PROCEEDINGS OF SCIENCE

pLISA: a parallel Library for Identification and Study of Astroparticles

Cristiano Bozza[1]
University of Salerno and INFN Gruppo Collegato di Salerno
Via Giovanni Paolo II 132, Fisciano (84084), Italy
E-mail: cbozza@unisa.it

Chiara De Sio
University of Salerno and INFN Gruppo Collegato di Salerno,
now at University of Bristol, School of Physics, Tyndall Avenue, Bristol BS8 1TL, United Kingdom
E-mail: chiara.desio@bristol.ac.uk

Rosa Coniglione
INFN Laboratori Nazionali del Sud
Via Santa Sofia 62, Catania (95123), Italy
E-mail: coniglione@lns.infn.it

INFN has produced a Machine Learning library in Python that applies Convolutional Neural Networks to various common problems in the field of astroparticle identification and study in suitable detectors. The library itself makes few assumptions and has few requirements that are easily met in most astroparticle detectors. The Parallel Library for Identification and Study of Astroparticles (pLISA) has been tested against simulated events for the KM3NeT/ARCA detector. Interesting preliminary results have been obtained for up/down-going particle classification, muon/electron neutrino classification, Z component of the direction and energy estimation. Already with very little optimization work and using limited hardware resources (one NVidia GTX GPU), pLISA was shown to compete with traditional algorithms. The approach allows improvements and also portability to other detectors. pLISA is based on commonly used open source frameworks, which helps ensuring portability and scalability.

The New Era of Multi-Messenger Astrophysics - Asterics2019
25 – 29 March, 2019
Groningen, The Netherlands

[1]Speaker

PoS(Asterics2019)018

1. Introduction

The project of a **p**arallel **L**ibrary for **I**dentification and **S**tudy of **A**stroparticles (pLISA) has three motivations. First, GPUs provide enormous computing power for an affordable price. Second, reconstruction of the primary particle in astroparticle detectors involves complex algorithms, often with sub-optimal efficiency. Third, algorithm evolution only happens in steps and is largely dependent on personal initiative, whereas a computer farm in constant training might provide continuous improvements. pLISA added the ambitious goal of being usable in different contexts with different detector geometries and features, with the only price of retraining, which of course can be done automatically as more data are available from detectors and/or simulations.

2. Structure and basic concepts

In its present shape, pLISA is an organised collection of Python scripts built on top of popular open packages: Scikit-Learn[1], TensorFlow[2] and Keras[3]. All those building blocks are available as open source and are actively developed by large teams. While this ensures that they are dependable tools, the open source approach guarantees that in case of need each component can be replaced.

pLISA is explicitly based on the assumption that the machine learning engine uses Convolutional Neural Networks (CNNs) or one of their possible evolutions. Data must be represented as a multidimensional rectangular grid, suitable for CNNs. The structure of the network that transforms input data into a suitable output is free and depends on the application. Some of the software packages used apply limitations on the number of dimensions of the data grid. It is expected that such limitations will be removed in the next future. Dead areas/volumes in the detector structure, in the current version of pLISA, are not tagged, and there is no distinction between "not being sensitive" and "having no signal in one area/volume for a specific event". Piecewise-defined detectors are also possible by padding sensitive areas/volumes with insensitive grid elements. However, it is possible to add element sensitivity as an additional parameter in the set of input data. pLISA assumes that the primary information for each event comes as a set of *hits*, i.e. of weighted signals localized in space, time (and possibly also

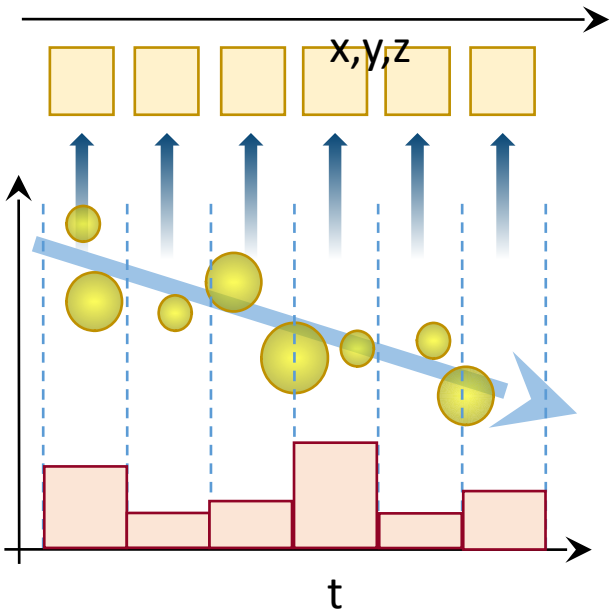

Figure 1: Binning event hits in space and time. The area of each circle may represent e.g. the energy deposition.

direction, e.g. for photomultipliers). The discretisation step maps the spatial and timing information onto a space-time grid encompassing the whole detector for the duration of the event. Hits in the same space-time bin are summed together (see Figure 1). This results in a partial loss of information but in a reduction of computational complexity and memory requirements, while speeding up the processing.

For an astroparticle detector, each event can be considered completely measured and studied if estimates are obtained of the primary particle type, energy and direction. Such goal may be achieved to a certain degree depending on the application. The choice of the inner layers of the CNN that is used to extract each parameter is not unique, and depends on the detector structure. Nevertheless, one may expect that most networks perform shape analysis on the configuration of hits and estimate the time evolution of the shape. Because the input neurons always receive a multidimensional image (including the case of a "movie", i.e. a set of images taken at regular time intervals), the purpose of most convolutional layers will be the extraction of synthetic information from local feature maps, interleaved with pooling/downsampling layers that discard intermediate results with low information content. In particular, robustness of the CNN to noise and to downgraded signals critically depends on how effective such layers are. It is well known that overtrained and overspecialized networks tend to produce outputs that rely on specific configurations and are unpredictably sensitive to small variations. The quality of the training datasets is of paramount importance and should span the parameter space as extensively as possible.

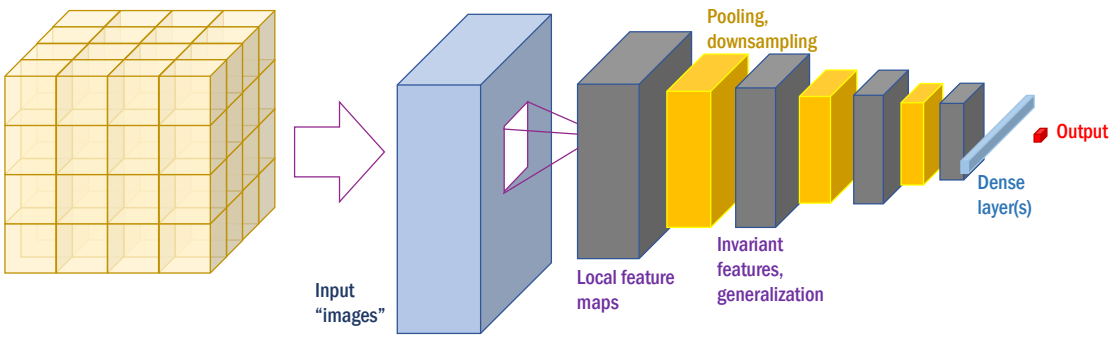

Figure 2: Intermediate and final layers of a CNN in pLISA.

Neurons involved in later stages implement extraction of invariant features. It is usually desirable that the output of the network has translational and rotational invariance with respect to the direction and position of the incoming primary particle. The training dataset should enhance such features. It is possible to "augment" the dataset by adding multiple translated and rotated copies of the same event to "teach" the network to work on global features rather than local configurations. Depending on the detector geometry, translational and rotational invariance may require more complex operations than just shifting the configuration of hits on a grid, and in some cases it may be needed to simulate the detector response for each copy of the event. In particular, a fully contained event may become partially contained. The final layers of the network are usually fully connected layers, mostly with the purpose of accurately reproducing the desired output. Depending on the application, it might be a nonlinear function of the outputs of neurons in the previous layers. Having a high number of neurons in the final layers helps

approximating virtually any output function. The typical network structure is sketched in Figure 2. Real implementations may vary by also allowing branched paths and alternate routes for the information flow to reconnect at later stages before the output.

3. Application to simulated events

Machine learning is a valuable tool if the estimation of output quantities can be truly independent on any *a priori* selection performed by "traditional" algorithms. pLISA does not use any preprocessed information. In order to develop the software and a library of networks[4,5], the KM3NeT/ARCA detector (for Astroparticle Research with Cosmics in the Abyss) [6,7] was used as a reference and events from standard simulations were considered. The training and validation data samples were made of mixed ν_μ and ν_e charged-current interactions in sea water. The first application developed for pLISA is particle identification. In a large fraction of cases it can be rather straightforward, as shown in Figure 3, but in other configurations the distinction may be more difficult. The related performances are shown in Figure 4. As expected, the CNN performs better in the cases in which the pattern of hits is better contained within the sensitive volume. Estimation of kinematical quantities (e.g. see Figure 5) for different primary particles is done by using just one network with neither *a priori* nor *a posteriori* particle type tagging. Traditional algorithms are often depending on the specific

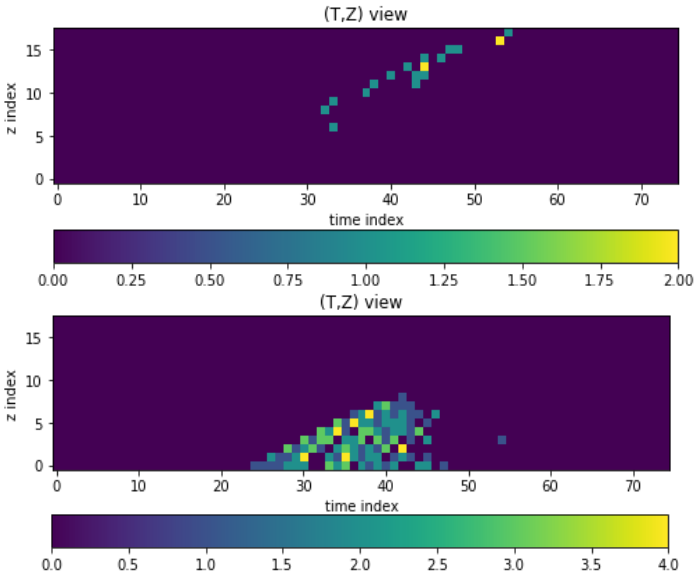

Figure 3: A muon neutrino event (top) and an electron neutrino event (bottom) from KM3NeT/ARCA. The colour scales shows the number of hits in the same space-time bin. Time and Z are replaced with the respective bin index.

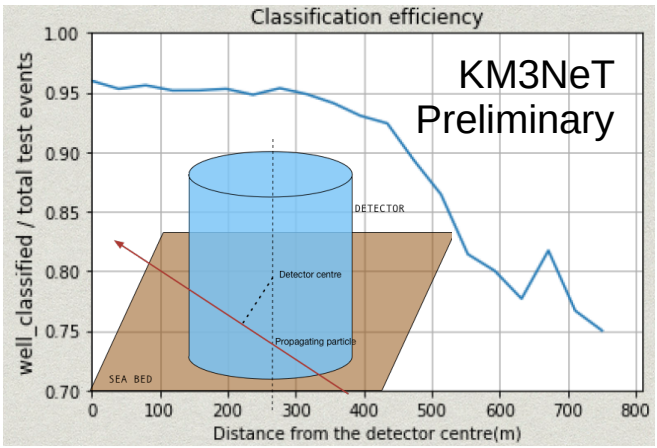

Figure 4: Particle identification efficiency as a function of the distance of the incident trajectory from the detector centre. The KM3NeT/ARCA detector is roughly cylindrical with 500 m diameter.

topology or particle: for example, the typical hit pattern of a high-energy muon produced in a charged-current neutrino interaction event is completely different from an electromagnetic shower started by an electron and different algorithms are used to estimate direction and energy. The CNN approach can provide direction and momentum or energy estimation for several particle categories in a single network. For comparison, a traditional algorithm has an RMS error of 0.001 on the estimation of the vertical director cosine for track-like events (mostly muons). pLISA has 0.002 on all events regardless of their classification, providing a result even when the traditional fitting algorithm does not converge.

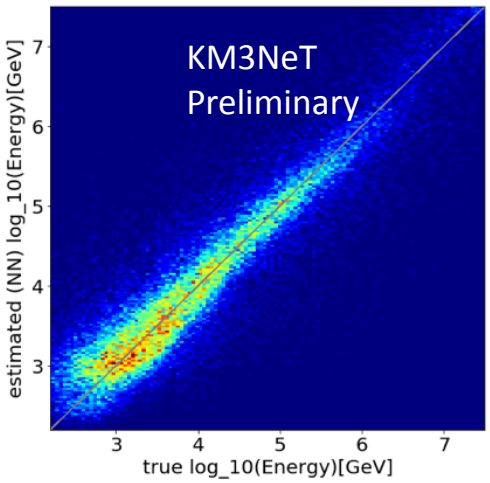

Figure 5: Energy estimated for a neutrino vs. true energy as known from simulation.

4. Conclusions and outlook

The results shown are very preliminary, as they were obtained with little training time. pLISA can already provide cross-checks to traditional custom algorithms, using instead widespread tools and technologies. The project is quickly evolving, and it is envisaged to improve by removing limitations such as the number of dimensions, while efficiency and detector distortion effects will soon be supported.

Acknowledgements

We acknowledge the support of the European Commission Grant 653477 (ASTERICS) in the framework of the Horizon 2020-INFRADEV programme.

References

[1] Pedregosa et al., *Scikit-learn: Machine Learning in Python*, *JMLR* **12** (2011) 2825-2830

[2] M. Abadi et al., *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*, https://arxiv.org/abs/1603.04467 (2015)

[3] F. Chollet et al., *Keras*, https://github.com/keras-team/keras (2015)

[4] C. De Sio, *Machine Learning in KM3NeT*, in proceedings of *VLVnT 2018, EPJ Web of Conferences* **207** (2019) 05004

[5] C. De Sio, *Event triggering and deep learning for particle identification in KM3NeT*, Ph.D. Thesis (2018) http://dx.doi.org/10.14273/unisa-1369

[6] S. Adrián-Martínez et al. (KM3NeT Coll.), *Letter of intent for KM3NeT 2.0*, *J. Phys. G* **43** (2016) 1

[7] S. Aiello et al. (KM3NeT Coll.), *Sensitivity of the KM3NeT/ARCA neutrino telescope to point-like neutrino sources*, *Astrop. Phys. G* **111** (2019) 100