GammaLearn - first steps to apply Deep Learning to the Cherenkov Telescope Array data

Thomas Vuillaume[1,*], *Mikael* Jacquemont[1,2,**], *Luca* Antiga[3], *Alexandre* Benoit[2], *Patrick* Lambert[2], *Gilles* Maurin[1], *Giorgia* Silvestri[3], and the CTA consortium

[1]Univ. Grenoble Alpes, Univ. Savoie Mont Blanc, CNRS, LAPP, 74000 Annecy, France
[2]Univ. Savoie Mont Blanc, CNRS, LISTIC, 74000 Annecy, France
[3]Orobix, 24121 Bergamo BG, Italy

Abstract. The Cherenkov Telescope Array (CTA) is the next generation of ground-based gamma-ray telescopes for gamma-ray astronomy. Two arrays will be deployed composed of 19 telescopes in the Northern hemisphere and 99 telescopes in the Southern hemisphere.

Due to its very high sensitivity, CTA will record a colossal amount of data that represent a computing challenge to the reconstruction software. Moreover, the vast majority of triggered events come from protons that represent a background for gamma-ray astronomy.

Deep learning developments in the last few years have shown tremendous improvements in the analysis of data in many domains. Thanks to the huge amount of simulated data and later of real data, produced by CTA, these algorithms look well-suited and very promising. Moreover, the trained neural networks show very good computing performances during execution. Here we present a first study of deep learning architectures applied to CTA simulated data to perform the reconstruction of the particles energy and incoming direction and the development of a specific framework, GammaLearn, to accomplish this task.

1 The Cherenkov Telescope Array

The Cherenkov Telescope Array (CTA) is the next generation of Imaging Atmospheric Cherenkov Telescopes (IACT) used for gamma-ray astronomy. CTA aims at studying the very-high-energy universe, such as cosmic ray origins, astrophysical phenomena, fundamental physics and cosmology [1].

CTA will be composed of two arrays of IACTs in two different observing locations (one in each hemisphere) in order to be able to observe the full sky and achieve different science goals. Today in pre-production phase, CTA is planned to acquire its first observational data in the coming years.

1.1 IACT data reconstruction

The high sensitivity of CTA will lead to the acquisition of unmatched volumes of raw data in gamma-ray astronomy with several PB per year. Such volumes of data require complex

---

*e-mail: vuillaume@lapp.in2p3.fr
**e-mail: jacquemont@lapp.in2p3.fr

systems to handle, transfer and archive the data, but also efficient analysis methods without sacrificing the instrument's performances.

Typical IACTs are composed of an array of telescopes operating as a single instrument to observe atmospheric showers generated by high-energy particles or photons entering the atmosphere. For gamma-ray astronomy, the showers initiated by charged particles represent a background that must be classified as such and eliminated to retrieve the gamma-ray light emitted by astronomical sources.

By combining the information from several telescopes, one can use stereoscopic reconstruction to derive the physical parameters (arrival direction, energy, particle type) of the original particle. This reconstruction is a complex but central process of the instrument. Recent advances in machine learning led to believe the stereoscopic reconstruction could be handled using convolutional neural networks (CNN) and some work has been done in this regard, in CTA [2, 3], or in other IACT [4].

These CNN have presented great performance improvements in many domains, including high-energy physics [5–7] and astronomy [8, 9], being able to extract more information from raw data than traditional methods based on human selected features. Moreover, they present the advantages to learn directly from data and therefore to replace complex processing pipelines and to be very fast during the prediction (a.k.a inference) step, thus making them usable at speeds comparable to data acquisition ones.

2 Deep learning for CTA

Training deep networks for regression and classification requires large amounts of annotated data. To do so in the case of CTA, we will use simulated monte-carlo data produced to understand the instrument's response function. Two main issues arise when trying to apply Deep Learning techniques to these data however:

- CTA will be composed of cameras of different types, some with unconventional images in the sense that they do not present cartesian grids of pixels and are therefore not adapted to standard Deep Learning frameworks

- CTA, alike other IACT, takes advantage of the stereoscopic information to reconstruct the primary particle parameters. A way to combine several images coming from different telescopes and presenting different characteristics (shapes, sizes...) needs to be found.

In order to tackle these issues, the GammaLearn project, born from a collaboration of the Laboratory of Annecy of Particle Physics (LAPP), the Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance (LISTIC) and an industrial partner specialized in Deep Learning solutions for manufacturers, Orobix, has been funded. We present here the preliminary stages of two solutions in development, to process unconventional images and take advantage of the stereoscopic information with Deep Learning.

2.1 Convolution and pooling for unconventional images

Convolution and pooling operations, which are core components of Deep Learning, as well as image processing tools in general have been developed to process rectangular and regular pixel grid images. However, some imaging sensors, including CTA LSTCam[10] and NectarCam[11], produce unconventional images. A common approach, allowing to use traditional image processing frameworks and in particular convolutional neural network ones, is to re-sample these unconventional images into ones with a rectangular and regular grid and a rectangular shape. Another approach, for unconventional images with regular grid,
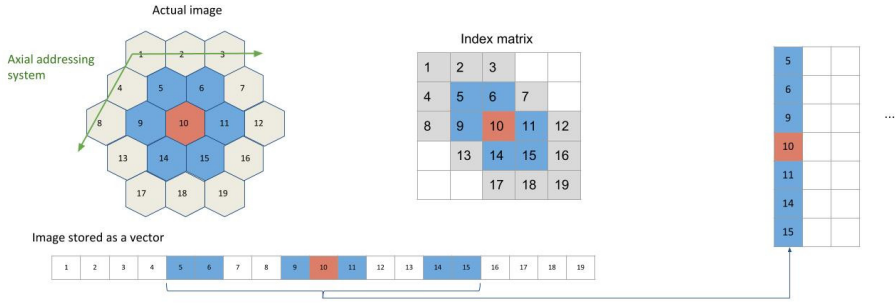
Figure 1. Building the indices matrix for hexagonal grid images. Assuming that the image is stored as a vector and that we know the position of each pixel in the axial addressing system, we can build an index matrix. With the appropriate kernel, in blue, we can identify the neighbours of each pixel of interest, in red, and fill the indices matrix.
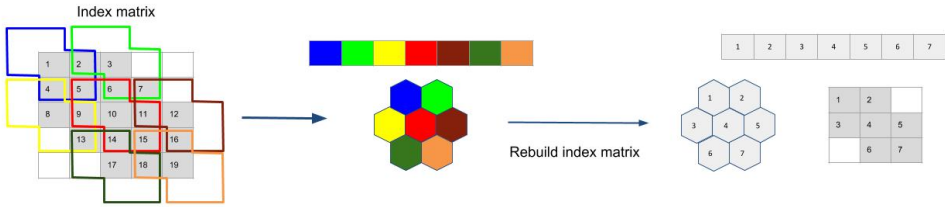
Figure 2. Rebuilding the index matrix after pooling (with hexagonal kernel).

for example hexagonal grid images, is to shift them into a Cartesian grid via a geometrical transformation.

However, such approaches needs additional image processing, generally performed on the CPU, slowing down both the training and inference phases of Deep Learning. Moreover, these approaches may result in image shape and size changing, and may introduce distortions that can potentially affect the performance of the method for its dedicated task.

In order to prevent these drawbacks, we propose to re-implement the convolution and pooling operators (thereafter denoted indexed operations) in the following way. Given that the neighbors of each pixel of interest is known and provided, we first build a matrix of neighbors indices for each pixel of interest (including itself). Then, we re-arrange the data according to this indices matrix and apply the classical general matrix multiply operation GEMM [12] for convolution or any suitable function (max, average, softmax) for pooling.

In Fig. 1 we exhibit an example of how to build the indices matrix for hexagonal grid images. After a pooling operation, the index matrix needs to be rebuild, as shown in Fig. 2.

This work will also be presented in more details in a paper, that has been accepted for publication for VISAPP 2019 conference.

2.2 Stereoscopy

Stereoscopy is an essential part of the IACT reconstruction methods. Indeed, single images only give a limited quantitiy of information compared to the association of images from

several telescopes combined with the position of these telescopes. However, traditional Deep Learning networks have been developed to deal with single images. An adaptation of these networks and establishment of new methods based on CNN is therefore required. [4] has shown that LSTM networks could be used to combine the images from several telescopes. In this preliminary work, we have considered a simpler solution consisting in duplicating the convolution part of the network (that learns a representation of the input) as many times as the number of telescopes, and then concatenating the obtained representations to feed the dense part of the network, leading to the architecture presented Fig. 3. This implies to feed the branches of telescopes that didn't trigger with inputs of zeros. The main disadvantage of this method is the number of telescope branches increasing dramatically when taking into account the whole CTA, leading to feeding the model with sparse inputs (as only several telescopes trigger for the same event). As we focus for this work on the four LSTs of CTA, this is not an issue.
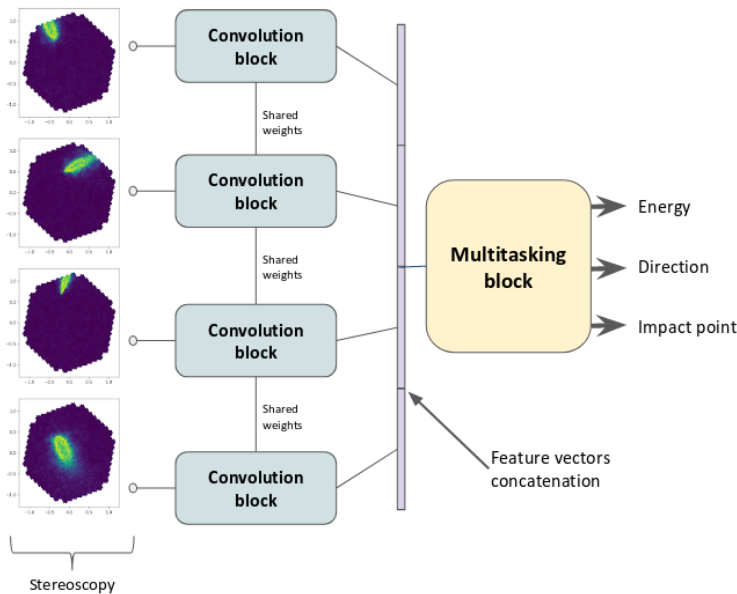
Figure 3. Preliminary architecture developed for the stereoscopic analysis of CTA data. For this preliminary work we focused on the four LSTs of the CTA North site leading to 4 entries to the network. Each entry is fed to the same convolution block (e.g. the convolutional part of VGG or a ResNet, in this case four convolutional layers with maxpooling and batchnorm). The feature vectors thus created are concatenated and the result is fed to the multitasking block (a fully connected network) inferring at the same time the energy, the arrival direction and the virtual (as the shower doesn't reach the ground) impact point of the shower event.

3 GammaLearner framework

The aim of the GammaLearn project is to find the best possible neural networks for gamma / cosmic rays separation and gamma parameters reconstruction. As the Deep Learning is a very empirical process, many hyperparameters and parameters initialization combinations will be explored and hopefully fine tuned. This represents a lot of learning cycles, and highlights the

need of a tool to ease this process.

The GammaLearner framework has been designed to address this issue, automatically dealing with bookkeeping all the experiments information. Moreover, it enables the use of the indexed operations and stereoscopy introduced before and developed especially for CTA in a more friendly manner.

3.1 GammaLearner components

As shown on Fig. 4, GammaLearner relies on PyTorch [13] for the Deep Learning fundamental components (tensor objects, automatic differentiation) augmented by the indexed convolution, described in 2.1, to handle unconventional images. It uses Ignite [14] for the organization of the training process itself, Tensorboard [15] (via TensorboardX [16]) and Gammaboard as visualization tools.

GammaLearner consists of an experiment runner tool, the engine of the framework, and a collection of functions and classes needed for the Deep Learning process to:

- load datasets,
- pre-process data (filter, augment, transform),
- train, validate and test networks,
- monitor the training process,
- visualize training results (Gammaboard).
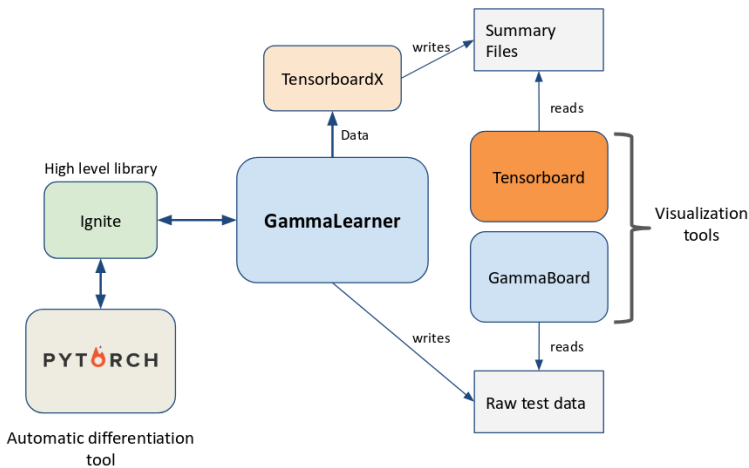
Figure 4. Description of the GammaLearner framework.

3.2 Experiment work-flow

To launch an experiment, i.e. train and test a neural network, one fills an experiment settings file (a python script) describing the conditions of the experiment : location of the data and data pre-processing, loss function, optimizer, network parameters (related to the network) etc.. If specific functions or classes are needed for this experiment and are not yet in the collection of files provided in the framework (i.e. criterions.py, optimizers.py, steps.py and plots.py), one

can simply write them in these files. One also defines the network to train, as a Python class inheriting from PyTorch nn.Module class to benefit from automatic differentiation. Then one starts the experiment runner, the settings and the network are saved for reproducibility, and the process runs to the end by its own.

During the training process, various metrics can be monitored via a log file and Tensorboard:

- loss,
- accuracy (in case of a classification task),
- features and convolution kernels,
- weights and gradients of the network,
- hardware usage.

Once the network is trained, the raw data of the test phase and the network are saved (network checkpoints can also be saved during the training). The test results can then be analyzed with the Gammaboard tool, that plots end result curves (energy and angular resolution, migration matrices) and allows an easy comparison between experiments and CTA requirements/performances.

4 Preliminary results

The architecture described Fig. 3 has been trained on CTA simulation data (90k gamma diffuse events from La Palma Prod3b[1], with the four LSTs of the chosen layout triggered) exhibiting an ability to learn, as shown in Fig. 5. The same architecture has then been used to compare the performances of the square kernels (3×3 pixels) and hexagonal kernels (nearest neighbors) for the convolution and the pooling layers on the hexagonal grid images of the LSTs. As shown in Fig. 6, the global loss (cumulative loss of every task, i.e. energy, direction and virtual impact point) is lower for hexagonal kernels, as expected as they match the pixel organization of the images. Indeed, the neighbors defined by the hexagonal kernel on a hexagonal grid image are the real neighbors of the pixel of interest, all with the same distance to it.
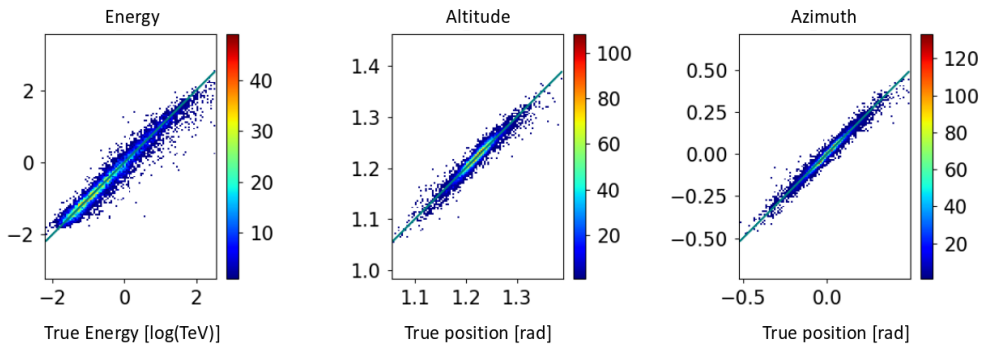
Figure 5. Migration matrices of the stereoscopic architecture described in 2.2 when trained on CTA gamma diffuse simulations (LST only). The Y-axis is the prediction of the network while the X-axis is the ground truth. The closer to the red line are the points, the better.

---

[1]The Prod3b corresponds to a batch of simulated events in order to choose the final layout of the different telescopes on both sites.
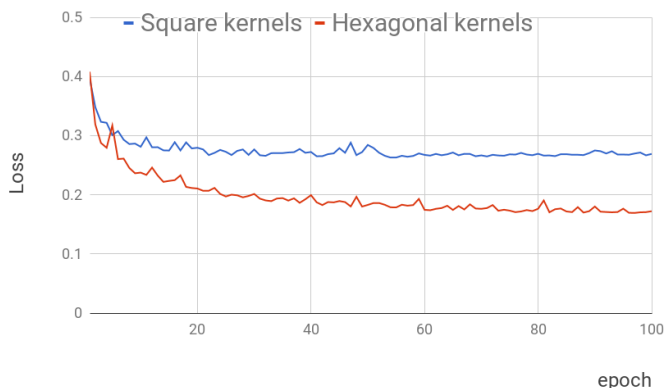
Figure 6. Global loss comparison between square kernels and hexagonal kernel during training. The X-axis represents the epochs of the training process, the Y-axis the loss value.

5 Conclusion

Machine learning is likely to gain more and more momentum in the coming years as it enables to deal with huge amount of data in an performant and efficient way. The impressive results obtained in other domains lead us to believe IACT reconstruction performances can be improved thanks to Deep Learning techniques such as deep convolutional neural networks.

We have presented in this work the issues specific to applying Deep Learning to CTA data (images shapes and pixels organisation and stereoscopy) and have proposed some approaches to solve these issues and applied them to show preliminary results on the regression of particles direction (altitude, azimuth) and energy.

This work needs to be developed and compared to state-of-the-art methods to show a potential advantage of Deep Learning. Many other challenges will likely need to be overcome (such as transfer learning from simulation to real data, robustness to real life noise) to apply these techniques to CTA real data.

Acknowledgement

References

[1] B.S. Acharya et al. (Cherenkov Telescope Array Consortium) (2017), `1709.07997`

[2] D. Nieto, A. Brill, B. Kim, T.B. Humensky, f.t. Cherenkov Telescope Array, ArXiv e-prints (2017), `1709.05889`

[3] S. Mangano, C. Delgado, M.I. Bernardos, M. Lallena, J.J. Rodríguez Vázquez, *Extracting Gamma-Ray Information from Images with Convolutional Neural Network Methods on Simulated Cherenkov Telescope Array Data*, in *Artificial Neural Networks in Pattern Recognition*, edited by L. Pancioni, F. Schwenker, E. Trentin (Springer International Publishing, Cham, 2018), pp. 243–254, ISBN 978-3-319-99978-4

[4] I. Shilon, M. Kraus, M. Büchele, K. Egberts, T. Fischer, T. Lukas Holch, T. Lohse, U. Schwanke, C. Steppa, S. Funk, ArXiv e-prints (2018), `1803.10698`

[5] J. Cogan, M. Kagan, E. Strauss, A. Schwarztman, Journal of High Energy Physics **2015**, 118 (2015), `1407.5675`

[6] J. Cogan, M. Kagan, E. Strauss, A. Schwarztman, Journal of High Energy Physics **2015**, 118 (2015)

[7] M. Andrews, M. Paulini, S. Gleyzer, B. Poczos, arXiv e-prints arXiv:1807.11916 (2018), `1807.11916`

[8] Q. Feng, T.T.Y. Lin, VERITAS Collaboration, *The analysis of VERITAS muon images using convolutional neural networks*, in *Astroinformatics*, edited by M. Brescia, S.G. Djorgovski, E.D. Feigelson, G. Longo, S. Cavuoti (2017), Vol. 325 of *IAU Symposium*, pp. 173–179, `1611.09832`

[9] K. Schawinski, C. Zhang, H. Zhang, L. Fowler, G.K. Santhanam, **467**, L110 (2017), `1702.00403`

[10] G. Ambrosi, Y. Awane, H. Baba, A. Bamba, M. Barceló, U.B. De Almeida, J. Barrio, O.B. Bigas, J. Boix, L. Brunetti et al., arXiv preprint arXiv:1307.4565 (2013)

[11] J. Glicenstein, M. Barcelo, J. Barrio, O. Blanch, J. Boix, J. Bolmont, C. Boutonnet, S. Cazaux, E. Chabanne, C. Champion et al., ArXiv e-prints (2013), `1307.4545`

[12] R. van de Geijn, K. Goto, *BLAS (Basic Linear Algebra Subprograms)* (Springer US, Boston, MA, 2011), pp. 157–164, ISBN 978-0-387-09766-4, `https://doi.org/10.1007/978-0-387-09766-4_84`

[13] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, *Automatic differentiation in PyTorch*, in *NIPS-W* (2017)

[14] *Ignite*, `https://github.com/pytorch/ignite` (2017–2018)

[15] GoogleInc., *Tensorboard*, `https://github.com/tensorflow/tensorboard` (2015–2018)

[16] *Tensorboardx*, `https://github.com/lanpa/tensorboardX` (2017–2018)