Why drop-max is effective in making convolutional neural networks (CNNs) more robust

Min Xian[1], Olga Kosheleva[2], Martine Ceberio[3], and Vladik Kreinovich[3]

[1]Department of Computer Science, University of Idaho, Idaho Fall, Idaho 83406, USA, mxian@uidaho.edu
Departments of [2]Teacher Education and [3]Computer Science, University of Texas at El Paso, 500 W. University, El Paso, TX 79968, USA, olgak@utep.edu, vladik@utep.edu

1. Convolutional neural networks (CNNs): a brief reminder

- In processing images, it turned out that convolutional neural networks – CNNs, for short – are very effective.
- A CNN is a multi-layer neural network, in which each neuron from each layer is associated with a certain point in the original image.
- The layers work as follows.
- Each neuron from the first layer uses, as inputs $x_1, \ldots, x_n$, intensities of the original image in this location and in several nearby locations.
- Each neuron from every other layer uses, as inputs, outputs $x_1, \ldots, x_n$ of the neurons from the previous layer corresponding to this location and to several nearby locations.
- Each layer is either a linear (convolutional) layer or a max-layer.

2. Convolutional neural networks (CNNs): a brief reminder (cont-d)

- In a linear layer, the output signal y is a linear combination of inputs

$$y = w_0 + w_1 \cdot x_1 + \ldots + w_n \cdot x_n.$$

- The coefficients w_i are determined during the training.
- In a max-layer, the output signal y is the maximum of all the inputs:

$$y = \max(x_1, \ldots, x_n).$$

- Layers of these two types intertwine:
  - for each linear layer, the next layer is a max-layer, and
  - for each max-layer, the next layer is a linear layer.

3. Problem: CNNs are not robust

- One of the main problems of the CNNs is that they are not very robust:
  - changes in a small number of pixels, changes that do not change our visual perception of the image,
  - can lead to a drastic change in the output of the neural network and
  - thus, can lead to a wrong classification of the image.

4. How to make CNNs more robust: an idea

- To find out how to make CNNs more robust, let us analyze how changes in a small number of pixels propagate via the CNN.
- When this change affect an input of a neuron from a linear layer, the inputs values are kind of averaged.
- So the initial effect decreases.
- For example, if all the weights $w_1, \ldots, w_n$ are equal and add up to 1, the effect decreases to $1/n$ of the original size.
- If these were the only layers, the effect of the small changed would eventually decrease and become negligible.
- However, in the max-layer:
  - if one of the changed inputs is larger than all other inputs,
  - then the output of the neuron also changes drastically.
- So, to make CNN more robust, a natural idea is to focus our efforts on max-neurons.

5. Empirical fact

- The above idea has indeed been successfully implemented.
- Namely, it turns out that replace the maximum of all the inputs with the second largest value makes CNN more robust.
- This was shown on the example of the breast cancer classification.
- Neurons that return the second largest of all the inputs are called *drop-max* neurons.
- The reason for these name is that they compute the max of what will happen if we drop the largest of the original n values.
- A natural question is: why,
  - out of many possible ways to make maximum more robust,
  - drop-max leads to good results?
- In this talk, we provide a theoretical explanation of the drop-max's success.

6. Analysis of the problem

- A max-neuron computes the maximum of its n inputs.
- If one of the input values is changed, we cannot always return the maximum of the original inputs.
- Indeed, it is possible that exactly the largest input was changed.
- So based on the remaining values, we cannot determine what was the value of the original largest input.
- Since we cannot always produce the maximum, at least we can try to be as close to the maximum as possible.
- For example, we can try to generate some value which is
  - between the second largest and the largest,
  - or at least between the third largest and the largest.
- Let us formulate this idea in precise terms.

7. Notations and definitions

- We will follow standard notations from statistics.
- Let us denote, for each tuple $x_1, \ldots, x_n$, by $x_{(i)}$, the i-th element in the ordering of these elements from the smallest to the largest:

$$x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(n)}.$$

- In these terms, the largest element from this tuple is $x_{(n)}$, and the second largest is $x_{(n-1)}$.
- We say that a function $f(x_1, \ldots, x_n)$ is *robust* if:
  - for every tuple $(y_1, \ldots, y_n)$,
  - whenever we form a new tuple $(x_1, \ldots, x_n)$ by replacing one of its elements y_i with a different number, then we will have

$$y_{(n-2)} \leq f(x_1, \ldots, x_n) \leq y_{(n)}.$$

8. Main result and comment

- *The only robust function $f(x_1, \ldots, x_n)$ is the function $f(x_1, \ldots, x_n) = x_{(n-1)}$ that returns the second largest element of the tuple.*
- *Comment.* What if we want to get even closer to the maximum and require that
$$y_{(n-1)} \le f(x_1, \ldots, x_n) \le y_{(n)}?$$
- This condition implies the inequality $y_{(n-2)} \le f(x_1, \ldots, x_n) \le y_{(n)}$.
- So, we conclude, from our main result, that under this condition, we will still have $f(x_1, \ldots, x_n) = x_{(n-1)}$.
- However, the following simple example shows that for this function $f(x_1, \ldots, x_n)$, we do not always have the desired inequality.
- Let us take $y_1 = 1$, $y_2 = 2$, and $y_3 = 3$.
- In this case, $y_{(1)} = y_1 = 1 < y_{(2)} = y_2 = 2 < y_{(3)} = y_3 = 3$.
- Let now change the value 3 to 0, i.e., take $(x_1, x_2, x_3) = (1, 2, 0)$.

9. Comment (cont-d)

- For this new tuple, we have $x_{(1)} = 0 < x_{(2)} = 1 < x_{(3)} = 2$.
- In this case, $n = 3$, so $f(1, 2, 0) = x_{(2)} = 1$ which is smaller than $y_{(n-1)} = y_{(2)} = 2$.
- Thus, it is not possible to require the value to be that close to the maximum.
- In this sense, the inequality $y_{(n-2)} \leq f(x_1, \ldots, x_n) \leq y_{(n)}$ is the best we can achieve.

10. Proof: Part 1

- Let us first prove that the function $f(x_1, \ldots, x_n) = x_{(n-1)}$ is robust in the sense of our Definition.
- Indeed, there are only four possibilities.
- We will show that the desired inequality holds for all four of them.

11. Proof: Part 1.1

- If all three original top values $y_{(n-2)} \le y_{(n-1)} \le y_{(n)}$ remain unchanged, then we have the following four sub-options for the replaced value r.
- If $r \le y_{(n-2)}$, then in the new order, we have

$$\ldots \le r \le \ldots \le y_{(n-2)} \le y_{(n-1)} \le y_{(n)}.$$

- Thus, we have $x_{(n-1)} = y_{(n-1)}$ and so, $y_{(n-2)} \le x_{(n-1)} = y_{(n-1)} \le y_{(n)}$.
- Hence, the desired inequality is satisfied.
- If $y_{(n-2)} < r \le y_{(n-1)}$, then in the new order, we have

$$\ldots \le y_{(n-2)} \le r \le y_{(n-1)} \le y_{(n)}.$$

- Thus, we have $x_{(n-1)} = y_{(n-1)}$) and so, $y_{(n-2)} \le x_{(n-1)} = y_{(n-1)} \le y_{(n)}$.
- Hence, the desired inequality is satisfied.

12. Proof: Part 1.1 (cont-d)

- If $y_{(n-1)} < r \leq y_{(n)}$, then in the new order, we have

$$\ldots \leq y_{(n-2)} \leq y_{(n-1)} \leq r \leq y_{(n)}.$$

- Thus, we have $x_{(n-1)} = r$ and so, $y_{(n-2)} \leq r \leq y_{(n)}$.
- Hence, the desired inequality is satisfied.
- Finally, if $r < y_{(n)}$, then in the new order, we have

$$\ldots \leq y_{(n-2)} \leq y_{(n-1)} \leq y_{(n)} < r.$$

- Thus, we have $x_{(n-1} = y_{(n)}$.
- Hence, the desired inequality is also satisfied.

13. Proof: Part 1.2

- Let us now consider the case when the original top value $y_{(n)}$ is changed.
- Then, we have the following three sub-options for the replacing value r.
- If $r \leq y_{(n-2)}$, then in the new order, we have

$$\ldots \leq r \leq \ldots \leq y_{(n-2)} \leq y_{(n-1)}.$$

- Thus, we have $x_{(n-1)} = y_{(n-2)}$.
- Hence, the desired inequality is satisfied.

14. Proof: Part 1.2 (cont-d)

- If $y_{(n-2)} < r \leq y_{(n-1)}$, then in the new order, we have

$$\ldots \leq y_{(n-2)} < r \leq y_{(n-1)}.$$

- Thus, we have $x_{(n-1)} = r$.
- So, $y_{(n-2)} < x_{(n-1)} = r \leq y_{(n-1)}$ and since $y_{(n-1)} \leq y_{(n)}$, we have $y_{(n-2)} < x_{(n-1)} \leq y_{(n)}$.
- Hence, the desired inequality is satisfied.
- If $r > y_{(n-1)}$, then in the new order, we have

$$\ldots \leq y_{(n-2)} \leq y_{(n-1)} < r.$$

- Thus, we have $y_{(n-1)} = x_{(n-1)} = y_{(n-1)}$.
- By the definition of the ordering, we have

$$y_{(n-2)} \leq y_{(n-1)} = x_{(n-1)} \leq y_{(n)}.$$

- Hence, the desired inequality is also satisfied.

15. Proof: Part 1.3

- Let us now consider the case when the second largest value $y_{(n-1)}$ is changed.
- Then, we have the following three sub-options for the replacing value r.
- If $r \leq y_{(n-1)}$, then in the new order, we have

$$\ldots \leq y_{(n-1)} \leq y_{(n)}.$$

- Thus, we have $x_{(n-1)} = y_{(n-1)}$.
- Hence, the desired inequality is satisfied.

16. Proof: Part 1.3 (cont-d)

- If $y_{(n-1)} < r \leq y_{(n)}$, then in the new order, we have

$$\ldots \leq y_{(n-2)} < r \leq y_{(n)}.$$

- Thus, we have $x_{(n-1)} = r$.
- So, $y_{(n-1)} < x_{(n-1)} = r \leq y_{(n)}$ and since $y_{(n-2)} \leq y_{(n-1)}$, we have $y_{(n-2)} < x_{(n-1)} \leq y_{(n)}$.
- Hence, the desired inequality is satisfied.
- If $r > y_{(n)}$, then in the new order, we have

$$\ldots \leq y_{(n-2)} \leq y_{(n)} < r.$$

- Thus, we have $x_{(n-1)} = y_{(n)}$.
- Hence, the desired inequality is also satisfied.

17. Proof: Part 1.4

- Finally, let us consider the case when the third largest value $y_{(n-2)}$ is changed.
- Then, we have the following three sub-options for the replacing value r.
- If $r \leq y_{(n-1)}$, then in the new order, we have

$$\ldots \leq r \leq \ldots \leq y_{(n-1)} \leq y_{(n)}.$$

- Thus, we have $x_{(n-1)} = y_{(n-1)}$.
- Hence, the desired inequality is satisfied.

18. Proof: Part 1.4 (cont-d)

- If $y_{(n-1)} < r \leq y_{(n)}$, then in the new order, we have

$$\ldots \leq y_{(n-1)} < r \leq y_{(n)}.$$

- Thus, we have $x_{(n-1)} = r$.
- So, $y_{(n-1)} < x_{(n-1)} = r \leq y_{(n)}$ and since $y_{(n-2)} \leq y_{(n-1)}$, we have $y_{(n-2)} < x_{(n-1)} \leq y_{(n)}$.
- Hence, the desired inequality satisfied.
- If $r > y_{(n)}$, then in the new order, we have

$$\ldots \leq y_{(n-1)} \leq y_{(n)} < r.$$

- Thus, we have $x_{(n-1)} = y_{(n)}$.
- Hence, the desired inequality is also satisfied.

19. Proof: Part 2.1

- Let us now prove that every robust function has the desired form.
- Let us first prove that for every robust function, we have $f(x_1, \ldots, x_n) \le x_{(n-1)}$.
- Indeed, let us form the new tuple $(y_1, \ldots, y_n)$ by replacing its largest element $x_{(n)}$ with $x_{(1)} - 1$, i.e.,

$$(y_1, \ldots, y_n) = (x_1, x_2, \ldots, x_{(n)-1}, x_{(1)} - 1, x_{(n)+1}, \ldots, x_{n-1}, x_n).$$

- For this tuple:

$$y_{(1)} = x_{(1)} - 1 < y_{(2)} = x_{(1)} \le y_{(3)} = x_{(2)} \le \ldots \le y_{(n)} = x_{(n-1)}.$$

- The tuple $(x_1, \ldots, x_n)$ is obtained from the tuple $(y_1, \ldots, y_n)$ by changing a single element.
- By definition of a robust function, this implies that $f(x_1, \ldots, x_n) \le y_{(n)}$.
- Since $y_{(n)} = x_{(n-1)}$, we conclude that indeed $f(x_1, \ldots, x_n) \le x_{(n-1)}$.

20. Proof: Part 2.2

- Let us now prove that for every robust function, we have $f(x_1, \ldots, x_n) \geq x_{(n-1)}$.
- Indeed, let us form a new tuple $(y_1, \ldots, y_n)$ by replacing its smallest element $y_{(1)}$ with $y_{(n)} + 1$, i.e.,

$$(y_1, \ldots, y_n) = (x_1, x_2, \ldots, x_{(1)-1}, x_{(n)} + 1, x_{(1)+1}, \ldots, x_{n-1}, x_n).$$

- For this tuple:

$$y_{(1)} = x_{(2)} \leq y_{(2)} = x_{(3)} \leq \ldots \leq y_{(n-2)} = x_{(n-1)} \leq y_{(n-1)} = x_{(n)} < y_{(n)} = x_{(n)}+1.$$

- The tuple $(x_1, \ldots, x_n)$ is obtained from the tuple $(y_1, \ldots, y_n)$ by changing a single element.
- By definition of a robust function, this implies that $f(x_1, \ldots, x_n) \geq y_{(n-2)}$.
- Since $y_{(n-2)} = x_{(n-1)}$, we conclude that indeed $f(x_1, \ldots, x_n) \geq x_{(n-1)}$.

21. Proof: final stage

- From Parts 2.1 and 2.2, we can conclude that $f(x_1, \ldots, x_n) = x_{(n-1)}$.
- Our main result is proven.

22. Auxiliary result

- In the previous sections, we analyzed what happens when we change one of the inputs.
- What will happen if we change $k > 1$ inputs?
- In this case, we have the following result.
- Let $k > 1$ be a natural number.
- We say that a function $f(x_1, \ldots, x_n)$ is *k-robust* if:
  - for every tuple $(y_1, \ldots, y_n)$,
  - whenever we form a new tuple $(x_1, \ldots, x_n)$ by replacing k of its elements y_i with different numbers, then we will have

$$y_{(n-(k+1))} \le f(x_1, \ldots, x_n) \le y_{(n)}.$$

- *For every $k > 1$, the only k-robust function $f(x_1, \ldots, x_n)$ is the function $f(x_1, \ldots, x_n) = x_{(n-k)}$.*

23. Comment

- We can show that this is the best possible result: in general, it is not possible to get closer to the maximum.
- To be more precise, it is not possible to have a stronger inequality $y_{(n-k)} \le f(x_1, \ldots, x_n) \le y_{(n)}$.
- Indeed, let us take $n = 2k+1$ and $y_i = i$ for all i from 1 to n, and let us replace the top k values with 0s, i.e., let us have

$$(x_1, \ldots, x_n) = (y_1, \ldots, y_{k+1}, 0, \ldots, 0) = (1, \ldots, k+1, 0, \ldots, 0).$$

- In this case,

$$x_{(1)} = \ldots = x_{(k)} = 0 < x_{(k+1)} = 1 < x_{(k+2)} = 2 < \ldots < x_{(2k+1)} = k+1.$$

- So $f(x_1, \ldots, x_n) = x_{(n-k)} = x_{(k+1)} = 1$, but $y_{(n-k)} = y_{(k+1)} = k+1 > 1$, so the inequality $y_{(n-k)} \le f(x_1, \ldots, x_n)$ is not satisfied.

24. Proof of the auxiliary result

- This proof is similar to the proof of the main result, with the following two main differences.
- Let us replace $x_{(n)}$ with $y_{(1)} - 1$, $x_{(n-1)}$ with $y_{(1)} - 2$, $\ldots$, $x_{(n-i)}$ with $y_{(1)} - (i+1)$, $\ldots$, and $x_{(n-(k-1))}$ with $y_{(1)} - k$.
- In this case, $y_{(n)} = x_{(n-k)}$, so the k-robustness condition $f(x_1, \ldots, x_n) \le y_{(n)}$ implies that
$$f(x_1, \ldots, x_n) \le x_{(n-k)}.$$
- To prove that $f(x_1, \ldots, x_n) \ge y_{(n)}$, we replace $x_{(1)}$ with $y_{(n)} + 1$, $x_{(2)}$ with $y_{(n)} + 2$, $\ldots$, $x_{(i)}$ with $y_{(n)} + i$, $\ldots$, and $x_{(k)}$ with $y_{(n)} + k$.
- In this case, $y_{(n-(k+1))} = x_{(n-k)}$, so the k-robustness condition $f(x_1, \ldots, x_n) \ge y_{(n-(k+1))}$ implies that
$$f(x_1, \ldots, x_n) \ge x_{(n-k)}.$$
- From these two inequalities, we conclude that indeed $f(x_1, \ldots, x_n) = x_{(n-k)}$. The proposition is proven.

25. Bibliography

- S. Sun, M. Xian, A. Vakanski, and H. Ghanem, "MIRST-DM: Multi-Instant RST with Drop-Max Layer for Robust Classification of Breast Cancer", *Proceedings of the 25th International Conference on Medical Image Computing and Computer Assisted Intervention MICCAI 2022, Singapore, September 18–22, 2022*, Springer Lecture Notes in Computer Science, Vol. 13434, 2022, ISBN 978-3-031-16439-2, doi 10.1007/978-3-031-16440-8_39

26. Acknowledgments

This work was supported by:

- the AT&T Fellowship in Information Technology,
- the Institute for Risk and Reliability, Leibniz Universitaet Hannover, Germany,
- the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) Focus Program SPP 100+ 2388, Grant Nr. 501624329,
- the European Union under the project ROBOPROX (No. CZ.02.01.01/00/22 008/0004590),
- the Center of Excellence in Econometrics, Faculty of Economics, Chiang Mai University, Thailand,
- the Ho Chi Minh City University of Banking, Vietnam, and
- Thang Long University, Hanoi, Vietnam.