Uncertainty Quantification for Results of AI-Based Data Processing: Towards Feasible Algorithms

C. Q. Lauter, M. Ceberio, and V. Kreinovich*

Department of Computer Science, University of Texas at El Paso
El Paso, Texas 79968, USA
*E-mail: vladik@utep.edu, www.cs.utep.edu/vladik

O. M. Kosheleva

Department of Teacher Education, University of Texas at El Paso
El Paso, Texas 79968, USA
E-mail: olgak@utep.edu

Need for indirect measurements. In many practical situations, we are interested in the value of a quantity $y$ that is difficult (or even impossible) to measure directly. Since we cannot measure $y$ directly, we measure it indirectly: we find easier-to-measure quantities $x_1, \ldots, x_n$ that are related to $y$ by a known dependence $y = f(x_1, \ldots, x_n)$, measure these quantities, and use the measurement results $\tilde{x}_1, \ldots, \tilde{x}_n$ to estimate $y$ as $\tilde{y} = f(\tilde{x}_1, \ldots, \tilde{x}_n)$. Computing $\tilde{y}$ is an important case of data processing.

Need for uncertainty quantification (UQ). Measurements are never 100% accurate, each measurement result $\tilde{x}_i$ is, in general, different from the actual (unknown) value $x_i$. As a result, even if the dependence $f$ is exact, the estimate $\tilde{y}$ is different from the actual value $y$. From the practical viewpoint, it is important to know how big can be the difference $\Delta y \overset{\text{def}}{=} \tilde{y} - y$. For example, if we are estimating the amount of water in an underground location as 100 units, then if it is $100 \pm 20$, this is worth exploring, while if it is $100 \pm 150$ then maybe there is no water at all, so we better perform more measurements before starting the expensive drilling process.

To estimate $\Delta y$, we need to have information about measurement errors $\Delta x_i \overset{\text{def}}{=} \tilde{x}_i - x_i$. Usually, we either know the probability distribution or we know the upper bound $\Delta_i$ on the absolute value $|\Delta x_i|$. In the second case, once we get the measurement result $\tilde{x}_i$, all we know is that the actual value $x_i$ is in the interval $[\tilde{x}_i - \Delta_i, \tilde{x}_i + \Delta_i]$; so this case is known as interval
uncertainty.

Traditional UQ techniques: a brief reminder. Based on the available information about $\Delta x_i$, most traditional UQ techniques call the program $f$ several times. When $n$ is small, we can use sensitivity analysis (i.e., in effect, numerical differentiation) to find the derivatives of $f$ with respect to the inputs, and then use these derivatives to explicitly estimate $\Delta y$ (i.e., either its standard deviation or its upper bound). When $n$ is large, this is rarely feasible – since in such cases, computing $f$ often takes hours on a high-performance computer, so we do not have the luxury of calling $f$ for each of thousands of variables. In such cases, we can use Monte-Carlo (MC) techniques – traditional MC techniques for probabilistic uncertainty and Cauchy-based MC techniques for the case of interval uncertainty.

AI-related challenges. AI techniques are becoming ubiquitous in data processing. Their results are good, but in terms of uncertainty quantification (UQ) they lead to new challenges. First, for some inputs $x_i$, we only have expert estimates, and these estimates are often formulated by using imprecise ("fuzzy") words from natural language. There are special techniques – known as fuzzy techniques – for translating these estimates into numerical machine-understandable form. In principle, there are UQ techniques for processing such fuzzy inputs – in effect, we need to select about ten certainty levels and perform interval estimations on each level. The challenge is that this multiplies the UQ time – which is often already long – by a factor of 10, which makes it not feasible.

Second, data processing is now often performed by a deep neural network (NN), with up to billions and trillions of parameters. In this case, repeating the computations several times – as in the traditional UQ techniques – is not possible.

What we propose. In this talk, we propose how to deal with both challenges. For fuzzy techniques, we can use the fact that all input fuzzy sets usually have similar shapes: e.g., all triangular or all trapezoid. We show that in this case – under the usual assumption that input errors are relatively small – it is possible to only perform computations for 1 or 2 certainty levels and then reconstruct the results for all other levels.

For neural data processing, we can use the fact that when a NN is trained, it already computes a lot of partial derivatives; so we can use these derivatives to estimate the derivatives of $f$ with respect to each $x_i$ – after which UQ is straightforward.