Boosting Geometric Certification of Neural Networks

Shubhangi Ghosh
Advisers: Dimitar Dimitrov, Maximilian Baader
Supervising Professor: Prof. Martin Vechev

ETH Zürich

ABSTRACT

Prior approaches to certify neural networks against geometric perturbations pass through the neural network, an abstract polyhedron domain of perturbed images for each image in a given dataset. The input polyhedra are represented by linear upper and lower constraints for each individual pixel in an image in terms of the perturbation parameters, generated by a combination of sampling and optimisation. Our key idea is to also generate linear constraints bounding affine combinations of multiple pixels in the image. The multiple-pixel constraints allow our method to capture *more precisely* the complex relationships between the original image, the geometric transformation parameters and the resulting transformed images. Our experimental results show the method is signicantly more precise than prior work. We observe that scalability is maintained with respect to time for datasets with smaller images (in terms of pixels); e.g. MNIST; and smaller networks, while our method is slower for datasets with coplex images; e.g. CIFAR; and larger networks.

1 INTRODUCTION

Neural networks deployed in safety-critical applications are required to be robust to geometric perturbations. The current state of the art, DEEPG[1], identifies linear constraints bounding individual pixels above and below in terms of the geometric parameters. These constraints are obtained by sampling attacks and optimisation to find tight, but sound, constraints. DEEPG does not encode the dependencies between pixels, which causes the method to lose precision. Through our method, we seek to identify interactions between pixels with low variability with respect to our perturbation parameters. An interaction between pixels is formulated as an affine combination of pixels, such that the linear coefficients of pixels sum up to one. We choose these interactions through two approaches:

- Difference of neighbouring pixels
- Interpolative Decomposition of the perturbation matrix (Terms explained later)

Intuitively, we expect the a pixel to have similar intensity as its neighbouring pixel. Hence, constraints generated from neighbourhoods help us gain precision. For our next approach, we consider the matrix, with its rows being the pixel values for a particular randomly sampled value of the perturbation parameters. Through a matrix decomposition technique, namely Interpolative Decomposition [5], we identify a smaller subset of pixels such that the remaining pixels can be expressed in terms of the fundamental pixels. For the remaining pixels, we consider the interaction as the difference between the pixel value and its linear approximation in terms of the fundamental pixels. We expect the difference to be close to zero. The number of pixels we choose to be fundamental is a parameter. This enables us to parameterise a trade-off between precision and runtime. Linear constraints for the affine interactions between pixels are also found in a similar fashion as DEEPG, by sampling and optimisation.

Main Contributions Our main contributions are:

- We devised two methods to define interactions between pixels, namely Neighbourhood constraints and Interpolative decomposition, which help us gain maximum precision.
- Implemented a complete framework to integrate bounds for pixel interactions with the DEEPG model. Since backsubsitution of the first layer neurons in terms of the pixels is not possible as earlier, we calculate the bounds through a linear program.
- Devised a method to parameterise a trade-off between precision of certification and runtime.

2 BACKGROUND

Interpolative Decomposition
An interpolative decomposition (ID) of a matrix $A \in C^{m\times n}$ of rank $k \leq min\{m,n\}$ is a factorization

$$A\Pi = [A\Pi_1 A\Pi_2] = A\Pi_1[IT]$$

where $\Pi = [\Pi_1, \Pi_2]$ is a permutation matrix with $\Pi_1 \in \{0,1\}^{n\times k}$, and $A\Pi_2 = A\Pi_1 T$. This can be equivalently written as $A = BP$, where $B = A\Pi_1$ and $P = [I,T]\Pi^T$, are the skeleton and interpolation matrices respectively.

If A does not have exact rank k, then there exists an approximation in the form of an ID such that $A = BP + E$, where $\|E\| \sim \sigma_{k+1}$ is on the order (k+1)-th largest singular value of A. σ_{k+1} is the best possible error for a rank-k approximation and is achieved by SVD. But for our application, we need the structure of A to be preserved, in the sense that we can identify a subset of fundamental columns of the matrix to be decomposed and express all other columns as linear combinations of the fundamental columns. This achieved by ID, and also it is cheaper to construct than SVD.

For our application, the columns of the matrix A represent the values of a particular pixel for all randomly sampled values of the perturbations. Thus, by using ID we identify a subset of fundamental pixels from an image and express all other pixels as linear combinations of these fundamental pixels.

3 METHODOLOGY

Motivation As mentioned before, DEEPG[1] does not capture dependencies of multiple pixels in terms of the geometric perturbation parameters. Our key insight is to include affine constraints over multiple pixels to describe our input polyhedral abstraction of transformed images. We first propose methods to obtain the affine expressions of multiple pixels that we want to consider, namely (i) difference of neighbourhood pixels, and (ii) constraints generated by

Interpolative decomposition over the matrix of transformed images. Owing to the multiple pixel constraints, the first layer neurons can no longer be substituted in terms of the input pixels, and thereby the geometric parameters. To integrate the now more precise input polyhedron in the certification pipeline, we now have to solve a linear program, in terms of the first layer neurons, the input pixels and the geometric parameters, to obtain tighter bounds for each neuron whose sign cannot be determined by DEEPG alone.

The DEEPG certification is performed by splitting the input domain of parameters and performing the certification for each split independently. This allows a gain of precision. We use the constraints over multiple pixels only to certify input perturbation splits which cannot be certified by DEEPG, owing to the longer certification times of our approach. Below, we describe how our approach extends the DEEPG methodology.

Step 1 We use DEEPG[1] to create single pixel constraints. That is we randomly pick perturbations in the geometric perturbation range to be certified and store the resulting images. For each pixel in the image, locally optimal linear constraints are formed in terms of the geometric parameters. This is done by solving a linear program to find weights and biases describing a linear upper and lower bound having the least volume while encompassing the perturbed images. The globally optimal constraints, which are sound over all perturbations in the given range, are found by numerically splitting the perturbation range iteratively. The functional form of the pixels is assumed to be Lipschitz continuous over the geometric parameters and mean value theorem is used to bound the maximum violation in each split.

Step 2 We try to certify all the splits of the input perturbation range using DEEPG. The specifications; the linear single pixel constraints; of the splits which are not certified by DEEPG are stored to be further attempted to be certified by the more precision input polyhedron obtained from our approach.

Step 3 We generate the affine combinations of multiple pixels that we want to consider through two strategies described below.

- **Neighbourhood constraints**
  We consider the difference of each pixel with its right and bottom neighbours.

- **Interpolative Decomposition [5]**
  We have already stored the random perturbation in the input perturbation range for the image under consideration. Using these, we construct a perturbation matrix, where each perturbed image is represented as a single column in the matrix. Let A be the perturbation matrix, with m rows for m random perturbations, and n columns equal to the number of pixels in an image. Consider $\hat{A} = A\Pi$, where Π is a permutation matrix denoting a permutation of the columns of A. Upon applying interpolative decomposition, we get:

  $$\hat{A} = \hat{A}_{:,J}X,$$

  where

  - J is a subset of r indices from $\{1,2,...,n\}$.
  - The $m \times r$ matrix $A_{:,J}$ represents the columns in A corresponding to the indices in J.
  - X is an $r \times n$, containing an $r \times r$ identity sub-matrix. The remaining $n-r$ columns of X represent the linear approximation of the $n-r$ pixels in terms of the r pixels.

  To construct our affine multiple-pixel expressions, we use the $n-r$ columns in X representing the $n-r$ pixels in terms of the r fundamental pixels. For each of the $n-r$ columns, we use the difference of the pixel referred to and its linear expansion in terms of the r pixels. For example, for the j^{th} column outside of the identity submatrix of X, our set of coefficients would be $\{-1, X_{:,j}\}$ corresponding to the pixels $\{x_j, x_{1-r}\}$. The corresponding indices in the image representation are also given as out by the algorithm used (Scipy's Interpolative Decomposition algorithm).

Step 4 We again generate constraints for the now available multiple-pixel affine expressions by solving a linear program similar to DEEPG. The globally optimal constraints are again found numerically. The global optimisation step for the multiple pixel constraints is slower because their variability in term of the geometric parameters is designed to be smaller as compared to individual pixels. Thus, we need more steps to obtain a finer split, which would give the maximum violation that we are looking for.

Step 5 We run our approach of propagating the more precise input polyhedron through the neural network only for the chunks that were not certified by DEEPG. Passing the input polyhedron with multiple pixel constraints is not straight-forward and cannot be done in a DeepPoly-like fashion. Now, for each neuron in the network, we first obtain the upper and lower bounding linear constraints in terms of the firat layer neurons. The constraints are translated in terms of the geometric parameters by solving a Linear Program, over the constrained space described by the input polyhedron.

4 EVALUATION

For our certification method, we consider as explained before, two methods of defining interactions between pixels: (1) Differences of neighbouring pixels and (2) Interpolative decomposition [2] of the attack matrix. We carry out the certification over splits of the input perturbation domain. We only attempt to certify those splits but our multiple-pixel approach, which cannot be certified by DEEPG [1]. We generate linear upper and lower bounding constraints for individual pixels and if needed, affine interaction between pixels and propagate the input polyhedron through the network to certify. Although, our approach is more precise than DEEPG, it is slower because the bounds of first-layer neurons cannot be backsubstituted in a Deep-poly-like fashion, and a linear program needs to be solved to bound the neurons in terms of the geometric parameters.

First, we demonstrate that our method gains precision over DEEPG and thus enables us to achieve higher certification rates. Second, we compare the average constraint generation time and the average chunk certification time of our approach with DEEPG. Third, we compare the time taken by our multiple-pixel approach and DEEPG (by increasing the number of splits) to achieve the same certification rate. Fourth, we analyse the precision-runtime tradeoff for the Interpolative Decomposition approach. Finally, we also share some observations about the performance of our method.

4.1 Experimental Setup

We evaluate on image recognition datasets: MNIST [4], Fashion-MNIST [6] and CIFAR-10 [3]. For each dataset, we select 100 images from the test set to certify. The same 100 images are used for DEEPG and our extended certification approaches. Among these 100 images, we discard all images that are misclassified even without any transformation. In all experiments for MNIST and Fashion-MNIST we evaluate a 3-layer convolutional neural network with 9 618 neurons, while for the more challenging CIFAR-10 dataset we consider a 4-layer convolutional network with 45 216 neurons. Details of these architectures are provided in Table 1. We certify robustness to composition of transformations such as rotation, translation, scaling, shearing and changes in brightness and contrast. All experiments were run with multithreading on 4 CPUs.

Table 1: Architectures used in experimental evaluation.

MNIST	FashionMNIST	CIFAR-10
CONV 32 4×4 + 2	CONV 32 4×4 + 1	CONV 32 4×4 + 1
CONV 64 4×4 + 2	CONV 32 4×4 + 2	CONV 32 4×4 + 2
FC 200	CONV 64 4×4 + 2	CONV 64 4×4 + 2
FC 10	FC 150	FC 150
	FC 10	FC 10

4.2 Precision gain over DEEPG

Table 2 reports certification rates of our approach compared with DEEPG over the same number of splits of the transformation domain. We observe significant gains in certification rate over DEEPG. The certification rate for Rotation(30) on the MNIST dataset is higher by 7.8%, while on the Sh(2), R(2), Sc(2), B(2, 0.001) experiment is higher by 12%. This evidence that our method gains precision in the input polyhedron by including linear constraints bounding affine multiple pixel combinations. This also justifies that the affine combinations of multiple pixels that we have chosen have lower variability in terms of the transformation parameter and significantly helps us shrink the input polyhedron. The performance improvement from Fashion-MNIST is comparatively lower.

4.3 Constraint generation and chunk verification time

The bottleneck in terms of time for **constraint generation** is the global optimization step. After the attacks are sampled, and linear constraints are generated locally optimal to the set of attacks, the bounds are optimised through a numerical approach. The pixels and the affine expressions of multiple pixels that we consider, are assumed to be Lipschitz continuous in terms of the transformation parameters, i.e. have bounded gradients. Mean value theorem is used to find the deviation of the locally optimal constraints in terms of the globally optimal constraints. The affine multiple pixel expressions exhibit lower variability in terms of the geometric parameters, and hence have lower Lipschitz constants. This increases the time for global optimisation because we need more splits for the Lipschitz cone to intersect the locally optimal linear bounds.

The time-bottleneck for **chunk verification** for multiple-pixel approaches is the linear program we use to reduce the linear bounds of neurons, from in terms of the first layer neurons, to in terms of the pixels and the transformation parameters. The first-layer neurons are affine combination of the image pixels. However, we cannot obtain linear bounds for the network neurons through affine combinations of existing pixel bounds in a DeepPoly-like fashion because we want to incorporate linear bounds over affine combinations of pixels. Thus, we solve multiple large linear programs, one each for every neuron, with the number of variables being equal to the sum of the total number of image pixels and the transformation parameters. The computational complexity of solving a linear program is exponential in the number of variables. However, since each neuron depends on only a small number of pixels, owing to the sparsity of convolutional neural networks, we choose to exploit this fact in our next steps. The linear program to solve the bounding constraints for each neuron can be solved in terms of only the image pixels it depends on, and hence the complexity can be reduced exponentially. This would be our future steps.

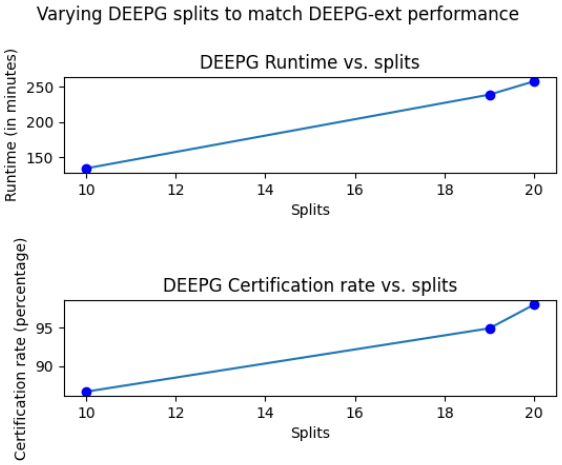
4.4 Comparison of Neighbours and Decomposition approaches

The average constraint generation time is greater for decomposition expressions than neighbourhood expressions because the decomposition expressions, by definition, have near zero variability. Hence, due to the low Lipschitz constant, the global optimization step is time-consuming. The average chunk verification time is also higher for decomposition constraints. This could be owing to the fact that the Linear Program we need to solve to obtain the bounds for the neurons is more complex.

The approach using decomposition expressions is however more precise and gives higher certification rate as compared to neighbourhood expressions for the same number of domain splits.

4.5 Experiment time for fixed certification rate

Figure 1: Varying splits of DEEPG to match the performance of DEEPG-ext



We observe that for datasets with images which have smaller number of pixels, and smaller neural networks that we have used for certification of MNIST images, our multiple pixel approach outperforms DEEPG both in terms of certification rate and time. For a fixed certification rate for the MNIST R(30) experiment, DEEPG with increased splits takes longer time (239 min) to certify than our multiple pixel approach with Decomposition expressions (171 min). However, this property no longer holds for the datasets with colour images certified by larger neural networks. The linear increase in the number of pixels results in the computational complexity of the linear program increasing exponentially. Thus, the increase in the number of pixels is the primary bottleneck, and this can be solved by breaking down the larger linear programs for obtaining the constraints for the network neurons into smaller linear programs over the relevant pixels.

We also observe that sometimes increasing the splits in the input domain leads to a lower certification rate.

4.6 Trade-off between precision and runtime with Interpolative decomposition

In Figure 2, we observe a trade-off between the rank of the approximation considered for Interpolative decomposition (k) and the runtime. Higher values of k increase the runtime but achieve higher certification rates.

4.7 Observations

- **Low certification rate for translation**
  The certification rate for images under the translation transformation is surprisingly low. However, the certification rate for splits is still competitive. The high certification rate for splits could have been retained due to the fact that we have split the domain too finely. The low image certification rate for translation needs to be investigated. It is possible that the bounding constraints are not precise enough for translation in terms of the parameters. Here, the parameters are the pixels.

Table 2: Comparison of DEEPG-ext, Neighbours and Decompostion, which uses linear constraints for affine combination of multiple pixels with DEEPG , which uses linear constraints for singular pixels. Here, Splits refers to the number of splits of the input domain and k refers to the rank of Interpolative decomposition. R(ϕ) corresponds to rotations with angles between $\pm\phi$; T(x, y), to translations between $\pm x$ pixels horizontally and between $\pm y$ pixels vertically; Sc(p), to scaling the image between $\pm p\%$; Sh(m), to shearing with a shearing factor between $\pm m\%$; and B(α, β), to changes in contrast between $\pm\alpha\%$ and brightness between $\pm\beta$.

		Accuracy (%)	Attacked (%)	Splits	k	Certified (%)		
						DEEPG	Neighbours	Deceomposition
MNIST	R(30)	99.1	0.0	10	40	86.7	**92.86**	**96.94**
	T(2, 2)	99.1	1.0	11	20	77.0	**78.0**	**79.0**
	Sc(5), R(5), B(5, 0.01)	99.3	0.0	2	10	32.0	**64.0**	**70**
	Sh(2), R(2), Sc(2), B(2, 0.001)	99.2	0.0	1	5	72.0	**81.0**	**84.0**
Fashion-MNIST	Sc(20)	91.4	11.2	10	10	70.8	**74.16**	**75.28**
	R(10), B(2, 0.01)	87.7	3.6	4	10	71.4	**73.81**	**76.19**
	Sc(3), R(3), Sh(2)	87.2	3.5	2	10	57	**59.3**	**62.79**

Table 3: Comparison of DEEPG-ext, Neighbours and Decompostion, with the baseline DEEPG in terms of constraint generation time and chunk verification time (in seconds)

Average constraint generation time			
Experiment	DEEPG	Neighbours	Deceomposition
MNIST R(30)	49.1	50.1	31.48
MNIST T(2,2)	493	198	883
MNIST Sh(2), R(2), Sc(2), B(2, 0.001)	85.0	429	108.7
Fashion-MNIST Sc(20)	39.61	44.3	26.1
Fashion-MNIST R(10), B(2, 0.01)	216.2	2386.9	9671

Avergae chunk certification time			
Experiment	DEEPG	Neighbours	Deceomposition
MNIST R(30)	5.3	23.4	20.4
MNIST T(2,2)	2.0	8.61	16.8
MNIST Sh(2), R(2), Sc(2), B(2, 0.001)	3.36	39	22.0
Fashion-MNIST Sc(20)	7.8	38.8	71.9
Fashion-MNIST R(10), B(2, 0.01)	3.85	419.87	75.18

We lose the smoothing effect of bilinear interpolation and the variation of the pixels can be too abrupt in terms of the pixels. The precision gained by multiple pixel constraints, still does not improve the certification rate, however.

- **Fall in certification rate for increasing splits for DEEPG** For some experiments, Translation(2,2), we observed a fall in certification rate for increase in the number of splits. This is unexpected and needs to be investigated.
- **Low improvement in certification rate for the Fashion-MNIST dataset** We observe a lower improvement of our approach in certification rate over DEEPG for the Fashion-MNIST dataset.

REFERENCES

[1] M. Balunovic, M. Baader, G. Singh, T. Gehr, and M. Vechev. Certifying geometric robustness of neural networks. In *Advances in Neural Information Processing Systems*, pages 15313–15323, 2019.

[2] H. Cheng, Z. Gimbutas, P.-G. Martinsson, and V. Rokhlin. On the compression of low rank matrices. *SIAM Journal on Scientific Computing*, 26(4):1389–1404, 2005.

[3] A. Krizhevsky. Learning multiple layers of features from tiny images, 2009.

[4] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2010.

[5] P.-G. Martinsson, V. Rokhlin, Y. Shkolnisky, and M. Tygert. Id: A software package for low-rank approximation of matrices via interpolative decompositions, version 0.2, 2008.

[6] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Figure 2: Varying parameter k of Interpolative decomposition