Smart Control of Monte Carlo Simulations for Astroparticle Physics

Mirko Bunse,[1] Christian Bockermann,[1] Jens Buss,[2] Katharina Morik,[1] Wolfgang Rhode,[2] and Tim Ruhe [2]

[1]*A.I. Group, Computer Science Dept., TU Dortmund, Germany*

[2]*Chair for Experimental Physics 5, Physics Dept., TU Dortmund, Germany*

firstname.lastname@udo.edu

Abstract. Imaging Air Cherenkov Telescopes, such as VERITAS and MAGIC, rely on models for signal separation and energy estimation. These models are fit to simulated airshowers, each of which is associated with a single initial particle of known type and energy. Simulating an airshower from such primary particles requires the execution of fine-grained stochastic processes to emulate the behaviour of millions of secondary particles.

In this work we explore the smart control of simulator runs to reduce the overall runtime of data generation whilst producing the most useful example set for model fitting. We give an overview of the approach and provide an evaluation using data from the widely used CORSIKA shower simulation.

1. Introduction

Cherenkov astronomy studies high-energy gamma radiation from celestial objects to reason about the characteristics and properties of the radiation's sources. Gamma radiation as well as cosmic ray hadrons (mainly protons) induce a cascade of secondary particles when entering the Earth's atmosphere. Gamma-rays fully retain their directional information on their trajectory to Earth; hadrons do not. Distinguishing gamma-induced showers from showers induced by hadronic particles is thus a central task in Cherenkov astronomy (Petry & MAGIC Telescope Collaboration 1999; Kieda & VERITAS Collaboration 2004; Anderhub et al. 2013).

The application of machine learning models, such as neural networks or random forests, is a prominent approach to tackle this problem (Bock et al. 2004; Bockermann et al. 2015). These models are fit to sets of training data which contain examples that either belong to the gamma class or to the hadron class. Since such labeled data is not naturally available, it has to be synthetically generated based on knowledge about particle interactions in the atmosphere. Simulation programs like CORSIKA (Heck et al. 1998) and AIRES (Sciutto 2005) use Monte Carlo simulations to produce labeled data closely resembling the real data observeable by a telescope. These simulations involve computations that are inherently costly due to their Monte Carlo nature and the underlying models of particle interaction.

We propose the *smart control* of simulations in order to reduce the cost of the overall training set by generating only those examples that are most informative with respect to signal classification. The idea is based on the following formalization: Given some simulation process g, which is parameterized by a vector $\mathbf{p} \in \mathcal{P}$ of settings, each application of g produces a new synthetic observation $(\mathbf{x}, y) \in X \times Y$, i.e.

$$g : \mathcal{P} \to X \times Y,$$

where X is the set of real-valued vector representations of the simulated shower (usually after the application of a preprocessing step) and Y is the set of class labels. Each use of g comes at some cost by means of computing resources and simulation time. Since this cost can depend on the input vector, we may refer to it as $c(\mathbf{p}) \in \mathbb{R}$. For simplicity, however, this paper is limited to a uniform cost model $c(\mathbf{p}) = 1 \,\forall \mathbf{p} \in \mathcal{P}$. A set of parameter vectors $P \subset \mathcal{P}$ yields a data set $T = \cup_{\mathbf{p} \in P} g(\mathbf{p})$ used to train a prediction model $m : X \to Y$. In this work we explore different strategies to select a sequence of input vectors $\langle \mathbf{p} \rangle$ yielding a cost-quality tradeoff between simulation time and model accuracy.

2. Related Work

Smart control of data generation stems from *active class selection* problems (Lomasky et al. 2007). For these problems, training data can be generated from an arbitrary distribution of labels $y \in Y$, given a limited budget. Active class selection is the task of chosing that distribution of Y that produces the most cost-efficient training set. Smart control of data generation is a more general task; it considers any parameter in the generation of data and is not limited to the class label. The active selection of training examples goes back to *active learning* (Settles 2010). However, this well-studied field does not consider the generation of novel examples. Here, we extend the strategies presented by Lomasky et al. (2007) to sample from $\mathcal{P}$ and not only from Y.

Rodriguez-Lujan et al. (2014) present a problem equivalent to what we introduce here as smart generation of data. They sample actively from a parameter space $\mathcal{P}$ to select examples for the calibration of a gas sensor array. Such an array is learning to predict the type of gas it senses inside a test chamber. In their case, $\mathcal{P}$ consists of the label, i.e., the type of gas, and an additional parameter, the concentration of that gas. Examples are generated by injecting the configured gas into the chamber and recording the sensor's response. The same level of freedom is present in the simulation of air showers; the label and additional parameters can be chosen arbitrarily for the production of each training example. Unfortunately, the sampling method presented by Rodriguez-Lujan et al. (2014) is not practical. It requires multiple example generation runs in every iteration. The induced cost is huge because only the best run is maintained in each step. In contrast, we propose to select examples before they are simulated.

3. Smart Control of Data Generation

We iteratively choose parameter vectors, of which each $\mathbf{p} \in \mathcal{P}$ provides a good trade-off between its cost $c(\mathbf{p}) \in \mathbb{R}$ and its expected utility $u(\mathbf{p}) \in \mathbb{R}$ for the prediction task. Dividing the parameter space $\mathcal{P}$ into ranges of parameters $R \subseteq \mathcal{P}$ allows us to assess $u(\mathbf{p})$ by $u(R)$ if $\mathbf{p} \in R$, assuming that R is sufficiently homogeneous. $u(R)$ can then

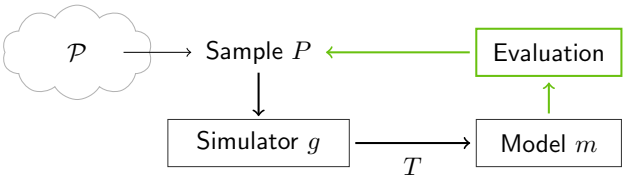

Figure 1. The Simulation g, parameterized by batches of inputs $P \subset \mathcal{P}$, produces sets of labeled instances $T = \cup_{\mathbf{p} \in P} g(\mathbf{p})$ that are used to train a model m. The batch of each next iteration is chosen based the expected utility and cost.

evaluate the existing model m_0 (trained on $T_0 \subset X \times Y$) on an existing evaluation test set $E_R = \cup_{\mathbf{e} \in R} g(\mathbf{e})$. In each iteration, the most promising parameter batch $P \subset \mathcal{P}$ is selected, $T = \cup_{\mathbf{p} \in P} g(\mathbf{p})$ is generated from that batch, and a new model m is trained on $T_0 \cup T$. Figure 1 shows this overall outline of our smart control framework.

The active class selection strategies proposed by Lomasky et al. (2007) find proportions $w(y)$ of classes $y \in Y$ that optimize m. We extend these strategies to weight parameter ranges $R \subseteq \mathcal{P}$ instead. Each parameter batch P is then sampled according to the range weights $w(R) := \frac{1}{u(R)} \cdot (\sum_{R' \subseteq \mathcal{P}} \frac{1}{u(R')})^{-1}$, i.e., according to the inverse relative utility of the existent model m for R. In our case, each R spans a range of logarithmic particle energies for a single particle type. The extended strategies are as follows:

- *Uniform* Each range has the same utility $u(R) := 1$. We use this uniform weighting as a baseline because it does *not* evaluate the model as proposed.
- *Inverse* Use the predictive accuracy as utility function $u(R) := Acc(R)$. The weights represent the relative inverse accuracy then. This strategy is motivated by the assumption that low accuracy is due to insufficient training data in the respective range.
- *Improvement* Like the inverse method, but based on the improvement of accuracy during the last iteration $u(R) := Acc(R) - lastAcc(R)$.
- *Redistricting* Count the number of examples n_R for which the prediction changed during the last iteration and let $u(R) := \frac{1}{n_R}$. The idea behind this strategy is that examples close to the decision boundary may be more informative than others. Ranges in which predictions frequently change are assumed to contain many examples close to that boundary.

4. Experiments

We evaluate the control strategies on data generated with CORSIKA (Heck et al. 1998) for FACT (Anderhub et al. 2013). A large set of that data was produced beforehand; we sample from that collection instead of running the costly simulation during the experiments. This setup directly translates to actually controlling the simulation in a practical application. We compare the strategies by their predictive accuracy on a separate test set. That set is uniformly distributed over the parameter ranges so that the accuracy is not biased towards any of the ranges.

Each strategy is plugged into our framework for 100 runs. We assess the mean accuracy of each strategy relative to the accuracy of our baseline, the uniform control

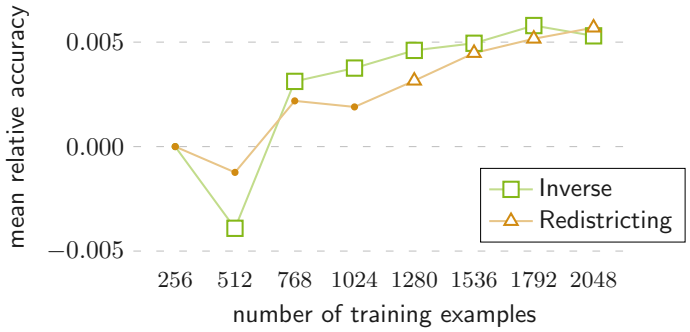

Figure 2. Mean accuracy of 100 experiments each – relative to the uniform control strategy. Squares and triangles show significant results. Dot marks show results that are not significant.

strategy. The significance of the results is computed via a t-test under the null hypothesis stating that the respective strategy performs equally good as the baseline. Figure 2 presents the results for the two strategies that outperformed the uniform control strategy. The *Improvement* strategy performed significantly worse.

5. Conclusion

Monte Carlo simulations allow us to parametrize the generated examples. The smart control framework proposed here leverages this concept to produce example sets that are better suited for model fitting than uninformed production strategies. Our experiments suggest that smart control can outperform a uniform distribution of simulation parameters. The benefit is significant, but rather small – about 0.5% in accuracy on a uniformly distributed test data set.

Acknowledgement. This work has been supported by the DFG, Collaborative Research Center SFB 876, project C3 (`http://sfb876.tu-dortmund.de/`).

References

Anderhub, H., et al. 2013, Journal of Instrumentation, 8, P06008. `1304.1710`
Bock, R. K., et al. 2004, Nuclear Instruments and Methods in Physics Research A, 516, 511
Bockermann, C., Brügge, K., Buss, J., Egorov, A., Morik, K., Rhode, W., & Ruhe, T. 2015, in European Conference on Machine Learning
Heck, D., Knapp, J., Capdevielle, J. N., Schatz, G., & Thouw, T. 1998, CORSIKA: a Monte Carlo code to simulate extensive air showers.
Kieda, D. B., & VERITAS Collaboration 2004, in AAS/High Energy Astrophysics Division #8, vol. 36 of Bulletin of the American Astronomical Society, 910
Lomasky, R., Brodley, C. E., Aernecke, M., Walt, D., & Friedl, M. 2007, in European Conference on Machine Learning, 640
Petry, D., & MAGIC Telescope Collaboration 1999, A&AS, 138, 601. `astro-ph/9904178`
Rodriguez-Lujan, I., Fonollosa, J., Vergara, A., Homer, M., & Huerta, R. 2014, Chemometrics and Intelligent Laboratory Systems, 130, 123
Sciutto, S. J. 2005, International Cosmic Ray Conference, 7, 9
Settles, B. 2010, University of Wisconsin, Madison, 52, 11