Questions and answers - Francois Drielsma Lecture

The following questions were submitted through Google Form. Some may have been answered in the Q&A session already. Nevertheless, we request our lecturers to provide written answers here for the benefit of those who could not attend that session. Thank you!

Slide not specified. How do you approach attaching an uncertainty on, say, a tag score from ParticleNet?

Question for Javier, please forward it to him.

Slide not specified. I'm wondering if your training dataset is from synthetic data or real observations/data with hand-labeling? If they are synthetic data, could you tell me the trained model performance in the real data--if there is a discrepancy, could you introduce how do you improve the robustness of the model trained on synthetic data? If they are real observations, could you let me know the approximated training set size and how does the size affect the model performance (i.e. how do you choose the current data set size and do you think increasing the set size would increase the performance or not?)

I will answer assuming this question pertains to the GNN-based clustering in LArTPCs, although my answer is mostly generic and applies to most supervised-learning tasks. The training dataset is synthetic; it consists of rasterized images of charged particle energy depositions in liquid argon, as simulated with Geant4. The validation procedure on data is similar to the one that one would use with traditional (read non-ML) reconstruction algorithms. It consists of evaluating well understood distributions on simulation and data, comparing them and looking for discrepancies. These metrics include but are not limited to:

- Momentum distribution of Michel decay electrons
- Pi0 invariant mass distribution
- Hand-labeled real data (vertex, PID, range, etc.)

The topic of addressing performance discrepancies between simulation and data is a very broad and interesting one. Generally, it stems from a legitimate difference between simulation and data, which should be addressed as much as possible with an appropriate suite of calibration targets. For what remains (calibration is never perfect), another way to mitigate them is by design, i.e. by engineering a reconstruction chain which is mostly insensitive to simulation details by making it perform generic tasks. For instance, an algorithm which does image classification on neutrino images is much more sensitive to changes in the kinematics distribution of the neutrinos in the simulation than an algorithm which does geometric tasks such as particle clustering, individual particle ID, etc. Another potential way to address them is by enforcing that your network does not learn features which are specific to the simulation, e.g. by using domain adversarial training (https://arxiv.org/pdf/1505.07818.pdf). The field of trying to solve the issue discussed here is called “domain adaptation” and there’s a lot of literature relating to this online, if you’re interested.

Slide 53. On this slide, I did not get why we are using Bell number here? Is there any other way we can find the number of possible partitions of a set of these objects?

The nth Bell number is simply the mathematical answer to the combinatorics question of “how many ways can I partition a set of N objects”? Wikipedia has a nice visual representation of all possible partitions of 5 objects (52 possibilities). The answer to that question is not task specific, there is no other way to find the number of possible partitions of a set. The issue with the number is how quickly it grows, which makes picking the best possible partition a tricky task that cannot be brute-forced, i.e. it is not practical to evaluate the partition score of all possible partitions.