathcal{Z}_\epsilon(x)$. In other words, all inputs from the hyper-cube $I_\epsilon(x)$ should be labeled as y_{true} by a neural network f. In this context, a fraction of incorrectly classified examples on the test set is called the *verified test error*.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

Verifiable robustness using IBP. Exact verification of the model robustness may be difficult even for simple neural networks. Thus we usually look for an easier task computing loose outer bound of $\mathcal{Z}_\epsilon(x)$ and control the class label inside this bound. In the IBP approach [5], we find the smallest bounding box at each layer that encloses the transformed bounding box from the previous layer. In other words, we bound the activation z_k of each layer by an axis-aligned bounding box

$$I(\underline{z}_k, \overline{z}_k) = [\underline{z}_{k,1}, \overline{z}_{k,1}] \times \ldots \times [\underline{z}_{k,D_k}, \overline{z}_{k,D_k}].$$

In the case of neural networks, finding such a bounding box from layer to layer fashion can be computed efficiently using the interval arithmetic. By applying the affine layer $h_k(z_{k-1}) = W_k z_{k-1} + b_k$ to $I(\underline{z}_{k-1}, \overline{z}_{k-1})$, the smallest bounding box $I(\underline{z}_k, \overline{z}_k)$ for output z_k is given by

$$\begin{aligned} \mu_{k-1} &= \tfrac{\overline{z}_{k-1}+\underline{z}_{k-1}}{2}, & r_{k-1} &= \tfrac{\overline{z}_{k-1}-\underline{z}_{k-1}}{2}, \\ \mu_k &= W_k\mu_{k-1} + b_{k-1}, & r_k &= |W_k| r_{k-1}, \\ \underline{z}_k &= \mu_k - r_k, & \overline{z}_k &= \mu_k + r_k, \end{aligned}$$

where $|\cdot|$ is an element-wise absolute value operator. For a monotonic activity function h_k, we get the interval bound defined by:

$$\underline{z}_k = h(\underline{z}_{k-1}) \ , \ \overline{z}_k = h(\overline{z}_{k-1}).$$

To obtain a provable robustness in the classification context, we consider the worst-case prediction for the whole interval bound $I(\underline{z}_K, \overline{z}_K)$ of the final logits. More precisely, we need to ensure that the whole bounding box is classified correctly, i.e. no perturbation changes the correct class label. In consequence, the logit of the true class is equal to its lower bound and the other logits are equal to their upper bounds:

$$\hat{z}_{K,y}(\epsilon) = \begin{cases} \overline{z}_{K,y}, & \text{for } y \neq y_{true}, \\ \underline{z}_{K,y_{true}}, & \text{otherwise .} \end{cases}$$

Finally, one can apply softmax with the cross-entropy loss to the logit vectors $\hat{z}_K(\epsilon)$ representing the worst-case prediction.

As shown in [5], computing interval bounds uses only two forward passes through the neural network, which makes this approach appealing from a practical perspective. Nevertheless, a direct application of the above procedure with a fixed ϵ may fail because propagated bounds are too loose especially for very deep networks. To overcome this problem Gowal et al. supplied the above interval loss with a typical cross-entropy cost applied to original non-interval data:

$$\text{IBP} = \kappa\ell(z_K, y_{true}) + (1-\kappa)\ell(\hat{z}_k(\epsilon), y_{true}),$$

where κ is a trade-off parameter. In the initial training phase, the model uses only classical loss function applied to non-interval data ($\kappa = 1$). Next, the weight

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

of the interval loss is gradually increased up to $\kappa = 1/2$. Moreover, the training starts with the small perturbation radius ϵ, which is also increased in later epochs. The training process is sensitive to these hyperparameters and finding the correct schedule for every new data set can be problematic and requires extensive experimental studies. This makes the whole training procedure time consuming, which reduces practicality of this approach.

Constrained interval bound propagation. To make IBP less sensitive to the training settings and provide more training stability (particularly for bigger ϵ), we propose to enhance the cost function. We want to directly control the bounding boxes at each layer of the network. More precisely, in addition to the IBP loss, we minimize the size of the outer interval bound at each layer. Thus our cost function equals

$$\text{constrainedIBP} = \kappa\ell(z_K, y_{true}) + (1-\kappa)\ell(\hat{z}_k(\epsilon), y_{true}) + \sum_{k=1}^{K} \|\overline{z}_k - \underline{z}_k\|^2.$$

We argue that such the addition would help to circumvent limitations of the original IBP. First, gradients would be calculated not only with respect to the the last layer but to all hidden layers. This should bring more training stability, especially at the early training stage. Second, we expect it would be easier for a model to have small interval bounds in the final layer when bounds are constrained in hidden layers. And indeed our experimental results support these research hypotheses.

3 Experiments

Here we report our experiments, which show the effect of the proposed loss function and give some insight why it is beneficial to minimize the interval bounds in hidden layers.

We conduct the experiments on CIFAR-10, SVHN and MNIST datasets. The neural network architectures used in the experiments are the same as in [5] and these are 3 convolutional nets called *small*, *medium* and *large*.

Faster convergence. First, we highlight that our approach minimizes the verified test error much faster than IBP. Since the performance of both methods on MNIST is comparable, we only report the results on most challenging cases of CIFAR-10 and SVHN with maximal perturbation radius $\epsilon = 8/255$.

Figure 2 shows clearly that the difference between both methods is substantial. In the case of CIFAR-10 after 100 epochs, the verified test error is over 20 percentage points lower, whereas the nominal error is close. The shape of the curves for SVHN is similar, but the gain in verified accuracy is slightly lower; after 50 epochs the verified error of constrainedIBP is also 20 percentage points lower than the one obtained by IBP, while after 100 epochs the difference is around 10 percentage points.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

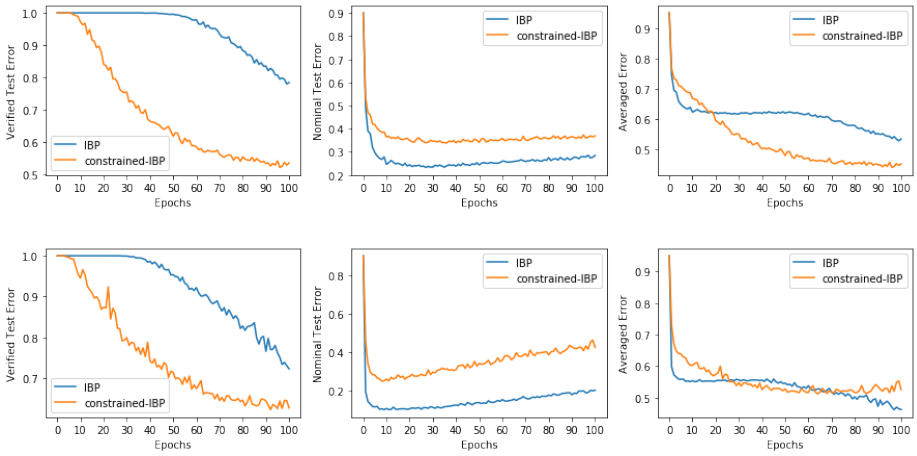

Fig. 2: Verifiably robust training for the CIFAR-10 (top row) and SVHN (botton row) with adversarial perturbations bounded by $\epsilon = 8/255$.

More stable training. Gowal et al. stated that their method needs to slowly increase ϵ (from 0 to ϵ_{train}) to provide stability and convergence during the training. For example, for CIFAR-10, this 'ramp-up' phase lasts 150 epochs. It raises a natural question whether we could speed-up the ϵ increase and whether our new term in the loss function is helpful in this regard.

We investigate the more dynamic ϵ changes to reduce the training time. For the MNIST dataset, increasing ϵ 2.5 faster results in lack of convergence for the original IBP method, see Figure 3. On the other hand, the additional term in the loss function helps to stabilize the training and obtain the minimization of verified error.

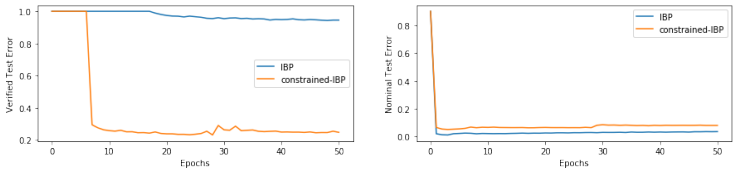

Fig. 3: Verifiably robust training for the MNIST dataset with adversarial perturbations bounded by $\epsilon = 0.4$. Experiments done on the small architecture. Perturbation radius ϵ was increased 2.5 times faster than in [5].

We also show that even if we keep the original ϵ changes, IBP may stuck in a local minimum for a very long time. The experiment was done on CIFAR-10 with the large architecture and $\epsilon = 4/255$. The test error goes down very quickly, reaching 0.2, whereas the verified test error remains 100% for over 100 epochs. On the contrary, our approach steadily minimizes the verified test error.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

4 Conclusion

We proposed to minimize the size of an outer bound of the adversarial polytope across hidden layers. This modification was motivated by the observation that IBP implicitly minimizes these bounds in the case of the successful, convergent training. By adding this constraint explicitly, the model become less sensitive to the change of hyper-parameters and, in consequence, we could increase the perturbation radius more dynamically to the desired value, which makes the training faster. The proposed idea is not limited to the IBP and can be incorporated in other robust training methods, such as the convex-optimization-based approaches.

Acknowledgement

This work was carried out when P. Spurek and M. Śmieja were on the internship at the Institute of Computer Science, Polish Academy of Sciences. The work of M. Śmieja was partially supported by the National Science Centre (Poland), grant no. 2018/31/B/ST6/00993. The work of P. Spurek was partially supported by the National Science Centre (Poland), grant no. 2019/33/B/ST6/00894. The work of J. Tabor was partially supported by the National Science Centre (Poland), grant no. 2017/25/B/ST6/01271.

References

[1] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[2] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[3] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*, 2019.

[4] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 3(POPL):41, 2019.

[5] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.