Why Skew-Normal Distributions and How They Are Related to ReLU Activation Function in Deep Learning

Damian Lorenzo Gallegas Espinosa,
Olga Kosheleva, and Vladik Kreinovich
University of Texas at El Paso, El Paso, TX 79968, USA
dlgallegose@miners.utep.edu, olgak@utep.edu, vladik@utep.edu

1. Why skew-normal distributions: a challenge

- In many practical situation, we have many small independent factors affecting the desired quantity.
- In such cases:
  - according to the Central Limit Theorem,
  - the resulting distribution is close to Gaussian (normal).
- It is known that:
  - the result of a linear function applied to a normal random variable
  - or, more generally, a linear combination of independent normal random variables
  - is still normal.
- However, not all distributions are normal.
- E.g., many empirical distributions are *skewed* (asymmetric), i.e., have non-zero third central moment.

2. Why skew-normal distributions: a challenge (cont-d)

- It is desirable to have a few-parametric generalization of the class of normal distributions that would allow to consider skewness.
- Many such generalizations are possible.
- Empirically, one of them – called skew-normal – has been most successful.
- Let us denote:
  - the probability density function (pdf) of the basic normal distrubution, with 0 mean and standard deviation 1 by $f(x)$, and
  - the corresponding cumulative distribution function by $F(x)$.
- Then the pdf $s(x)$ of skew-normal distrubution has the form $s(x) = f(x) \cdot F(\alpha \cdot x)$ for some α.
- How can we explain why this particular generalization turned out ot be the most successful?

3. Our explanation

- We need a non-normal distribution.
- We already have a normal distribution.
- So a natural idea is to apply some function to the normal random variable.
- This does not restrict the class of distributions, since:
  - all continuous distributions can be obtained from each other by applying some function;
  - this is, by the way, a usual way to simulate different distributions.
- Since applying a linear function will still keep it normal, let us apply a nonlinear function.
- From the practical viewpoint, the faster-to-compute this nonlinear function, the better.
- So what is the fastest-to-compute nonlinear function?

4. Our explanation (cont-d)

- In the computer, the fastest possible operations are:
  - unary minus (it changes just one bit) and
  - min and max (that require, on average, 2 bit operations).
- We can also have constants like 0.
- Addition and subtraction require as many bit operations as there are bits – i.e., 64 on most computers.
- Multiplication and division require even more bit operations.
- So, the fastest way is to compute min or max of x and $-x$, i.e., to compute $|x|$ or $-|x|$.
- (It would be even faster to take $\max(0, x)$, but this would not lead to a continuous random variable.)

5. Our explanation (cont-d)

- Interestingly:
  - if add $|x|$ to the set,
  - i.e., if we consider linear combinations of $|x|$ (for a normal random variable x) and independent normal random variables,
  - then we get exactly all skew-normal distributions.
- So, we have the desired explanation.

6. How is this related to ReLU?

- In a neural network, we have several units (called neurons) that perform some transformations.
- Some neurons process inputs.
- Others process results of the previous neurons, etc.
- If all neurons were linear, then we would only get linear functions, and many real-life dependencies are nonlinear.
- Thus, to be able to describe real-life dependencies, we need to add some nonlinear transformations.
- Similarly to the skew-normal case, we are looking for the fastest-to-compute transformations.
- The more inputs, the longer time is takes to process them.
- So the fastest are functions of one variable.

7. How is this related to ReLU (cont-d)

- Similar to the above, we conclude that the fastest-to-compute function is $\max(x, 0)$.
- We can also have $\min(0, x)$ which is equivalent from the viewpoint of further linear transformations: it is $-\max(0, -x)$.
- This is exactly:
  - the transformation provided by Rectified Linear (ReLU) neurons,
  - neurons that turned out to be most effective in machine learning.

8. Acknowledgments

This work was supported in part:

- by the US National Science Foundation grants:
  - 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science),
  - HRD-1834620 and HRD-2034030 (CAHSI Includes),
  - EAR-2225395 (Center for Collective Impact in Earthquake Science C-CIES),
- by the AT&T Fellowship in Information Technology, and
- by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) Focus Program SPP 100+ 2388, Grant Nr. 501624329,