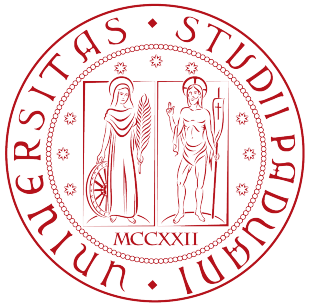

CONVOLUTIONAL NEURAL NETWORK SINGLE-TELESCOPE RECONSTRUCTION FOR THE LARGE SIZE TELESCOPE OF CTA

NICOLA MARINELLO
Department of Information Engineering
Università degli Studi di Padova

Ingegneria delle Telecomunicazioni
8 July 2019

Academic Year 2018 - 2019

Supervisor: Alessandro Chiuso
Co-Supervisor: Stefano Ghidoni

Co-Supervisor: Rubén López-Coto
Istituto Nazionale di Fisica Nucleare

To confine our attention to terrestrial matters
would be to limit the human spirit.

— Stephen Hawking

Dedicated to my family.

DECLARATION

I herewith declare that I have produced this thesis without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This thesis has not previously been presented in identical or similar form to any other Italian or foreign examination board.

Padova, 8 July 2019

Nicola Marinello

ABSTRACT

Very High Energy γ-ray astronomy explores the highest energy radiation coming from the non-thermal most violent processes in the Universe as that proceedings from black hole vicinities or stellar explosions. These highly energetic photons produce a cascade of particles when they reach the Earth and interact with the molecules in the atmosphere. These particles move faster than the speed of light in the atmosphere and produce a dim and fast flash of blue Cherenkov light that is detected by Imaging Atmospheric Cherenkov Telescopes (IACTs). When a telescope detects an event, produces an image of it that can then be analyzed to infer the event properties. The Cherenkov Telescope Array (CTA) is the consortium constructing the next generation of IACTs, composed of 99 telescopes in the Southern Hemisphere site (Atacama Desert, Chile) and 19 telescopes in the Northern Hemisphere site (La Palma, Spain). The two observatories will be composed of telescopes of three different sizes, being the largest telescopes (23 m diameter) called Large Size Telescopes (LSTs) and specialized in detecting the faintest possible light to lower the energy threshold of the system. To detect a signal using this technique it is very important to separate between the events produced by a γ ray and the background produced by hadrons, that is detected a signal to background ratio lower than $1/1000$. Currently, the standard algorithm used for γ/hadron separation and event reconstruction is the Random Forest (RF) which relies on a set of parameters derived from the event image, known as Hillas parameters. This algorithm is very robust, however it lacks of pixel-wise information that is lost during the image parameterization. Nevertheless recent developments on Deep Learning (DL) techniques such as Convolutional Neural Network (CNN) made them more attractive to become a standard for this kind of analysis. CNNs are able to analyze the full image and get rid of any kind of parameterization. They can perform classification and regression tasks and therefore can be applied to separate γ initiated from hadronic initiated events and to reconstruct their properties, as energy and incoming direction.
In this thesis I propose a full Convolutional Neural Network Single-Telescope Reconstruction analysis chain, for the LST of CTA. We compare both simple and state-of-the-art architectures for γ/hadron separation, energy and direction reconstruction, showing that our analysis chain significantly outperforms the RF algorithm in all three tasks.
The thesis is organized as follows: in Chapter 1 I introduce γ-ray astronomy and the imaging atmospheric Cherenkov technique, in Chapter 2 I briefly talk about DL and review the architectures used in the analysis chain. In Chapter 3 I discuss about the dataset used

and how the images are interpolated, in Chapter 4 I present the full analysis chain and compare the performances with the RF while in Chapter 5 I conclude the thesis with a glimpse to possible future works.

SOMMARIO

L'astronomia a raggi γ ad alte energie studia la radiazione energetica proveniente dai più violenti processi non termici dell'universo, come quella che si origina da buchi neri o esplosioni di stelle. Questi fotoni ad alta energia producono una cascata di particelle quando raggiungono la terra e interagiscono con le molecole dell'atmosfera. Queste particelle si propagano ad una velocità maggiore di quella della luce nell'atmosfera e producono un debole e veloce flash di luce blu, che viene osservata dai telescopi Cherenkov, chiamati anche Imaging Atmospheric Cherenkov Telescopes (IACTs). Una volta osservato l'evento, il telescopio produce un immagine che verrà successivamente analizzata per dedurre le proprietà dello stesso. Cherenkov Telescope Array (CTA) è il consorzio che si sta occupando di costruire la prossima generazione di telescopi Cherenkov. Verranno costruiti due array e saranno composti da 99 telescopi nel sito dell'emisfero sud (Deserto di Atacama, Cile) e da 19 telescopi nel sito dell'emisfero nord (La Palma, Spagna) rispettivamente. I due osservatori saranno composti da telescopi di tre dimensioni diverse: il più grande sarà il Large Size Telescope (LST) con i suoi 23 m di diametro, specializzato nel rivelare i flash di luce più deboli, contribuendo ad abbassare la soglia di energia dell'intero sistema. Per rivelare un segnale utilizzando questa tecnica è molto importante separare eventi prodotti da raggi γ dagli eventi adronici di background, quest'ultimi molto più frequenti. Infatti il rapporto tra la frequenza di eventi di segnale ed eventi di background è inferiore a $1/1000$. Attualmente, l'algoritmo standard utilizzato per la separazione degli eventi di segnale e background e per la ricostruzione degli eventi è il Random Forest (RF), che si basa su un set di parametri estratti dall'immagine dell'evento, conosciuti come parametri di Hillas. Questo algoritmo, pur essendo molto robusto, non ha alcuna informazione sull'effettivo valore dei pixels dell'immagine dell'evento, che vengono persi durante l'estrazione dei parametri stessi. Ciononostante, i recenti sviluppi del Deep Learning (DL), anche grazie a strumenti come le Convolutional Neural Networks (CNNs), in grado di analizzare le immagini dell'evento senza basarsi su alcun tipo di parametrizzazione, ne hanno reso attrattivo l'utilizzo in questo tipo di analisi. Questi modelli possono svolgere operazioni di classificazione e regressione e possono essere dunque applicati per separare gli eventi prodotti da un raggio γ da quelli adronici. Inoltre possono essere impiegati per ricostruire la loro energia e la direzione di arrivo.
In questa tesi propongo una ricostruzione completa a singolo telescopio basata su CNNs per l'LST di CTA. Confronteremo architetture semplici e all'avanguardia per la separazione di eventi di segnale da eventi di background e per la ricostruzione dell'energia e della direzio-

ne. Vedremo inoltre che i modelli hanno prestazioni significativamente maggiori del RF in tutte e tre le operazioni da eseguire per ricostruire interamente le proprietà dell'evento.
La tesi è organizzata come segue: nel Capitolo 1 introduco l'astronomia a raggi γ e l'imaging atmospheric Cherenkov technique, nel Capitolo 2 discuto brevemente il DL e introduco le architetture utilizzate. Nel Capitolo 3 discuto il dataset e di come le immagini vengono interpolate, nel Capitolo 4 presento i risultati e li confronto con le prestazioni del RF, mentre nel Capitolo 5 concludo la tesi, con uno sguardo ai possibili lavori futuri.

ACKNOWLEDGMENTS

This thesis represents the conclusion of my studies at the University of Padova, started in October 2013 and completed in July 2019. Nevertheless it is just the last portion of a considerable work made of commitment and sacrifice, essential to achieve such a great conclusion. During this work I was greatly supported by my supervisors, but especially during these years I had many people around me, as my family, my friends and classmates, without whom it would never have been possible to achieve this result. Everyone has contributed in their own way to this work and to make magnificent those that will remain the best years of my life.

First of all, I wish to thank my supervisor Rubén for his help during these months of work and for his endless support whenever I had a question or a problem. I also thank him for making me discover what it means doing research and for giving me the opportunity to closely work with an international scientific community.

I would like to acknowledge Prof. Alessandro Chiuso for giving me the possibility to work on a thesis which covers topics between astrophysics and engineering and for his valuable comments on this work.

A very special gratitude goes out to Prof. Mosè Mariotti, Prof. Michele Doro and to the research team of INFN of Padova for having included me in the group and for having provided equipment and staff that made this work possible.

The completion of my dissertation and of this great achievement would not have been possible without the endless support and nurturing of Eleonora, companion of hundreds of hours of study but especially a life-companion.

I am also grateful to all my classmates and friends: Luca, Marta, Laura, Alberto, Flavio, Federico, Riccardo, Enrico and Federico. Without your unwavering support this achievement would not have been possible.

I am extremely grateful to my friends of a lifetime Andrea and Simone whom I grew up with and with whom I shared some of the best periods of my life.

I also wish to thank my friends and the people I met during my Erasmus in Oslo in 2017. Particularly Marco, my classmate and flatmate at the student residence of Sogn, with whom I started my journey and from whom I learned so much in those months. Marco Pannone and Alberto, mates of parties and incredible adventures along with Marco, between the student residences of Sogn and Kringsjå.

Lastly, I acknowledge the support of NVIDIA Corporation with the donation of the Titan V GPU used for this research.

CONTENTS

1 INTRODUCTION 1
- 1.1 γ-ray astrophysics 2
- 1.2 Cherenkov effect 3
- 1.3 The imaging atmospheric Cherenkov technique 4
- 1.4 Cherenkov Telescope Array (CTA) 5
- 1.5 Large Size Telescope 7
- 1.6 IACT data analysis 8

2 DEEP LEARNING AND CONVOLUTIONAL NEURAL NETWORKS 11
- 2.1 Convolutional Neural Networks 11
- 2.2 Architectures review 12
  - 2.2.1 VGG-style networks 12
  - 2.2.2 Deep residual convolutional networks 12
  - 2.2.3 Densely connected convolutional networks 13
  - 2.2.4 Squeeze-and-Excitation networks 14
- 2.3 Common techniques 15
  - 2.3.1 Overfitting reduction 15
  - 2.3.2 Training acceleration 15
- 2.4 Deep Learning applied to IACT event reconstruction 16

3 DATASET AND IMAGE RESAMPLING 17
- 3.1 Image pre-processing 19
- 3.2 Event tensor composition 20
- 3.3 Potential drawbacks 21
- 3.4 Data generator 21

4 EVENT RECONSTRUCTION 23
- 4.1 γ/hadron separation 24
  - 4.1.1 Models performances 25
  - 4.1.2 Peak times channel impact on performances 26
- 4.2 Direction reconstruction 28
  - 4.2.1 Models performances 29
  - 4.2.2 Peak times channel impact on performances 30
- 4.3 Energy reconstruction 33
  - 4.3.1 Models performances 34
  - 4.3.2 Peak times channel impact on performances 36

5 CONCLUSIONS 39

A APPENDIX 43
- A.1 Convolutional Neural Networks architectures 43
  - A.1.1 VGG-9 43
  - A.1.2 ResNetF 44
  - A.1.3 ResNetFSE 45
  - A.1.4 ResNetH 46
  - A.1.5 DenseNet-BC $(L = 64, k = 12)$ 47

A.2 Data generator 48
A.3 Random Forest input parameters 50

BIBLIOGRAPHY 51

LIST OF FIGURES

Figure 1.1 Cosmic Ray Spectra of Various Experiments. (Hanlon, 2019) 2

Figure 1.2 On the left a gamma ray initiated shower, on the right an hadronic initiated shower. (Wagner, 2007) 3

Figure 1.3 Rendering of different sizes and types of Imaging Atmospheric Cherenkov Telescopes. (Courtesy of CTAO.) 4

Figure 1.4 Artistic picture of the Cherenkov light during the night. (Courtesy of CTAO.) 5

Figure 1.5 Event image produced by an hadronic initiated shower. 6

Figure 1.6 Event image produced by a γ-ray initiated shower. 6

Figure 1.7 Rendering of the CTA array in the Southern Hemisphere. (Courtesy of CTAO.) 7

Figure 1.8 Top panel: raw camera image. Bottom panel: cleaned image and fitted ellipse with some Hillas parameters. 10

Figure 2.1 Residual learning: a building block (He et al., 2016). 13

Figure 2.2 A 5-layer dense block with a growth rate of k = 4. Each layer takes all preceding feature-maps as input. (Huang et al., 2017). 14

Figure 2.3 A Squeeze-and-Excitation block (Hu, Shen, and Sun, 2018). 14

Figure 3.1 Dataset example images. On the left column an event from the `point-like` subset, on the right column an event from the `proton` subset. In the top row the image charges associated to the events are shown. In the bottom row the corresponding peak times images are represented. 18

Figure 3.2 Energy distribution of the datasets used. 19

Figure 3.3 Interpolated example images. On the top row an event and its peak times from the `point-like` subset, on the bottom row their interpolated counterpart. 20

Figure 4.1 Training accuracy (dotted) and validation accuracy (solid) of the different models during the training. 26

Figure 4.2 ROC curves obtained by testing the models on the test set. 27
Figure 4.3 ROC curves obtained by testing the ResNetFSE on the test set, for different energy bins. 27
Figure 4.4 Gammaness distribution produced by the ResNetFSE on the test set. 28
Figure 4.5 Training loss (dotted) and validation loss (solid) during the training. 31
Figure 4.6 Top panel: angular resolution across the energy spectrum for each model. Bottom panel: resolution improvement with respect to the RF for each model. 31
Figure 4.7 θ^2 for each energy bin. The red line represents the 68% containment of the histogram. 32
Figure 4.8 Training loss (dotted) and validation loss (solid) during the training. 35
Figure 4.9 Energy biases across the energy spectrum for each model. 35
Figure 4.10 Top panel: energy resolution across the energy spectrum for each model. Bottom panel: resolution improvement with respect to the RF for each model. 36
Figure 4.11 2D histogram of the true and reconstructed energy values by the model ResNetFSE on the test set. 37
Figure 4.12 Energy biases based on the predictions of the model VGG-9 on the test set. 38

LIST OF TABLES

Table 1.1 Energy domains of the γ-ray astrophysics 3
Table 3.1 Dataset for different event types. 17
Table 4.1 Summary results for separation on the test set. PT denotes if the model has access to the peak time information (i.e. the time gradient for the RF and the additional tensor dimension for the CNNs). 28
Table 4.2 Summary results for direction estimation on the test set. PT denotes if the model has access to the peak time information (i.e. the time gradient for the RF and the additional tensor dimension for the CNNs). 33

Table 4.3 Summary results for energy estimation on the test set. PT denotes if the model has access to the peak time information (i.e. the time gradient for the RF and the additional tensor dimension for the CNNs). 37
Table A.1 VGG-9 architecture. 43
Table A.2 ResNetF architecture. 44
Table A.3 ResNetFSE architecture. The ratio parameter of the SE block is indicated with r. 45
Table A.4 ResNetH architecture. 46
Table A.5 DenseNet-BC $(L = 64, k = 12)$ architecture. The network is 64-layer deep, with growth rate $k = 12$, bootleneck layers and compression $= 0.5$ 47

LISTINGS

Listing A.1 Classifier data generator 48
Listing A.2 Random Forest input parameters 50

ACRONYMS

CR Cosmic Ray

VHE Very High Energy

EM Electromagnetic

IACT Imaging Atmospheric Cherenkov Telescope

QE Quantum Efficiency

PMT Photomultiplier Tube

PWNe Pulsar Wind Nebulae

SNR Supernova Remnant

SMBH Super-massive Black Hole

GW Gravitational Waves

DM Dark Matter

CTA Cherenkov Telescope Array

MAGIC Major Atmospheric Gamma-ray Imaging Cherenkov Telescope

VERITAS Very Energetic Radiation Imaging Telescope Array System

H.E.S.S. High Energy Stereoscopic System

LST Large Size Telescope

MST Medium Size Telescope

SST Small Size Telescope

FoV Field of View

MC Monte-Carlo

ML Machine Learning

RF Random Forest

DL Deep Learning

CNN Convolutional Neural Network

ANN Artificial Neural Network

MLP Multilayer Perceptron

ROC receiver operating characteristic

AUC area under the curve

TPR true positive rate

FPR false positive rate

BN batch normalization

SE Squeeze-and-Excitation

SGD Stochastic Gradient Descent

RNN Recurrent Neural Network

AI Artificial Intelligence

NSB Night Sky Background

1 INTRODUCTION

In 1912 the Austrian physicist Victor Franz Hess analyzed the data acquired during a flight with an aerostatic balloon and he showed that the level of ionized particles decreased up to an altitude of about 1 km, but above that level, it increased considerably. This phenomenon suggested that the origin of this increase was an extraterrestrial radiation source later called Cosmic Rays (CRs). Such a scientific breakthrough won him the Nobel prize in Physics in 1936.

CRs are energetic particles coming from space to which the Earth and any other celestial bodies are exposed. They are mainly composed by protons and helium nuclei (99%), but also contain heavier nuclei, electrons, positrons, antiprotons and neutrinos. They are originated from many sources inside and outside of our Galaxy and their energy spectrum spans for more than twelve decades. CRs are the particles with the highest energies known and their study allows to understand the composition and evolution of the universe. For example they are used to infer useful properties about our Galaxy formation and composition.

The energy of CRs start at $\sim$ 100 MeV and reaches energies up to $\sim 10^{20}$ eV. The solar magnetic field blocks most of the particles coming from outside the solar system below 1 GeV, therefore CRs below this energy are of solar origin. Also, as the energy increases, the flux decreases as it can be seen in Figure 1.1.

γ rays, usually included inside the CR classification, are massless photons, as the ones conforming X-rays or visible light. They typically have energies above 0.1 MeV and wavelengths that are far shorter than those of the other types of electromagnetic radiation that we normally encounter; visible light, for example, has wavelengths more than 10^6 times longer than γ rays.

When CRs arrive to the Earth, they interact with the atmosphere particles and generate a particle cascade, also known as a particle shower. We can distinguish between two types of showers: Electromagnetic (EM) initiated showers and hadronic initiated showers. An EM shower is originated by a γ ray entering into the atmosphere with an energy $E \gtrsim 84$ MeV that undergoes $e^{\pm}$ pair creation in the presence of air nuclei. Then the $e^{\pm}$ pair creation produces new γ rays via bremsstrahlung that produce a cascade effect as showed in Figure 1.2. When the particle generating the shower is an hadron, it is called hadronic shower. In this case mostly pions are created (90%, in roughly equal proportions $\pi^0 : \pi^+ : \pi^- \rightarrow 1 : 1 : 1$) as well as kaons (10%) and light baryons ($p, \bar{p}, n, \bar{n}$) in a much smaller proportion. Hadrons and pions undergo

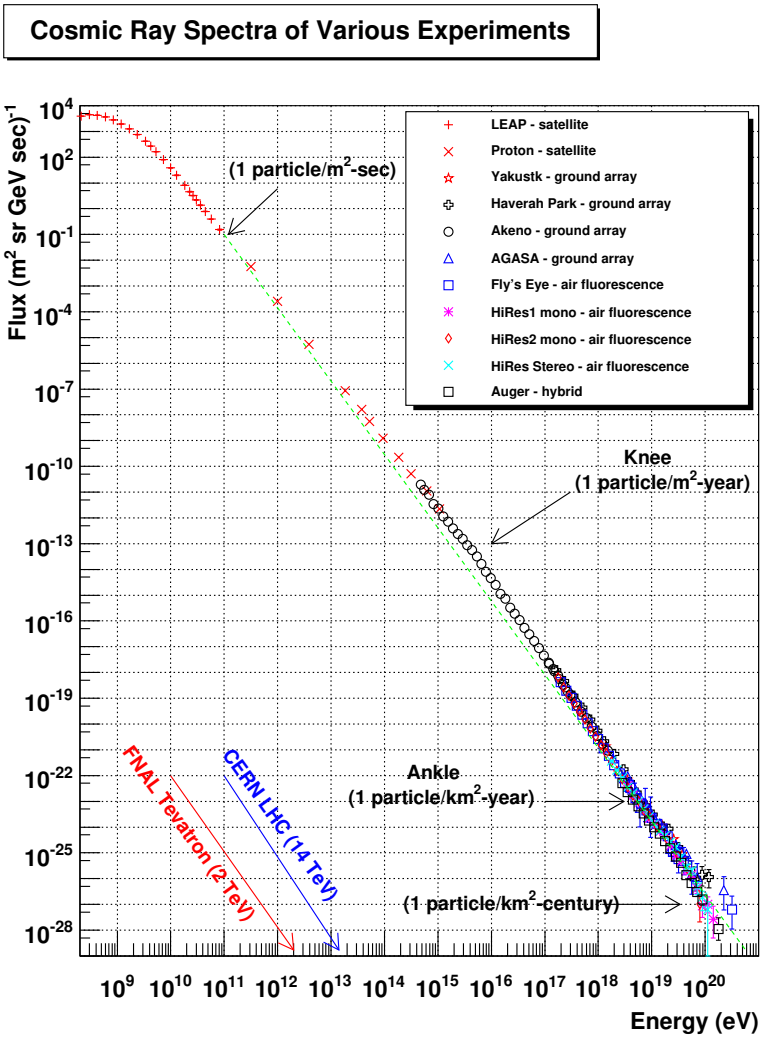

Figure 1.1: Cosmic Ray Spectra of Various Experiments. (Hanlon, 2019)

further collisions and several EM subcascades are generated as showed in Figure 1.2.

However γ rays represent a tiny fraction of the CRs arriving to the atmosphere and there are also natural sources of γ rays on Earth that include gamma decay from naturally occurring radioisotopes and also radiation from various atmospheric interactions with CRs particles.

1.1 γ-RAY ASTROPHYSICS

γ-ray astrophysics studies the electromagnetic spectrum beyond energies of ~ 1 MeV and it is divided into several energy domains, all of them shown in Table 1.1. γ rays study allow us to have a great view on the non-thermal Universe and to explore the most violent phenomena happening on very different astronomical scales (De Angelis and Mallamaci, 2018). They can be used to observe sources in our

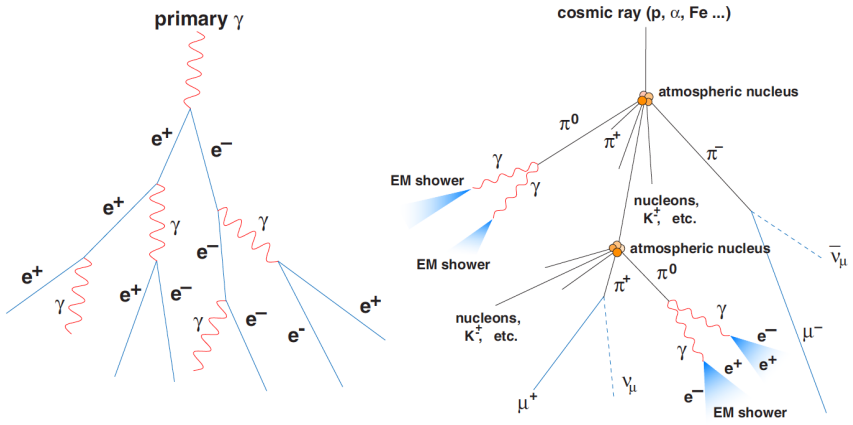

Figure 1.2: On the left a gamma ray initiated shower, on the right an hadronic initiated shower. (Wagner, 2007)

Galaxy as Pulsar Wind Nebulae (PWNe), Supernova Remnants (SNRs) and micro-quasars as well as sources outside the Milky Way as star formations or particles escaping Super-massive Black Holes (SMBHs) located in the center of some Galaxies. Also jointly observations with Gravitational Waves (GW) can provide complementary informations about the sources and their environments. Moreover it can helps scientists to answer to fundamental physics still opened questions as for example the possibility of the violation of the Lorentz invariance or the existence of exotic particles and objects, like axion-like particles and primordial black-holes. It could also helps to get a deeper knowledge of aspects related to Dark Matter (DM) particles or matter-antimatter asymmetry.

Name	Abbreviation	Energy range
low energy	LE	1 MeV - 30 MeV
high energy	HE	30 MeV - 50 GeV
very-high energy	VHE	50 GeV - 100 TeV
ultra-high energy	UHE	100 TeV - 100 PeV
extremely-high energy	EHE	$>$ 100 Pev

Table 1.1: Energy domains of the γ-ray astrophysics

1.2 CHERENKOV EFFECT

The Cherenkov effect has been experimentally detected by Pavel Cherenkov in 1934 (Cherenkov, 1934). When a charged particle moves in a dielectric material with a speed v larger than the phase velocity of light, namely if $v > c/n$, where n is the refraction index and c is the speed of light, it generates an electromagnetic wave also called

Cherenkov light. This radiation is emitted in the form of a cone at an angle θ such that:

$$\cos\theta = \frac{c}{vn(\lambda)} \tag{1.1}$$

where $n(\lambda)$ is the spectral index of the medium, which depends on the wavelength of the Cherenkov light. As the emitted radiation has the shape of a cone, a vertical ultrarelativistic particle illuminates a doughnut ring on the ground. This phenomenon occurs when a γ ray enters into the atmosphere: the charged particles $e^{\pm}$ moves at a speed $v > c/n$ through the air and produce the Cherenkov light. The same phenomenon occurs when hadrons interact with the atmosphere: a particle cascade is produced and these particles emit Cherenkov light.

As the energy of an incident γ-ray photon is proportional to the Cherenkov photon density detected on the ground, this can be used to reconstruct their energy. In the case of hadrons, this relation is not fulfilled.

1.3 THE IMAGING ATMOSPHERIC CHERENKOV TECHNIQUE

The Cherenkov effect produced by the γ rays that interact with the atmosphere can be captured by the Imaging Atmospheric Cherenkov Telescopes (IACTs). In order to be able to gather as much Cherenkov light as possible, these telescopes are equipped of large reflectors and a pixelized camera to get a digital image of the captured event as it can be seen in Figure 1.3. Most of the times they belong to an array of telescopes that allows to achieve collection areas of the order of $\sim$km^2.

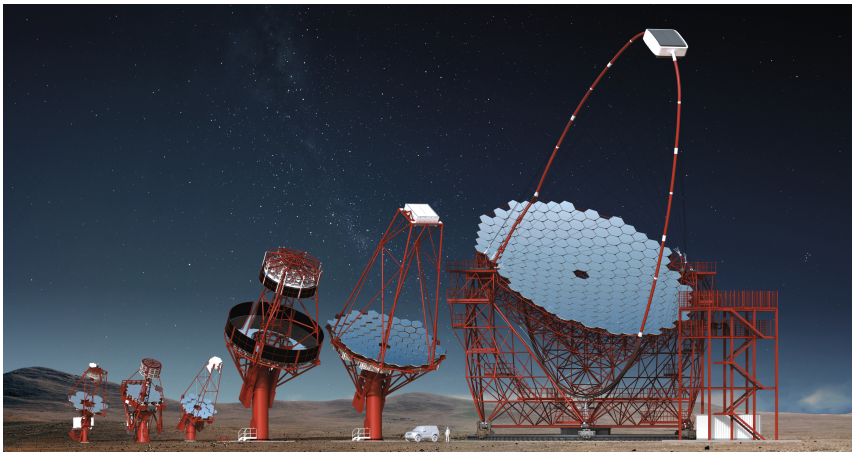

Figure 1.3: Rendering of different sizes and types of Imaging Atmospheric Cherenkov Telescopes. (Courtesy of CTAO.)

As they are sensitive to the Night Sky Background (NSB), they can only observe during dark time as shown in artist's Figure 1.4.

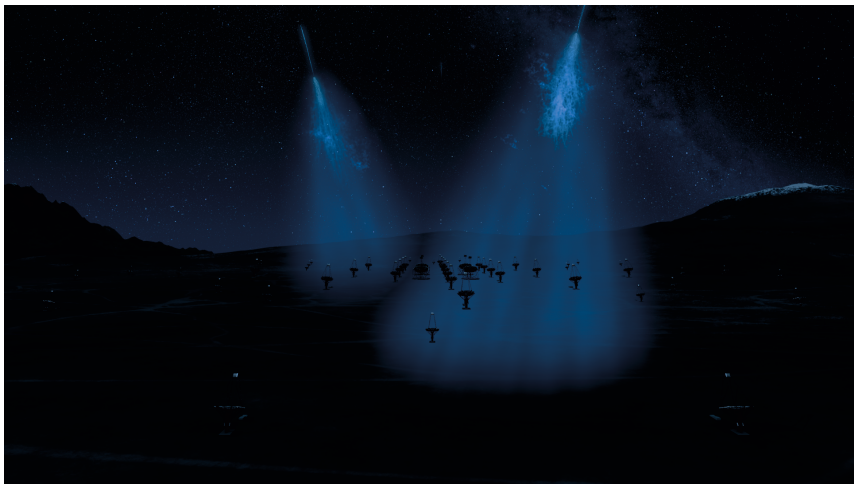

Figure 1.4: Artistic picture of the Cherenkov light during the night. (Courtesy of CTAO.)

The camera is not always recording what is captured by the telescope but it is triggered when a certain number of pixels are above a certain threshold. These pixels are usually fast, high-Quantum Efficiency (QE) Photomultiplier Tubes (PMTs).

Once the digital image of the event is captured, it can be used to classify the event type (hadronic or electromagnetic) and to estimate its direction and energy. Usually more than one telescope is used to capture the same event at the same time. Having multiple telescopes allows to proper reconstruct the direction of the incident γ ray and have a better background suppression.

The two types of showers (hadronic initiated and gamma initiated) produce different images on the camera, as shown in Figure 1.5 and in Figure 1.6 respectively. As general rule we can say that γ-ray showers produce elliptical shapes, while hadronic showers produce more irregular images. However, in many cases, the two images can be very similar and very difficult to be distinguished.

1.4 CHERENKOV TELESCOPE ARRAY (CTA)

The Cherenkov Telescope Array (CTA) (Acharya et al., 2013) is an international project aimed at building the next generation of ground-based Very High Energy (VHE) γ-ray instrument. The CTA will be located on two sites: one in the Southern Hemisphere (Cerro Paranal, Chile), shown in Figure 1.7 and another in the Northern Hemisphere (La Palma, Spain). It is proposed as an open observatory and it involves about 1400 scientists from 31 countries worldwide. The CTA has been designed in order to overtake the current generation of IACTs consisting of the Major Atmospheric Gamma-ray Imaging Cherenkov Telescope (MAGIC), the Very Energetic Radiation Imaging Telescope Array System (VERITAS) and the High Energy Stereoscopic System (H.E.S.S.).

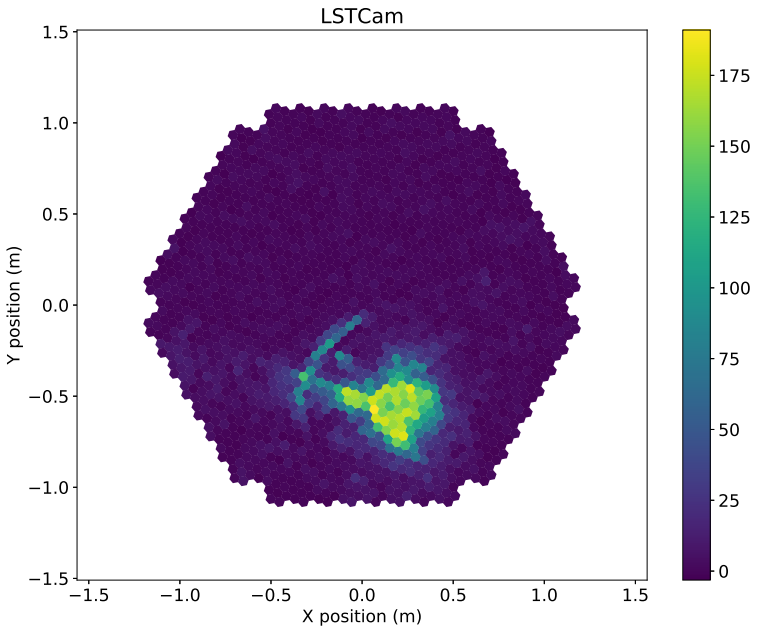

Figure 1.5: Event image produced by an hadronic initiated shower.

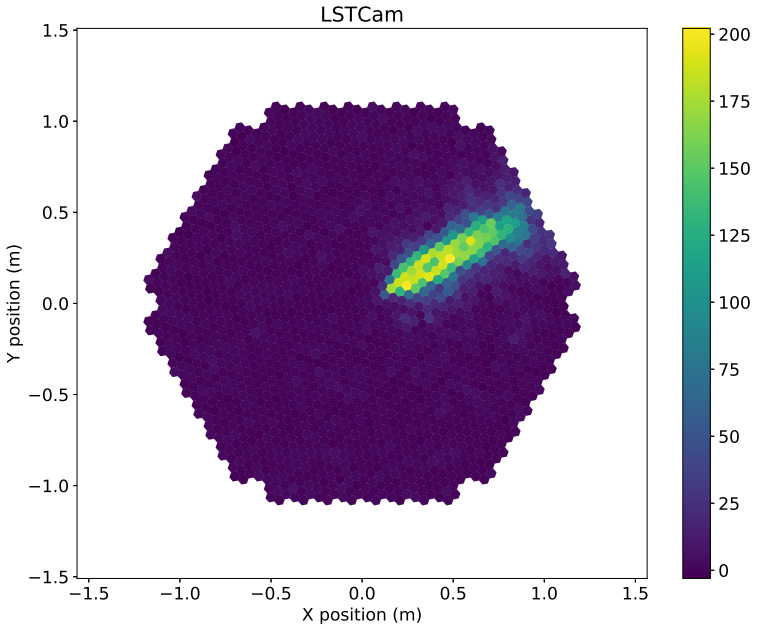

Figure 1.6: Event image produced by a γ-ray initiated shower.

In the array there will be three telescope types:

- Large Size Telescopes (LSTs) - These telescopes have a 23 m diameter reflector with a 27.8 m focal length. Their purpose will be to achieve the lowest possible energy threshold by an IACT.
- Medium Size Telescopes (MSTs) - They have 12 m reflectors with 17 m focal length. Mainly designed to improve the sensitivity at the medium energy band.

- Small Size Telescopes (SSTs) - Many of these telescopes will be placed only in the Southern observatory and they will be used only to capture γ rays at the highest energies. Their main purpose is to extend the collection area of the array.

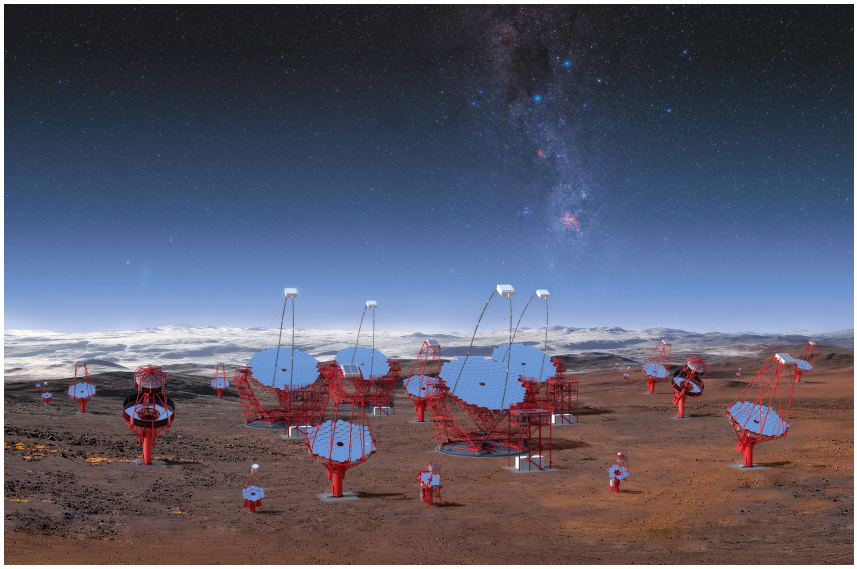

Figure 1.7: Rendering of the CTA array in the Southern Hemisphere. (Courtesy of CTAO.)

1.5 LARGE SIZE TELESCOPE

There will be at most four LSTs per site, designed to work in stereo in order to increase the collection area and have a more powerful γ/hadron separation. This improvement is more important at low energies where hadron generated showers and accidental triggers are more similar to γ-ray showers.

The LST is an alt-azimuth telescope. A structure made of reinforced carbon fibre and steel tubes supports a 23 m diameter parabolic reflective surface. This surface create a collecting area of 400 m^2 which reflects the Cherenkov light to the camera where the PMTs convert it in electrical signals, subsequently processed by the electronics. As already mentioned, the LST has been designed to cover the unique low energy sensitivity of CTA between 20 GeV and 150 GeV. The camera has a total Field of View (FoV) of about 4.3 degrees and a total number of channels of 1855 divided into 265 PMT modules that convert the light into electrical signals. Each module is formed by 7 Hamamatsu PMTs, to which are attached light guides in order to collect as much as possible light reflected by the mirror and to reject the NSB light that comes from larger angles. Each PMT signal is then preamplified and sent to the readout board and the triggering system, which determines whether trigger the telescope. The triggering strategy is based both on

the temporal evolution of the signal produced in the camera and the shower topology.

1.6 IACT DATA ANALYSIS

In this context, data analysis plays a fundamental role as images acquired by the telescopes have to be analyzed in order to extract meaningful informations. First of all the gamma-ray initiated showers are a very small fraction of the total events recorded by an IACT (the proportion is $< 1000 : 1$ in the regime where the LST will work) then it is crucial to develop an algorithm that is able to efficiently apply background rejection. The next steps require to accurately reconstruct the position of their source in the sky and the energy of the γ-ray events. This set of operations is known as *event reconstruction*.

All the analysis chains used so far rely on some kind of parametrization. The first primitives analysis were based only on the Hillas parameters (Hillas, 1985). The Hillas parameters are derived by fitting an ellipse to the pixels after an image cleaning process (Aharonian et al., 2006). These parameters extract several informations, some of them described below:

- **Hillas parameters**:
  - **Intensity**: also called *size*, is the total charge contained in the image.
  - **Width**: RMS spread along the minor axis of the ellipse. It measures the lateral development of the shower.
  - **Length**: RMS spread along the major axis of the ellipse. It measures the longitudinal development of the shower.
  - **CoG**: center of gravity of the image.
  - ϕ: angle between the line connecting the CoG with the center of the camera and the x-axis.
  - ψ: angle between major axis of the ellipse and and the x-axis.
  - **r**: angular distance between the CoG and the center of the camera.
  - **Skewness**: 3rd order Hillas parameter.
  - **Kurtosis**: 4th order Hillas parameter.

Some of these parameters are shown in Figure 1.8. Other parameters that can be extracted from an event image are:

- **Time parameters**:
  - **Time gradient**: slope of the linear fit to the arrival time projection along the major axis of the ellipse.

  - **Time intercept**: intercept on the ordinate axis of the linear fit to the arrival time projection along the major axis of the ellipse.

- **Image quality parameters**:
  - **LeakageN**: fraction of the size of the source contained in the N outermost rings of the camera.
  - **Number of islands**: number of non-connected pixels that survived the image cleaning.

As a typical signal in the camera has an elliptical shape, the arrival direction is reconstructed by tracing the main axis of the ellipse which corresponds to the projected direction of the shower in the camera FoV, to the γ-ray source in the sky.

Other two techniques that significantly improves the performances with respect to the Hillas analysis are the Model++ method (de Naurois and Rolland, 2009) and the ImPACT method (Parsons and Hinton, 2014). These algorithms utilize likelihood fitting of camera pixel amplitudes to semi-analytical shower models or templates filled with Monte-Carlo (MC) simulations.

Currently, the algorithm that achieves the best performances and that is nowadays the standard algorithm for IACT data analysis is the Random Forest (RF) (Albert et al., 2008). The RF exploits the Hillas parameters and other variables such as timing parameters or image quality parameters. It provides stable results and is robust with respect to input parameters, even if strongly correlated. There are only few parameters to tune and it is fast to train, test and use for inference. This is crucial, as IACTs produce a huge amount of data that has to be processed as fast as possible. For completeness, to train the RF and to derive the results in this thesis, I used the version implemented in *cta-lstchain*. All the parameters used are listed in Appendix A.2. It is also worth to mention that, since the analysis software for the LST is still under development, the variables for the RF input and the hyperparameters values of the RF are not fully optimized.

The first LST, known as LST1 and located in the Northern Hemisphere array, finalized its construction and was inaugurated in October 2018. It is currently undergoing the commissioning phase, that will be finalized in early 2020. Since the LST will be the only telescope of the array in operation for some time and due to the fact that there is no standard way to apply single telescope reconstruction, it is necessary to explore techniques that allows to efficiently exploit the telescope potential. Although the RF is very robust, it lacks of pixel-wise information that is lost during the image parameterization.

In this thesis I propose a Convolutional Neural Network (CNN) based analysis chain capable of single-telescope γ/hadron separation and event reconstruction for the LST. The key idea is to avoid every kind of parametrization on the raw images produced by the telescope

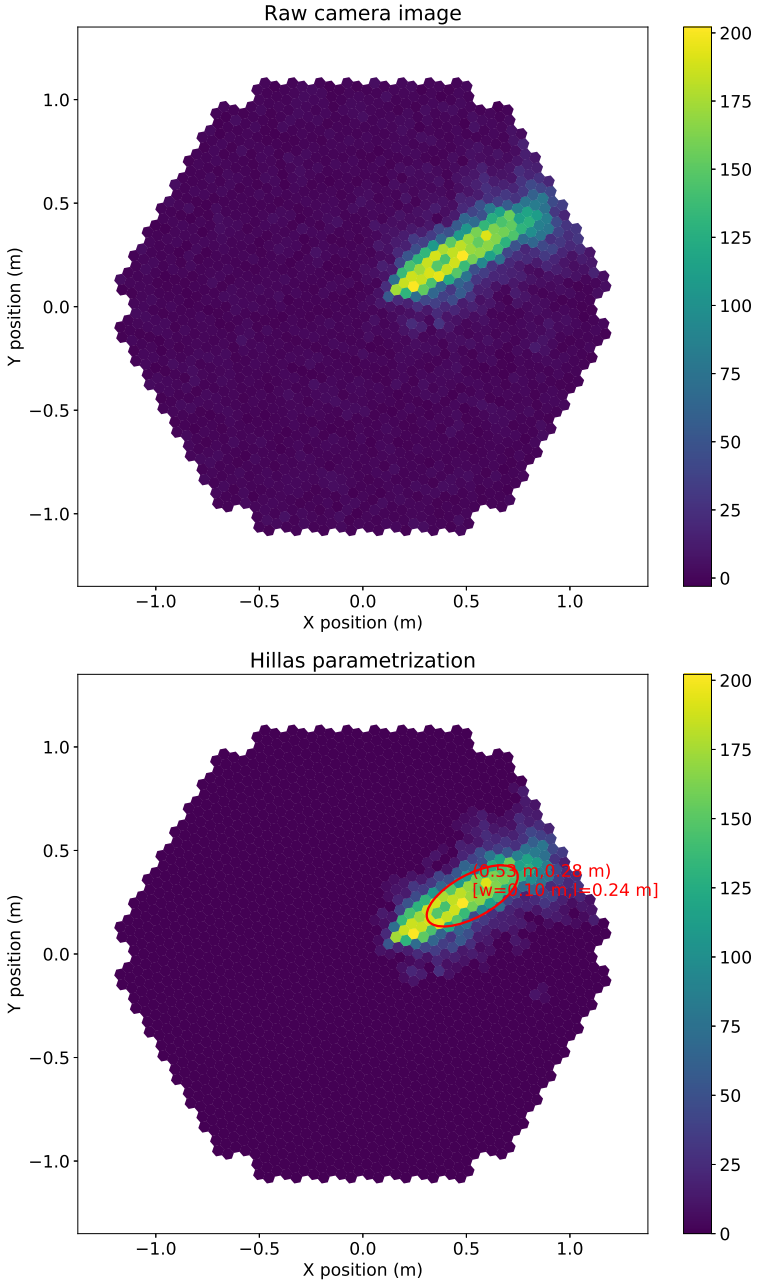

Figure 1.8: Top panel: raw camera image. Bottom panel: cleaned image and fitted ellipse with some Hillas parameters.

and letting the networks learning what are the relevant features useful to discriminate γ-initiated showers from hadronic showers and to compute the event properties, i. e. energy and direction. To perform full event reconstruction different dedicated CNNs have been trained for the three different tasks. I will show that this new analysis chain outperforms the current standard algorithm for separation and event reconstruction based on the RF and that it achieves new state-of-the-art performances.

2

DEEP LEARNING AND CONVOLUTIONAL NEURAL NETWORKS

Machine Learning (ML) is the subfield of Artificial Intelligence (AI) that focuses on teaching machines how to learn from data to perform a specific task. Until the introduction of the first ML algorithms, programmers had to explicitly define a set of rules in order to make a machine capable to solve a specific task. However this required a lot of efforts and specific domain knowledge. With the automated detection of patterns in data, a ML algorithm learns itself how to effectively perform a specific task from the experience. This approach allows machines to learn how to fulfill tasks that are relatively easy for a human being but very hard to be described by a finite set of instructions. On the other hand, ML also allows to perform tasks that would be too hard for a human being to be accomplished as, for instance, give the probability that an IACT event image was initiated by an electromagnetic primary particle instead of an hadronic primary particle.

One of the most promising branches of ML is Deep Learning (DL). DL is one way to solve *representation learning* problems i.e., it allows to extract useful informations from raw data and represent it in a hierarchical way. This hierarchical representation can then be used to perform classification and regression problems. DL has gained momentum during the last few years and nowadays is one of the most popular and prolific scientific research area in Machine Learning and Artificial Intelligence. The raise of DL has consistently increased in the last years thanks also to the huge improvements in computational power, especially due to the GPU usage for matrix operations.

2.1 CONVOLUTIONAL NEURAL NETWORKS

Among all, a very prominent and revolutionary DL tool, mainly applied in Computer Vision, are the Convolutional Neural Networks (CNNs) (LeCun et al., 1989). The CNNs are Artificial Neural Networks (ANNs) that perform the convolution operation multiple times on the input image and sample it progressively reducing the image size. Their novel approach has led to a series of breakthroughs, particularly for image classification when the AlexNet (Krizhevsky, Sutskever, and Hinton, 2012) significantly overperformed the existing algorithms on ImageNet. Since that moment CNNs started to become very popular. There are also examples of other tasks performed by CNNs such as object localization into an image (Liu et al., 2016), video interpolation (Jiang et al., 2018), speech recognition (Abdel-Hamid et al., 2014) and

many others. CNNs are composed of multiple layers that learn features with different levels of abstraction. Their advantage with respect to Multilayer Perceptron (MLP) networks is the lower computational complexity, thanks to local connections, shared weights and pooling operation. The building blocks of the modern CNNs include convolutional layers, pooling layers (Lecun et al., 1998), the ReLU activation function (Glorot, Bordes, and Bengio, 2011; Jarrett et al., 2009; Nair and Hinton, 2010) and the batch normalization (BN) (Ioffe and Szegedy, 2015). Usually the first layers of CNNs are convolutional and pooling layers, whereas last layers are ordinarily fully connected with the softmax or linear activation function in the very last layer, respectively for classification and regression problems. For binary classification problems the last layer can be composed by a single neuron with the sigmoid activation function.

A crucial feature of the CNNs is their depth. In Simonyan and Zisserman, 2015 the authors presented the Very Deep Convolutional Neural Network Architecture, also known as VGG Net which uses very small convolution filters and depth from 16 to 19 layers. Over the time many other solutions have been proposed in order to exploit more deeper networks such as (He et al., 2016; Krizhevsky, Sutskever, and Hinton, 2012) showing that deeper networks perform better.

2.2 ARCHITECTURES REVIEW

In this section we will briefly review the explored architectures for the analysis chain. We will discuss the key concepts and their strengths.

2.2.1 *VGG-style networks*

When using CNNs, a common base model to build is a VGG-style network (Simonyan and Zisserman, 2015). It consists of a stack of convolutional layers with very small filters (3×3), pooling layers, ReLU layers, and on top of it a fully connected section that ends the network.
The VGG-style base model that was employed is a 9-layers VGG-style network, whose architecture is represented in Table A.1.

2.2.2 *Deep residual convolutional networks*

A deep residual convolutional network is based on the concept of the *deep residual learning* framework. This framework was introduced by He et al., 2016 and it allows to train considerably deeper networks than before and to enjoy the accuracy gains from greatly increased depth. The idea behind residual learning is to consider $\mathcal{H}(\mathbf{x})$ as the mapping

function of a stack of few layers and then let the layers approximate the residual function

$$\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$$

by using a shortcut connection, as depicted in Figure 2.1.

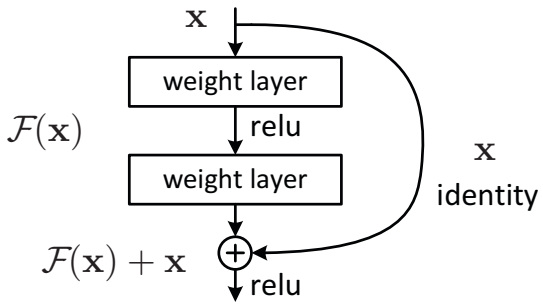

Figure 2.1: Residual learning: a building block (He et al., 2016).

This is motivated by the degradation problem, as multiple nonlinear layers have difficulties in approximating identity mappings. With the residual learning the solvers may drive the weights of the multiple nonlinear layers to zero to approximate identity mappings. The shortcut connections are used every stack of few layers and do not increase the number of parameters or the computational complexity.
The employed residual convolutional networks are 26-layer and 58-layer CNNs based on the *deep residual learning* framework, whose structure is shown in Table A.2 and in Table A.4 respectively.

2.2.3 *Densely connected convolutional networks*

A densely connected convolutional network is based on the concept of dense connectivity, introduced by Huang et al., 2017. Their envision is that it is possible to exploit the potential of the CNNs through *feature reuse* by concatenating feature-maps learnt by different layers. This approach yields condensed models that are highly parameter efficient. As shown in Figure 2.2, densely connected networks have blocks called *dense blocks* where the ℓ^{th} layer receives the feature-maps of all preceding layers, $\mathbf{x}_0, \ldots, \mathbf{x}_{\ell-1}$, as input:

$$\mathbf{x}_\ell = H_\ell\left([\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{\ell-1}]\right)$$

where $H_\ell(\cdot)$ refers to a non-linear transformation of the the ℓ^{th} layer and $[\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{\ell-1}]$ refers to the concatenation of the feature-maps produced in layers $0, \ldots, \ell - 1$.
Between dense blocks are placed the *transition layers* which do convolution an pooling, thus performing down-sampling. Moreover, a good effect of the more efficient parameters use in densely connected networks is to be less prone to overfitting. The model proposed, whose

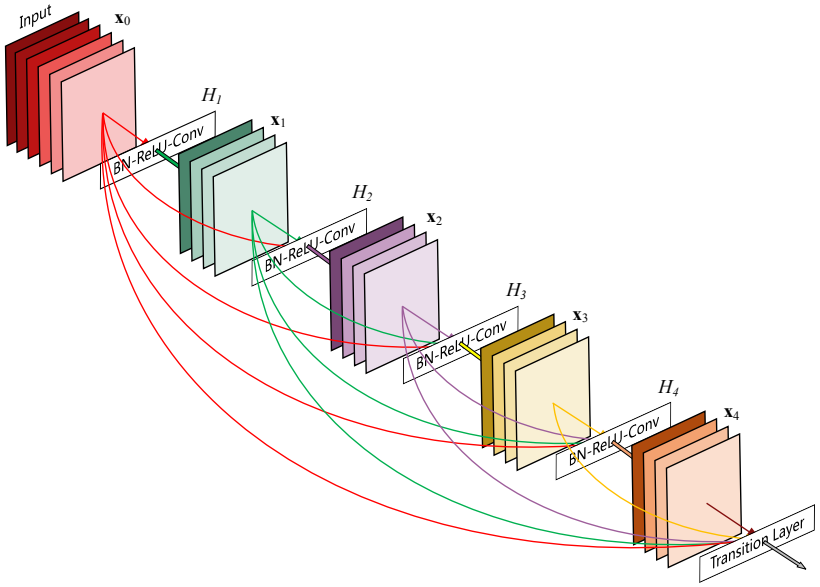

Figure 2.2: A 5-layer dense block with a growth rate of k = 4. Each layer takes all preceding feature-maps as input. (Huang et al., 2017).

architecture is shown in Table A.5 is a 64-layers densely connected convolutional network with growth rate $k = 12$ that makes use of the *bottleneck layers* and the *compression* technique, making it particularly parameter-efficient.

2.2.4 *Squeeze-and-Excitation networks*

A Squeeze-and-Excitation network (Hu, Shen, and Sun, 2018) is a CNN that makes use of the architectural unit called *Squeeze-and-Excitation (SE)* block. The SE block has been designed to explicitly model the channels interdependencies and thus to improve the quality of the representations produced by the network. This is achieved by a mechanism named *feature calibration* through which the network can excite relevant features and suppress irrelevant features. As it can be seen in Figure 2.3 a SE building block can be constructed for any given transformation $\mathbf{F}_{tr} : \mathbf{X} \mapsto \mathbf{U}, \mathbf{X} \in \mathbb{R}^{H' \times W' \times C'}, \mathbf{U} \in \mathbb{R}^{H \times W \times C}$ where H and W are the feature maps spatial dimension and C is the number of channels.

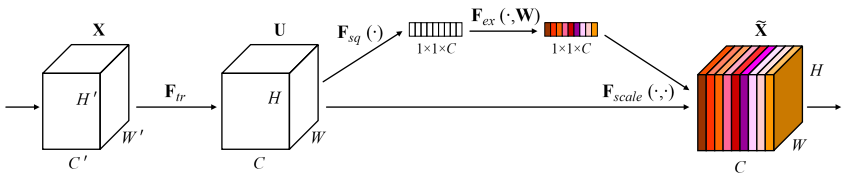

Figure 2.3: A Squeeze-and-Excitation block (Hu, Shen, and Sun, 2018).

The SE block performs first a *squeeze* operation $\mathbf{F}_{sq}$ which produces a channel descriptor followed by an *excitation* operation $\mathbf{F}_{ex}$ which produces a collection of per-channel modulation weights. At the end, the weights are multiplied with the corresponding feature maps of $\mathbf{U}$ and a new stack of feature maps with the same size is produced. This new stack can then be fed to the subsequent layers of the network. The main advantages of the SE block are that they can be directly implemented in the existing architectures, while enhancing the performances with a lightweight computation increment.
We applied the SE block to the ResNetF network A.2 and we have built its squeeze-and-excited counterpart ResNetFSE, whose structure is represented in Table A.3.

2.3 COMMON TECHNIQUES

When approaching a new problem that has to be solved using an ANN it is common to start with a baseline model, i.e. a simple model that allows to make sure that the whole training/testing pipeline is working correctly and to test the problem difficulty. A common simple baseline model is any VGG-style network. These are very easy to be built, since the only thing needed to build such a model is to stack convolutional, ReLU and pooling layers.

2.3.1 *Overfitting reduction*

An usual technique to address overfitting of large networks is *dropout* (Srivastava et al., 2014) that consists of randomly drop units (along with their connections) from the neural network during training. This allows to virtually train different architectures at the same time forcing the model to learn more robust features. Another regularization technique that helps to reduce overfitting is the *weight decay*, also known as the L^2 parameter norm penalty. In this case a penalty, called *regularizer* is added to the cost function. The penalty term has the form of $R(\mathbf{w}) = \lambda\|\mathbf{w}\|^2$ where λ is the penalty factor and $\mathbf{w}$ represents the layer weights.

2.3.2 *Training acceleration*

A fundamental method to improve the training speed of a neural network is batch normalization (BN) (Ioffe and Szegedy, 2015). It addresses the *internal covariate shift* by normalizing the layers input, i.e. subtracting the mean and dividing by the variance which are estimated during the training phase. It also acts as a regularizer and the authors claim that in some cases it eliminates the need for dropout.

The Stochastic Gradient Descent (SGD) is the most popular optimization algorithm for ML and DL. However there have been efforts

in trying to design optimization algorithms that provide fast convergence. Adam (Kingma and Ba, 2014) is one of them and, despite its controversials, if properly tuned, can be effectively in Computer Vision tasks.

2.4 DEEP LEARNING APPLIED TO IACT EVENT RECONSTRUCTION

The application of DL techniques to IACT data analysis is relatively recent. CNNs, Recurrent Neural Networks (RNNs) and MLP networks have been applied to IACT images in recent years in order to perform γ/hadron separation and event reconstruction. In (Feng, Lin, and VERITAS Collaboration, 2017) the authors showed for the first time the possibility to apply CNNs to IACT images to perform classification and regression tasks. In (Lukas Holch et al., 2017) they applied CNNs on H.E.S.S. stereoscopic simulated data using networks with $2-4$ convolutional layers, to perform γ/hadron separation, energy and direction reconstruction. In (Murach, Gajdus, and Parsons, 2015) the authors applied MLP networks to perform full single-telescope reconstruction on H.E.S.S. II events. nevertheless the MLP networks that they have used take as input the Hillas parameters extracted from the events, instead of taking as input the event images. In (Nieto et al., 2017) the authors applied state-of-the-art CNNs architectures to perform single-telescope background rejection on MST events. In (Shilon et al., 2019) they applied stereo background rejection by combining a CNN with a RNN and a CNN to perform direction reconstruction. Furthermore they pointed out the risks of using simulated data when developing DL-based analysis chains. However none of them apply state-of-art CNN architectures for full single-telescope reconstruction of LST events. It is also important to mention that most of the just presented studies, apply a cleaning process on the images before submit them to the CNNs, contrary to what is done here. The closest to this work is (Mangano et al., 2018), however it applies stereoscopic reconstruction and it takes into consideration only the events that trigger all the four LSTs in the array and a different events energy range. To the best of our knowledge this is the first time that CNNs are applied for full single-telescope reconstruction of LST events.

3

DATASET AND IMAGE RESAMPLING

In order to achieve the highest performances using ML algorithms, especially when DL models are employed, it is crucial to have access to a very large amount and good quality data. We can assume the quality of our dataset arbitrarily high since the events have been simulated by a simulator. On the other hand we have a sufficiently large dataset that is representative of the real-world.

The dataset was acquired from a set of MC data `DL1_ML1`, available in the CTA internal wiki and it is composed by many `HDF5` files. Each file contains a certain number of events representative of the entire dataset distribution. The files are divided into three event types: `point-like`, `diffuse` and `proton`. The `point-like` files contain events initiated by VHE γ rays coming from a source located at the center of the camera, `diffuse` files contain events initiated by VHE γ rays coming from sources randomly located on a cone with 10 degree radius with respect to the center of the camera and `proton` files contain events initiated by protons coming from sources randomly located on a cone of radius 10 degrees with respect to the center of the camera. The reason behind the division of the gammas events into two subsets is that `diffuse` are used to train the models, while `point-like` are used to test them. We train on `diffuse` because we do not want to bias the networks to the direction of the gammas but we want them learning the shape and infer their properties through it. Each event was simulated by a MC simulator, that simulates the entire physical event and the data acquisition of the telescope. The MC simulations reproduce the images captured by the telescope camera as a result of the interactions between hadrons and γ rays with the atmosphere. Finally, the events in the dataset are simulated taking into account the conditions of the Southern-hemisphere array at Paranal, Chile, which triggered at least one LST, impact parameter range 2500 m, pointing direction of the array *altitude*$= 70^\circ$, *azimuth*$= 0^\circ$, *intensity* > 50 and *leakage1* < 0.2. The total number of events for each event type is given in Table 3.1.

Event type	# of events	Energy range
`point-like`	$\sim 0.644 \times 10^6$	[0.003, 300] TeV
`diffuse`	$\sim 0.503 \times 10^6$	[0.003, 300] TeV
`proton`	$\sim 1.146 \times 10^6$	[0.005, 600] TeV

Table 3.1: Dataset for different event types.

The relevant informations for our purposes, contained in the dataset, are the intensity recorded by the pixels of the telescope camera, the peak time of each pixel, the event energy and the arrival direction of the γ ray. The peak times are essentially represented as another telescope camera image, where each pixel is associated with a peak time instead of a charge, as it can be seen in Figure 3.1. Given a specific camera pixel, the peak time represents the temporal offset between the moment in which the pixel recorded the energy peak and the triggering time of the telescope.

A key observation is that the dataset is not balanced in terms of energy distribution. There are energy bands with order of magnitude more statistics than others as it is clear from Figure 3.2. The reason for this unbalance is that the air shower simulation consumes many resources and the simulation is optimized to spend the same time simulating the same amount of energy in each decade of energy. This implies simulating two orders of magnitude less events in each subsequent decade in energy.

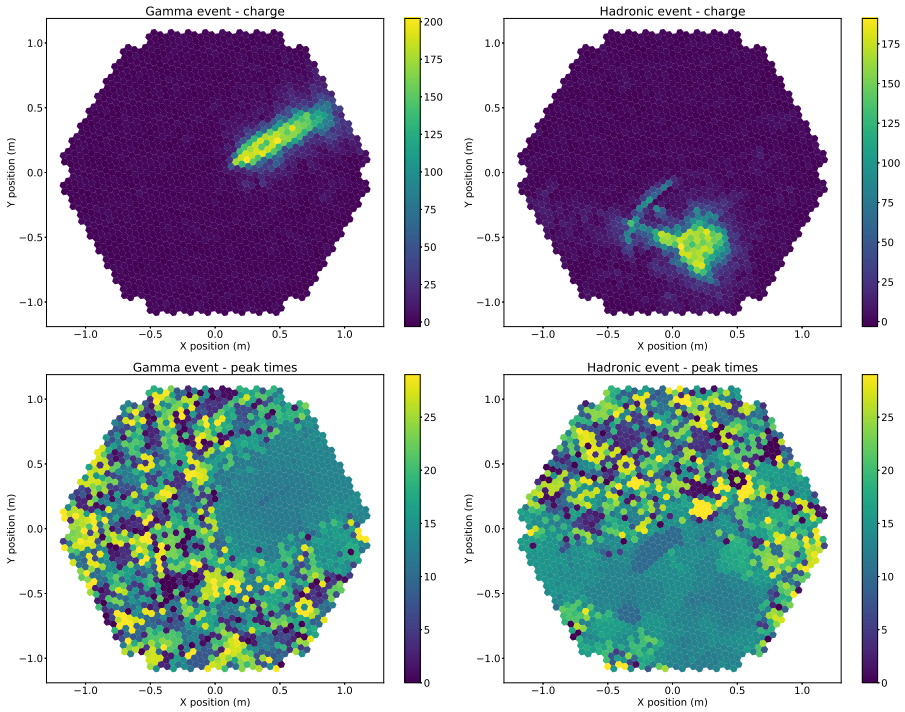

Figure 3.1: Dataset example images. On the left column an event from the `point-like` subset, on the right column an event from the `proton` subset. In the top row the image charges associated to the events are shown. In the bottom row the corresponding peak times images are represented.

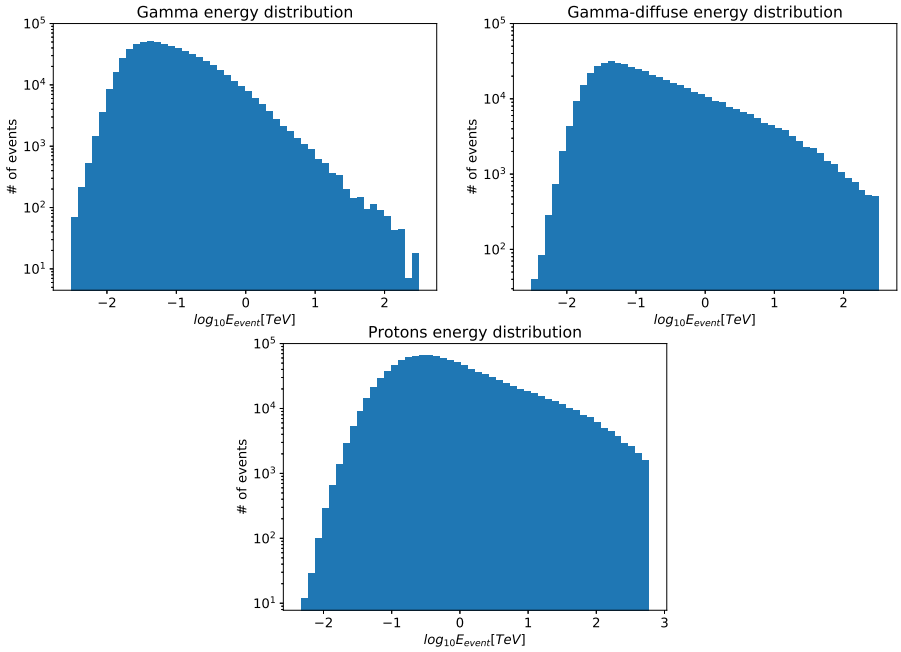

Figure 3.2: Energy distribution of the datasets used.

3.1 IMAGE PRE-PROCESSING

Since our implementation relies on the DL framework Keras (Chollet, 2015) and the backend Tensorflow (Abadi et al., 2015), it is necessary to resample the simulated telescope camera events. This is due to the fact that the telescope camera is arranged in an hexagonal grid and has an hexagonal shape, while the frameworks require square grid input images. The benefit of resampling the pixel values into a regular grid is that it allows to seamlessly apply the modern DL frameworks, however this operation inherently introduces distortion that can affect model performances, as the original camera image is not preserved. We selected the size of the grid such that each hexagonal pixel is not represented by more than two grid pixels, i. e. one grid pixel should be large half the distance between two hexagonal pixels center. Since the LST camera fits inside a square with side length equal to 2.5 m, the interpolated image has to be 100×100 pixels. Furthermore the final interpolated image, outside the camera area was padded with zeros.

In order to interpolate the camera images we consider them as a set of points whose coordinates refer to the centers of the hexagons. The new pixels, arranged in a regular grid, will be considered as discrete samples of a continuous function, computed by a bicubic interpolation function. Although the bicubic interpolation function is one of the most computational demanding resamplig function, we selected this one as the operation is applied just once to the entire dataset, while providing high accuracy, less artifacts and it excels at shape conservation (Lukas Holch et al., 2017). We can see in Figure

3.3 an example of an interpolated charge image and an interpolated time image.

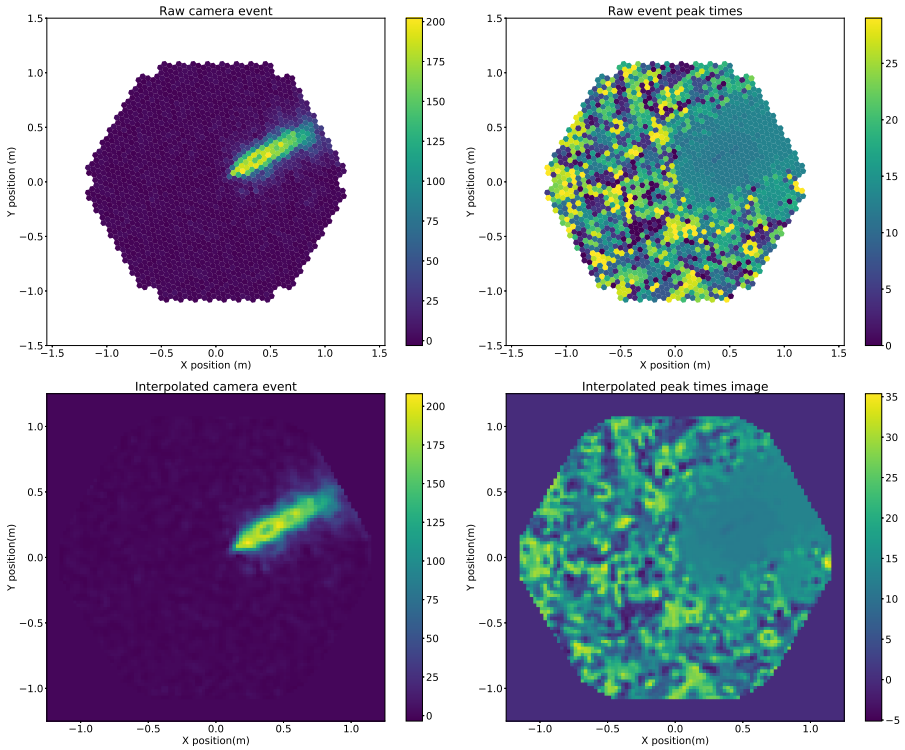

Figure 3.3: Interpolated example images. On the top row an event and its peak times from the `point-like` subset, on the bottom row their interpolated counterpart.

3.2 EVENT TENSOR COMPOSITION

A question that will be addressed later in the thesis is whether the peak times image can improve the model performances in any of the three tasks i.e. γ/hadron separation, energy or direction reconstruction. In order to feed a CNN with events that contain informations about the charge and the peak time of single PMTs, each input tensor needs to be composed of two channels. The first channel contains the interpolated charge image and the second channel the interpolated peak times image. Otherwise, if the peak times image is not provided to the neural network, the tensor will be composed by one channel, containing only the interpolated charge. Nevertheless, we will show that providing two-channel tensors to the neural networks, clearly improves the performances for all the three tasks. This will therefore be considered as the standard input tensor when exploring advanced architectures.

3.3 POTENTIAL DRAWBACKS

- A small caveat of this thesis is that although we have a dataset that resembles well enough reality, the results presented in this thesis will be as good as some of the characteristics of these simulations correspond to reality. Small differences in noise simulation, additional sources of real noise as stars in the FoV of the telescope, may be sources of a change in performance with respect to the one derived in this thesis.

- Since we are using a dataset produced to simulate the event acquisition of the entire array at Paranal, we are using events that trigger more than one telescope. This is not realistic as if the LST is the only telescope in operation the events are recorded when the LST is triggered.

- We are using k different images of the same event to train and test the models, where $k \leq 4$ is the number of LSTs which captured it. This is done to increase the dataset size, however it introduces a bias that, in some way, affects the models performances.

- The LST electronics has two gains to increase the dynamic range of the measurements. For the production of the dataset, specially prepared to perform CNN performance studies, only the high gain was used to produce it. As a result of this incorrect generation of the MC products, all signals with more than ~ 100 photoelectrons were saturated and the performances at the highest energies are compromised. This bug in the production of the dataset was found thanks to the work presented in this thesis and is currently being corrected for future productions.

3.4 DATA GENERATOR

Since the entire dataset does not fit in memory, it was necessary to write a Keras generator that is able to seamlessly load the minibatches from the disk and to manage the dataset during the training phase. The generator is responsible for indexing the dataset and shuffling the data at the end of each epoch, load the images and the corresponding labels (the energy and the arrival direction when training for energy and direction reconstruction) and return the minibatches to the optimizer. It also controls the minibatch size and the number of channels of the input tensor. Moreover, in order to hugely speed up the indexing phase, it makes use of multiprocessing, distributing the computation load across all the available CPU cores. We report, as an example, the generator used for the training of a background rejection model in Appendix A.1.

4

EVENT RECONSTRUCTION

When observing a candidate γ-ray source, in order to be confirmed, a significant number of γ rays coming from the same direction in the sky, need to be detected. This is due to the extremely low number of γ-ray events compared to hadronic events that trigger the telescope. Therefore a powerful background rejection and a very high angular resolution is demanded as the better the background rejection and the better the angular reconstruction, the more *significant* the signal is. Moreover accurately reconstruct the energy of the primary particle give a precise determination of the spectral parameters of the source.

In practice an IACT event analysis chain has to perform the three following tasks:

- *γ/hadron separation*: the background events need to be distinguished by the γ-initiated events. This is a binary classification problem where each event has to be labeled as background or signal or, more precisely, has to be labeled with the probability to be γ-initiated.
- *direction reconstruction*: each event comes from a specific direction in the sky that has to be reconstructed. This is a regression problem where, given the event tensor, a vector of two numbers, representing the sky direction of the γ ray is computed. The two numbers representing the sky direction are called *azimuth* and *altitude*.
- *energy reconstruction*: each event is associated to its primary particle's energy that has to be reconstructed. This is a regression problem where, given the event tensor, a scalar representing the energy of the primary particle is computed.

Normally, IACT data analysis is performed using the informations provided by more than one telescope. This type of analysis is also called *stereo reconstruction*. However the scope of this thesis is applying DL techniques to perform full event reconstruction, exploiting only the data produced by one LST of CTA i. e., *single-telescope reconstruction*. This makes the event reconstruction extremely challenging as there is a lack of informations from the other telescopes that are helpful, especially when reconstructing the arrival direction. We tested different CNN architectures and we trained the models to perform only one specific task. This makes easier to devise models with higher performances at the cost of an increased analysis chain computational complexity.

4.1 γ/HADRON SEPARATION

The first operation that has to be accomplished when analysing IACT events is the γ/hadron separation. This operation consists on discriminating between events that are γ-initiated and hadronic-initiated. In order to train a model on a training set to perform γ/hadron separation, we need to define a loss function to be optimized. Given the training set $S = ((\mathbf{x}_1, y_1) \dots (\mathbf{x}_{N^T}, y_{N^T}))$ of N^T events, where $\mathbf{x}_i$ represents an event tensor and $y_i \in \{0, 1\}$ represents its label, the label 0 is used in case of an hadronic event and 1 in case of a γ event. We define the loss function, called *binary crossentropy*, that has to be minimized during the training process, as:

$$\text{BCE} = -\frac{1}{N^T} \sum_{i=1}^{N^T} y_i \cdot \log\left(p\left(y_i\right)\right) + \left(1 - y_i\right) \cdot \log\left(1 - p\left(y_i\right)\right) \quad (4.1)$$

where $p(y_i)$ is the output of the model, i. e. the probability for an event to be γ-initiated, also called score or, in the context of IACT data analysis, *gammaness*. To evaluate the models rejection power and to compare them we need to define some performance metrics. We can start by defining the widely used performance metric *accuracy* as:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.2)$$

where TP, TN, FP and FN are respectively the *true positive*, *true negative*, *false positive* and *false negative* predictions. In practice this metric provides the proportion of correct predictions on the test set.

Typically in ML, when dealing with a binary classification problem, to compute the accuracy, the scores are approximated to the nearest integer between 0 and 1. What we want to do instead, is to optimize the signal-to-noise ratio to get the maximum significance of the signal we are detecting. For this reason the accuracy is not really representative of the real model performances. As we are interested to investigate the models performances across different gammaness thresholds we need a metric, giving an overview of the performances for different thresholds. The receiver operating characteristic (ROC) curve is visually the best way to do it. Moreover it is also possible to compute a single number, called area under the curve (AUC), computed as the normalized integral below the curve. Let us define the quantity *recall* or true positive rate (TPR) $F_1(t)$ as the proportion of positive cases classified as positive and the quantity false positive rate (FPR) $F_0(t)$ as the proportion of negative cases classified as positive, where t represents the gammaness. The ROC curve is then a plot of $F_1(t)$ against $F_0(t)$.

Setting $v = F_0(t)$, we can formally define the area under the ROC curve as:

$$\text{AUC} = \int_0^1 F_1\left(F_0^{-1}(v)\right) dv \tag{4.3}$$

With such a definition the perfect separation occurs when AUC $= 1$ and a random separation when the ROC curve degenerates into a straight line passing through the origin, with angular coefficient equal to 1.

4.1.1 *Models performances*

To perform background rejection, we explored different handcrafted architectures based on different frameworks (i.e. residual framework, densely connected networks, etc.). We tried to derive the best architecture per framework and we evaluated the performances on the test set. The dataset includes $N \simeq 0.807 \times 10^6$ `diffuse` and `proton` events, that have been splitted with proportion 80%-20% in training set and validation set respectively, which is used to keep overfitting under control, while the test set contains $M \simeq 1.291 \times 10^6$ `point-like` and `proton` events. For each training we fixed a budget of 50 epochs to fairly compare the models and due to time constraints, while the model tested on the test set is the model with the highest validation accuracy. Each model has been trained using the Adam optimizer, with $lr = 0.001$ and batch size 128 (64 for the DenseNet as a larger batch size does not fit into GPU memory), which has already shown its good generalization properties for this task (Nieto et al., 2017). Moreover it has been used a reduce lr on plateau policy to decrease the lr. We tested a simple VGG-style network, a densely connected network, a residual network and its squeeze-and-excited counterpart. All the models make use of the BN, the VGG-style models are regularized with dropout while ResNets and the DenseNet with weight decay. In Figure 4.1 we represent the accuracy progression during the training. As we can see, all the models start to overfit at a certain point during the training and the validation accuracy saturates. In Figure 4.2, we show the ROC curves obtained by testing the models on the test set. The primary important result is that all the models significantly outperforms the RF in terms of accuracy and AUC score, providing a much better background rejection power across all the thresholds. These results show once again the effectiveness of CNNs in performing image classification problems, which still holds for γ/hadron separation. The ResNetF and the DenseNet have practically the same performances, yet the latter has an higher computational complexity. The ResNetFSE has taken advantage of the SE block, providing the best performances while keeping a lower computational complexity than the DenseNet. It improved the accuracy by 14.5% and the AUC

score by 13.5% with respect to the RF. In Figure 4.3 we report the ROC curves produced by the ResNetFSE on the test set for events of different energy bands, while in Figure 4.4 we report the gammaness distribution produced by the model on the test set. As clear from Figure 4.3 the model rejection power increases as the event energy range increases, refolding at the highests. This could be due to the gain-related dataset issue mentioned in Section 3.3. A performances summary is presented in Table 4.1.

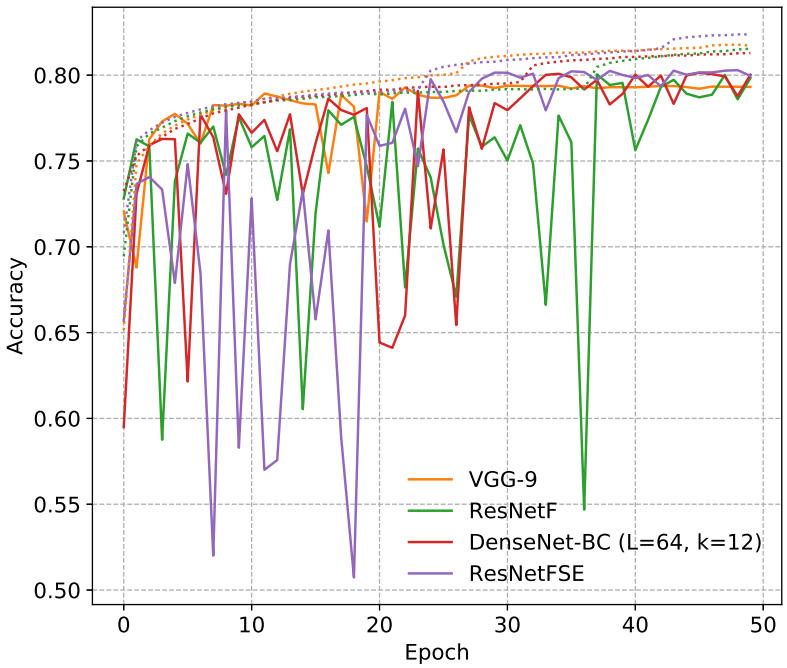

Figure 4.1: Training accuracy (dotted) and validation accuracy (solid) of the different models during the training.

4.1.2 *Peak times channel impact on performances*

As it can be seen in Figure 4.2, we investigate whether the CNNs can gain rejection power when including the peak times in the event tensor. We compared the two situations using the VGG-9 baseline model, that has been trained using the same dataset in one case with one-channel input tensors and in the other case two-channel input tensors. While intuitively, the peak times should not bring a great advantage, they actually helps to better separate the events. Indeed the model trained without the peak times image obtained an accuracy of 0.770 and an AUC score of 0.863 on the test set, while the model trained with the peak times obtained an accuracy of 0.788 and an AUC score of 0.885. This means that the peak times image contains informations that the CNNs are able to exploit, to obtain a much higher accuracy. In conclusion, as the model that has access to the peak times image

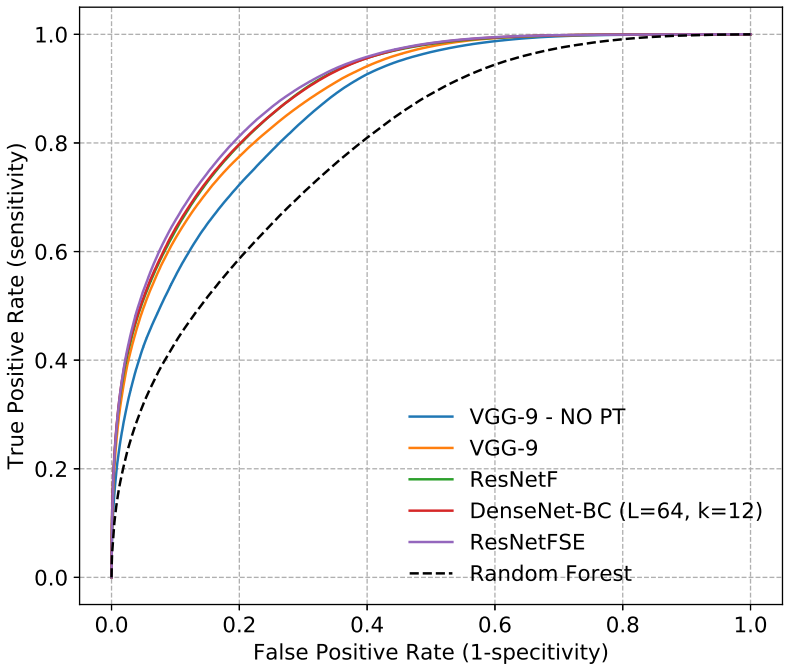

Figure 4.2: ROC curves obtained by testing the models on the test set.

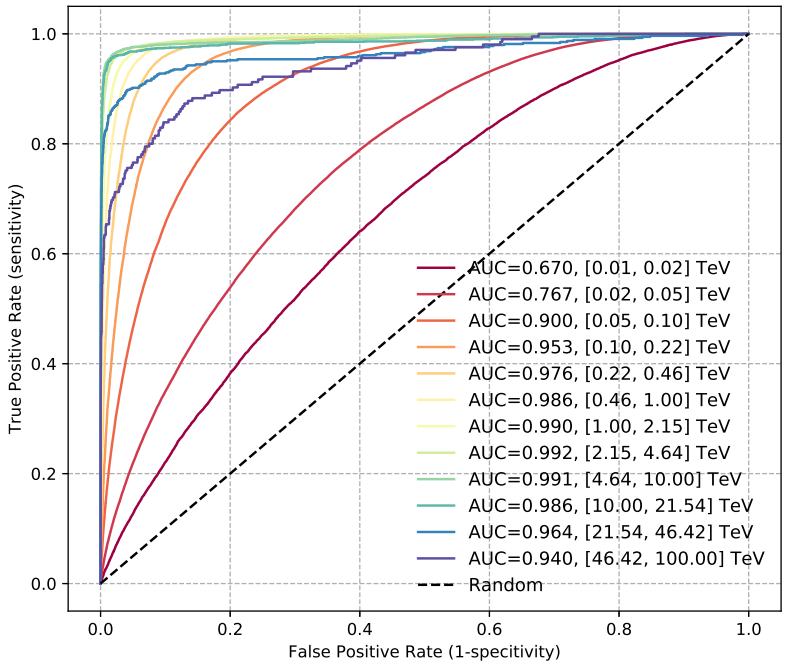

Figure 4.3: ROC curves obtained by testing the ResNetFSE on the test set, for different energy bins.

obtained higher scores, the CNNs can gain rejection power when the peak times are included in the input tensor. We can also notice that the model trained without the additional time channel is still more

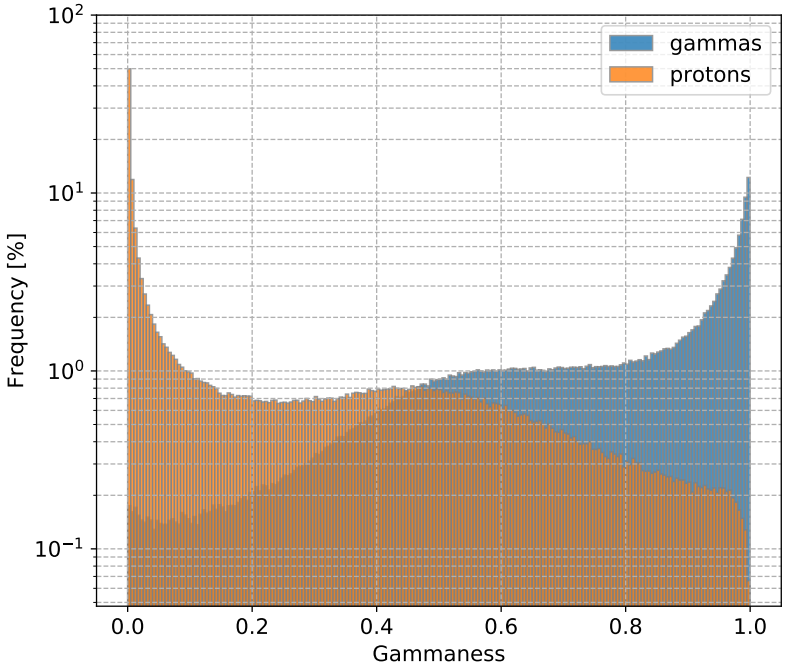

Figure 4.4: Gammaness distribution produced by the ResNetFSE on the test set.

powerful than the RF, that has access to the parameters derived from peak time informations.

Model	PT	Accuracy	AUC
Random Forest A.2	yes	0.705	0.793
VGG-9 A.1	no	0.770	0.863
VGG-9 A.1	yes	0.788	0.885
ResNetF A.2	yes	0.801	0.894
DenseNet-BC $(L = 64, k = 12)$ A.5	yes	0.800	0.895
ResNetFSE A.3	yes	**0.807**	**0.900**

Table 4.1: Summary results for separation on the test set. PT denotes if the model has access to the peak time information (i.e. the time gradient for the RF and the additional tensor dimension for the CNNs).

4.2 DIRECTION RECONSTRUCTION

The operation of event direction reconstruction consists in retrieving, using the event image, the *azimuth* and the *altitude* of the point in the sky where the γ-ray source that produced the γ ray is located. However we will not directly estimate the azimuth and the altitude, but the difference in azimuth and in altitude between the source and the pointing direction of the telescope, called Δ_{az} and Δ_{alt} respectively.

The two numbers can then be used to compute the azimuth and the altitude of the γ-ray source. In order to estimate the arrival direction reconstruction precision, we introduce the quantity *angular resolution* θ_{68}, measured in degrees. One, to compute the angular resolution for a given energy bin, has to consider the simulated and reconstructed events inside the energy bin and compute the following quantity:

$$\theta^2 = (\Delta_{az} - \hat{\Delta_{az}})^2 + (\Delta_{alt} - \hat{\Delta_{alt}})^2 \tag{4.4}$$

where Δ_{az} and Δ_{alt} are the difference between the azimuth and the altitude of the pointing direction of the telescope and the azimuth and the altitude of the source, while $\hat{\Delta_{az}}$ and $\hat{\Delta_{alt}}$ are their reconstructed counterpart. Then, it has to be taken the root square of the 68% containment of the histogram of θ^2.

In order to train a model on a training set to perform the direction reconstruction and to evaluate its performances, we need to define a loss function. Given the training set $S = \left((\mathbf{x}_1, \mathbf{d}_1) \dots \left(\mathbf{x}_{N_\gamma^T}, \mathbf{d}_{N_\gamma^T}\right)\right)$ of N_γ^T γ-events where $\mathbf{x}_i$ represents an event tensor and $\mathbf{d}_i$ represents the vector containing Δ_{az} and Δ_{alt}, so that $\mathbf{d}_i = [\Delta_{az}, \Delta_{alt}]^T$, we can define the loss function, called *mean absolute error*, that has to be minimized during the training process, as:

$$\text{MAE}_d = \frac{1}{N_\gamma^T} \sum_{i=1}^{N_\gamma^T} \|\mathbf{d}_i - \hat{\mathbf{d}}_i\| \tag{4.5}$$

where $\hat{\mathbf{d}}_i$ is the reconstructed counterpart of $\mathbf{d}_i$.
This loss function is well-suited for distributions that are exponentially decaying. In principle, for a gaussian distribution, as the θ^2 distribution is expected to behave, the MSE [1] should behave better. However, we found that the MAE_d is the one giving the best performance, so the distribution may be deviating from our expectations.

4.2.1 *Models performances*

To perform the direction reconstruction task, we explored different handcrafted architectures based on different frameworks (i.e. residual framework, densely connected networks, etc.). The dataset employed for direction reconstruction includes $N_\gamma \simeq 0.504 \times 10^6$ `diffuse` events, splitted with proportion 80%-20% in training set and validation set respectively which is used to keep overfitting under control, while the test set contains $M_\gamma \simeq 0.645 \times 10^6$ `point-like` events. For each training we fixed a budget of 50 epochs to fairly compare the models and due to time constraints, while the model tested on the test set is

1 $\text{MSE}_d = \frac{1}{N_\gamma^T} \sum_{i=1}^{N_\gamma^T} \|\mathbf{d}_i - \hat{\mathbf{d}}_i\|^2$

the one with the lowest validation MAE_d. Each model has been trained using the Adam optimizer, with $lr = 0.001$, batch size 128 (64 for the DenseNet as a larger batch size does not fit into GPU memory) and a reduce lr on plateau policy to decrease the lr. We tested a simple VGG-style network, a densely connected network, a residual network and a squeeze-and-excited residual network. All the models make use of the BN, the VGG-style models are regularized with dropout while ResNets and the DenseNet with weight decay. In Figure 4.5 is represented the loss progress during the training. As it can be seen the VGG-9 and the ResNetH start to overfit during the training, while the DenseNet and the ResNetFSE still fit well the data at the end of the 50 epochs. For this reason it is likely there is still room to improve their performances with longer trainings. In Figure 4.6 we represent the final angular resolution curves, obtained by computing the quantity θ_{68} on the test set for each energy bin. All the models perform better than the RF [2] across the entire energy spectrum and substantially increase the angular resolution. They have very similar performances and the VGG-9 already delivers performances comparable to the other models, or even slightly better when compared with the DenseNet, while offering a much lower computational complexity. It overtakes the RF performances up to $\sim 182\%$. The ResNetH, which is the network with the highest number of parameters, has practically the same performances of the aforementioned ones. As expected the models gain angular resolution as the energy increases, nevertheless they visibly lose it at high energies. This is due to the scarce amount of events to estimate the points and due to the gain-related dataset issue mentioned in Section 3.3. The model that provides the highest improvement across the energy spectrum with respect to the RF, is the ResNetFSE, indeed it overtakes the angular resolution up to $\sim 271\%$. This demonstrates the effectiveness of the SE block even for regression tasks. A performances summary is presented in Table 4.2. The lowest MAE_d on the test set has been obtained by the VGG-9 model. This is due to a low number of outliers, as the model is simpler and tends to produce less dispersive predictions. However, in principle, the quantity that we want to minimize is θ_{68}.

4.2.2 *Peak times channel impact on performances*

As for the background rejection task in Section 4.1.2, we investigate whether the the CNNs can gain accuracy in terms of direction estimation when including the peak times in the event image. We expect

2 We also performed the direction reconstruction using the RF existing on *cta-lstchain*, but the results were worst than those reached in other reported performances. To make a fair comparison of the RF performances for the direction reconstruction, we took the best results for this task obtained by the cta-lstchain implemented RF and applying no cuts to the dataset as we do in the one reconstructed in this thesis using CNNs.

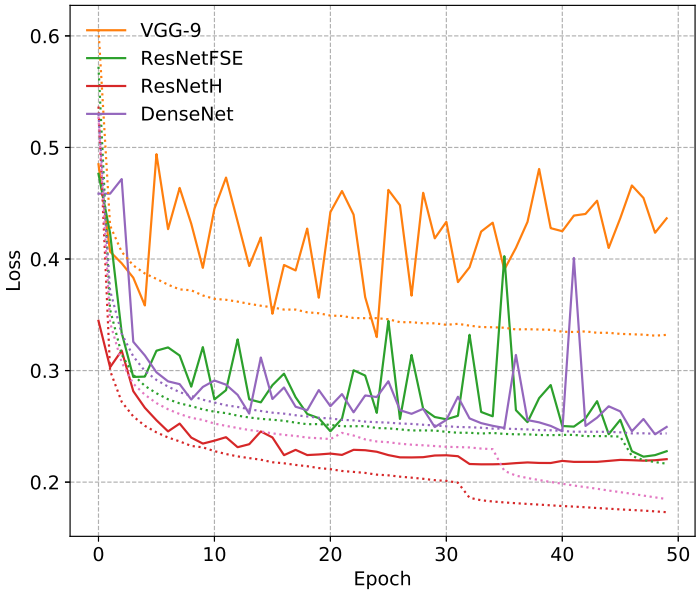

Figure 4.5: Training loss (dotted) and validation loss (solid) during the training.

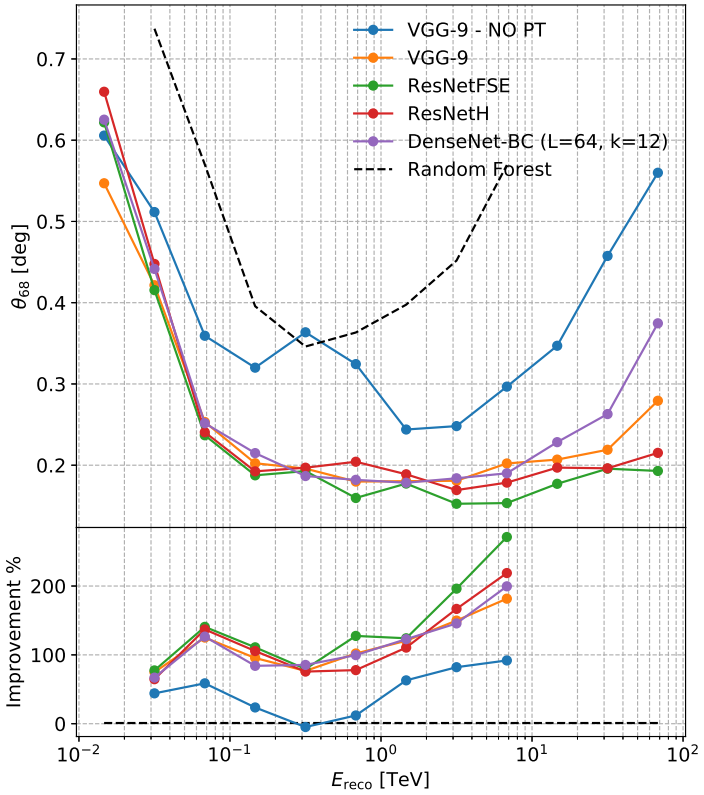

Figure 4.6: Top panel: angular resolution across the energy spectrum for each model. Bottom panel: resolution improvement with respect to the RF for each model.

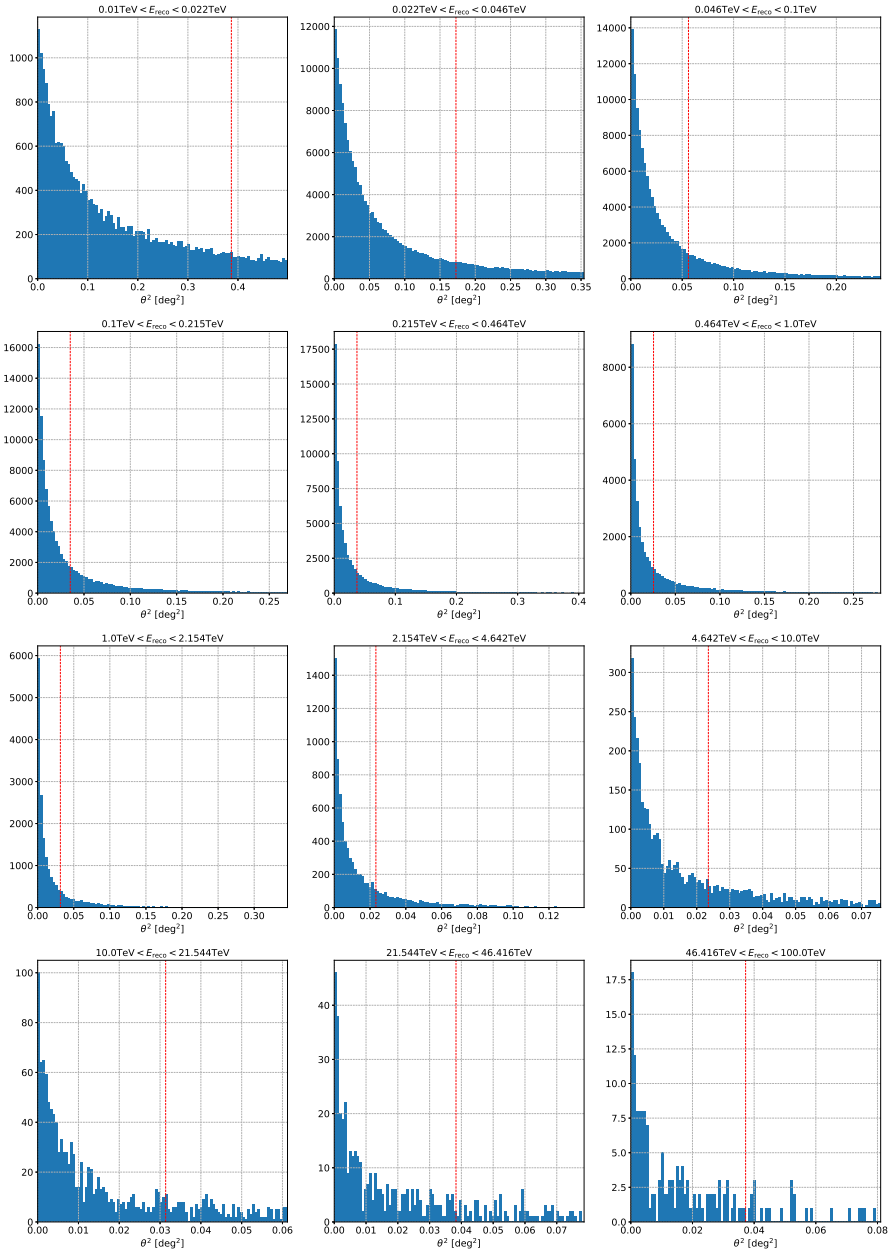

Figure 4.7: θ^2 for each energy bin. The red line represents the 68% containment of the histogram.

that the peak times hugely improves direction estimation as the peak time associated to each pixel inherently brings informations about the arriving direction. We take as example the VGG-9 baseline model that has been trained using the same dataset in one case with one-channel input tensors and in the other case with two-channel input tensors. The model trained without the peak times image obtained a MAE_d of 0.240 on the test set, while the model trained with the peak times obtained a MAE_d of 0.169, with an improvement up to $\sim 86\%$ in the angular resolution at middle energies. We can see from Figure 4.6 that, as expected, the difference between the two situations is very large

across the entire energy spectrum. This means that the CNNs are able to exploit the informations carried by the peak times to significantly increase the angular resolution.

Model	PT	MAE_d
Random Forest A.2	yes	-
VGG-9 A.1	no	0.240
ResNetH A.4	yes	0.181
DenseNet A.5	yes	0.180
ResNetFSE A.3	yes	0.171
VGG-9 A.1	yes	**0.169**

Table 4.2: Summary results for direction estimation on the test set. PT denotes if the model has access to the peak time information (i.e. the time gradient for the RF and the additional tensor dimension for the CNNs).

4.3 ENERGY RECONSTRUCTION

The operation of event energy reconstruction consists in retrieving the energy E_{gammas} of the primary particle that interacted with the atmosphere and produced the event. In order to train a model on a training set to perform the energy estimation, we need to define a loss function that has to be optimized during the training. Given the training set $S = \left((\mathbf{x}_1, e_1) \dots \left(\mathbf{x}_{N_\gamma^T}, e_{N_\gamma^T}\right)\right)$ of N_γ^T γ events where $\mathbf{x}_i$ represents an event tensor and e_i represents the logarithm of its primary particle's energy i. e. $e_i = \log_{10}(E_{\text{gammas}})$, we can define the loss function, called *mean absolute error*, that has to be minimized during the training process, as:

$$\text{MAE}_e = \frac{1}{N_\gamma^T} \sum_{i=1}^{N_\gamma^T} |e_i - \hat{e}_i| \tag{4.6}$$

where $\hat{e}_i$ is the reconstructed energy logarithm of the i-th event. With such a definition, the perfect reconstruction occurs when $\text{MAE}_e = 0$. The MAE_e is used also as metric function to compare the models performances on the test set. It is particularly well-suited for this task as, at the same time, it is convenient to optimize and it provides the same percentage error on different energy bands.

Although the MAE_e provides an objective way to compare the models performances we need something more functional to evaluate

how well the models perform. To this aim we define the *relative energy error* as:

$$E_{\text{err}} = \frac{E_{\text{gammas}} - E_{\text{reco}}}{E_{\text{gammas}}} \tag{4.7}$$

where E_{rec} is the reconstructed event energy. Afterwards we divide the energy spectrum in bins and, for each of them, we fit a gaussian to the distribution of the relative energy errors of each bin. The *energy bias* of the system is the μ of that gaussian. The *energy resolution* is defined as the σ of the gaussian.

4.3.1 *Models performances*

We explored different handcrafted architectures based on different frameworks (i.e. residual framework, densely connected networks etc.). We tried to derive the best architecture per framework and we evaluated the performances on the test set. The dataset includes $N_\gamma \simeq 0.504 \times 10^6$ `diffuse` events, that have been splitted with proportion 80%-20% in training set and validation set respectively, which is used to keep overfitting under control, while the test set contains $M_\gamma \simeq 0.645 \times 10^6$ `point-like` events. For each training we fixed a budget of 50 epochs to fairly compare the models and due to time constraints, while the model tested on the test set is the one with the lowest validation MAE_e. Each model has been trained using the Adam optimizer, with $lr = 0.001$ and it has been used a reduce lr on plateau policy to decrease the lr. All the models make use of the BN, VGG-style models are regularized with dropout while ResNets and the DenseNet with weight decay. In Figure 4.8 we can see the training and validation loss progress during the training of each model. In Figure 4.9 we show the bias curves, while in Figure 4.10, we show the resolution curves and their improvement curves with respect to the RF. The bias and resolution curves are derived from the fitted gaussians of the relative energy errors. As we can see, for the entire energy spectrum the models achieve a better energy resolution than the RF except for the VGG-9 - NO PT that was not trained to analyze the peak times. However this model has still better performances at low energies, while performing worst at the highests. For what concern the other models trained with the peak times, we can say that they perform much better than the RF across the entire energy spectrum, although at the highest energies the resolution drops. This could be due to the gain-related dataset issue mentioned in Section 3.3, due to the scarce amount of events to estimate the points but also due to an increasing fraction of truncated ellipses in the images. It is also worth to notice that the models perform better at the low-medium energy range, where the event images have an higher quality and much more light. While all the models are able to provide a very similar energy

resolution, by using more complex models we get at the same time a lower bias and thus a lower MAE_e.

The model that improved most the energy resolution is the ResNetFSE, with an enhancement up to $\sim 161\%$ at medium energies and an improvement of the MAE_e of 45.33%. In Figure 4.12 we show the relative energy error histograms for different energy bins, computed with the prediction of the ResNetFSE on the test set while in Figure 4.11 we represent the 2D histogram, again for the predictions of the ResNetFSE.

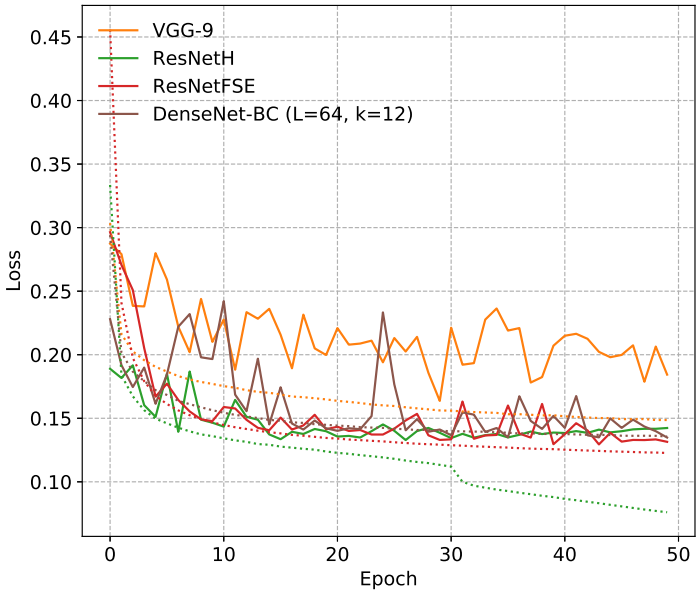

Figure 4.8: Training loss (dotted) and validation loss (solid) during the training.

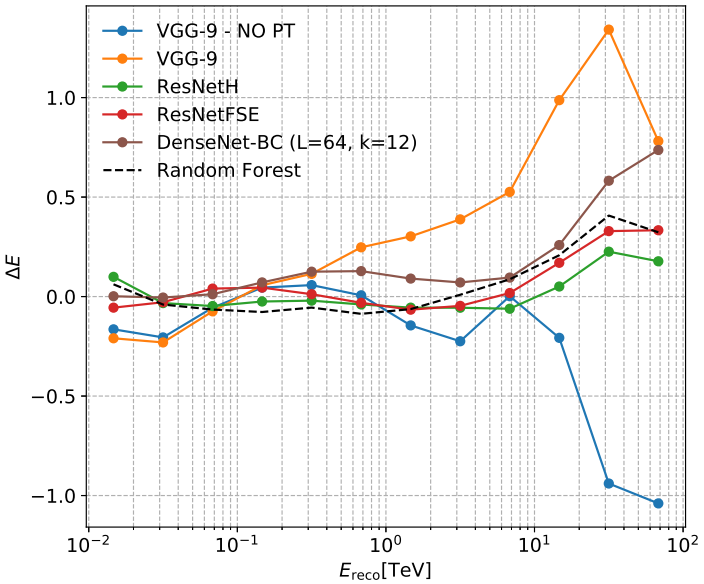

Figure 4.9: Energy biases across the energy spectrum for each model.

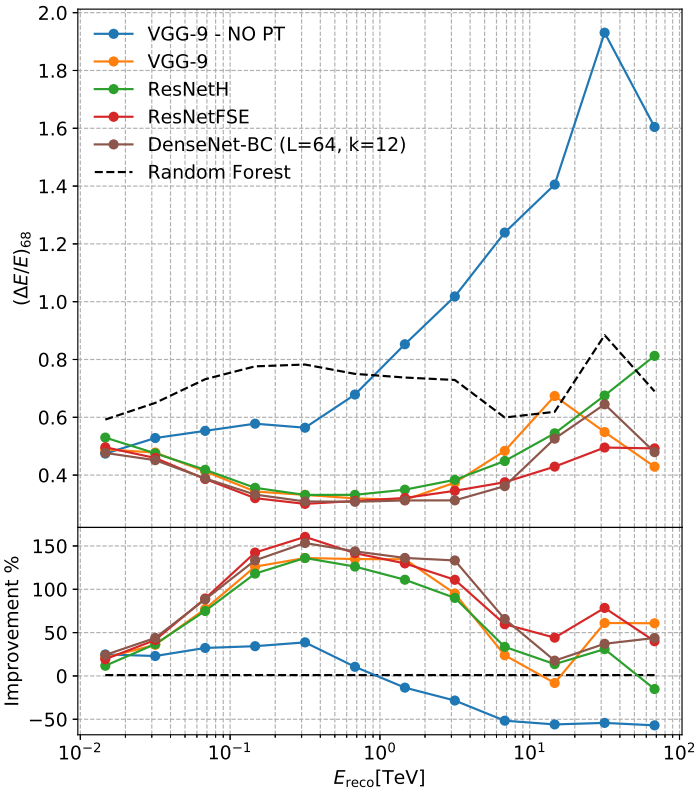

Figure 4.10: Top panel: energy resolution across the energy spectrum for each model. Bottom panel: resolution improvement with respect to the RF for each model.

4.3.2 *Peak times channel impact on performances*

As for the background rejection task in Section 4.1.2 and for the incoming direction reconstruction in Section 4.2.2, we investigate whether the CNNs can gain accuracy in terms of energy estimation when including the peak times in the event image. We have taken as example the VGG-style baseline model that has been trained using the same dataset, in one case with one-channel input tensors and in the other case with two-channel input tensors. The model trained without the peak times image obtained a MAE_e of 0.191 on the test set, the model trained with the peak times obtained a MAE_e of 0.141. We can see in Figure 4.9 that the two models have similar performances for the bias, however, as shown in Figure 4.10, the model that has access to the peak time information has a much better resolution across the entire energy spectrum. This means that the peak times image contains informations that the CNNs are able to exploit to heavily increase the energy resolution.

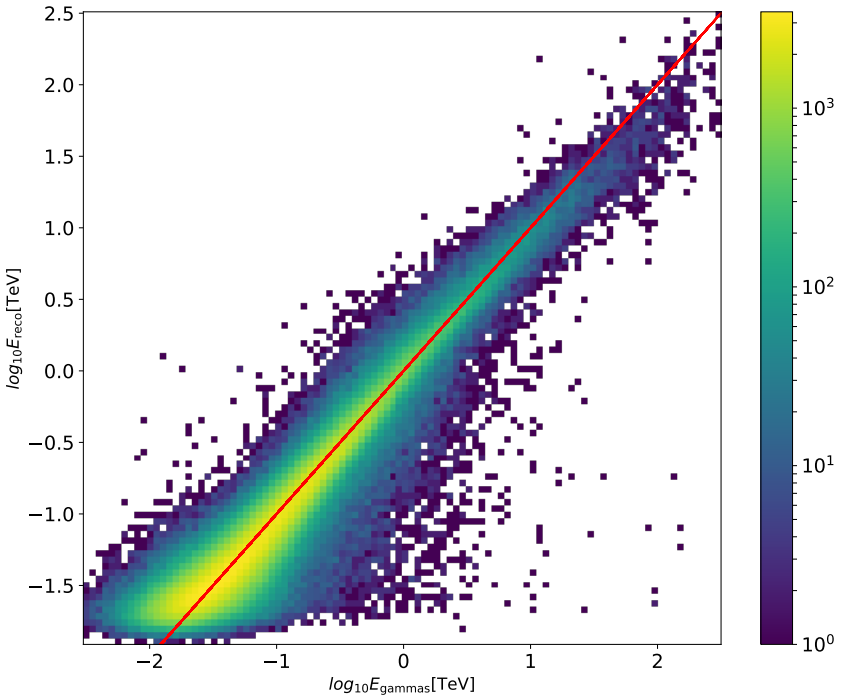

Figure 4.11: 2D histogram of the true and reconstructed energy values by the model ResNetFSE on the test set.

Model	PT	MAE
Random Forest A.2	yes	0.227
VGG-9 A.1	no	0.191
VGG-9 A.1	yes	0.141
DenseNet-BC $(L = 64, k = 12)$ A.5	yes	0.127
ResNetH A.4	yes	0.133
ResNetFSE A.3	yes	**0.125**

Table 4.3: Summary results for energy estimation on the test set. PT denotes if the model has access to the peak time information (i.e. the time gradient for the RF and the additional tensor dimension for the CNNs).

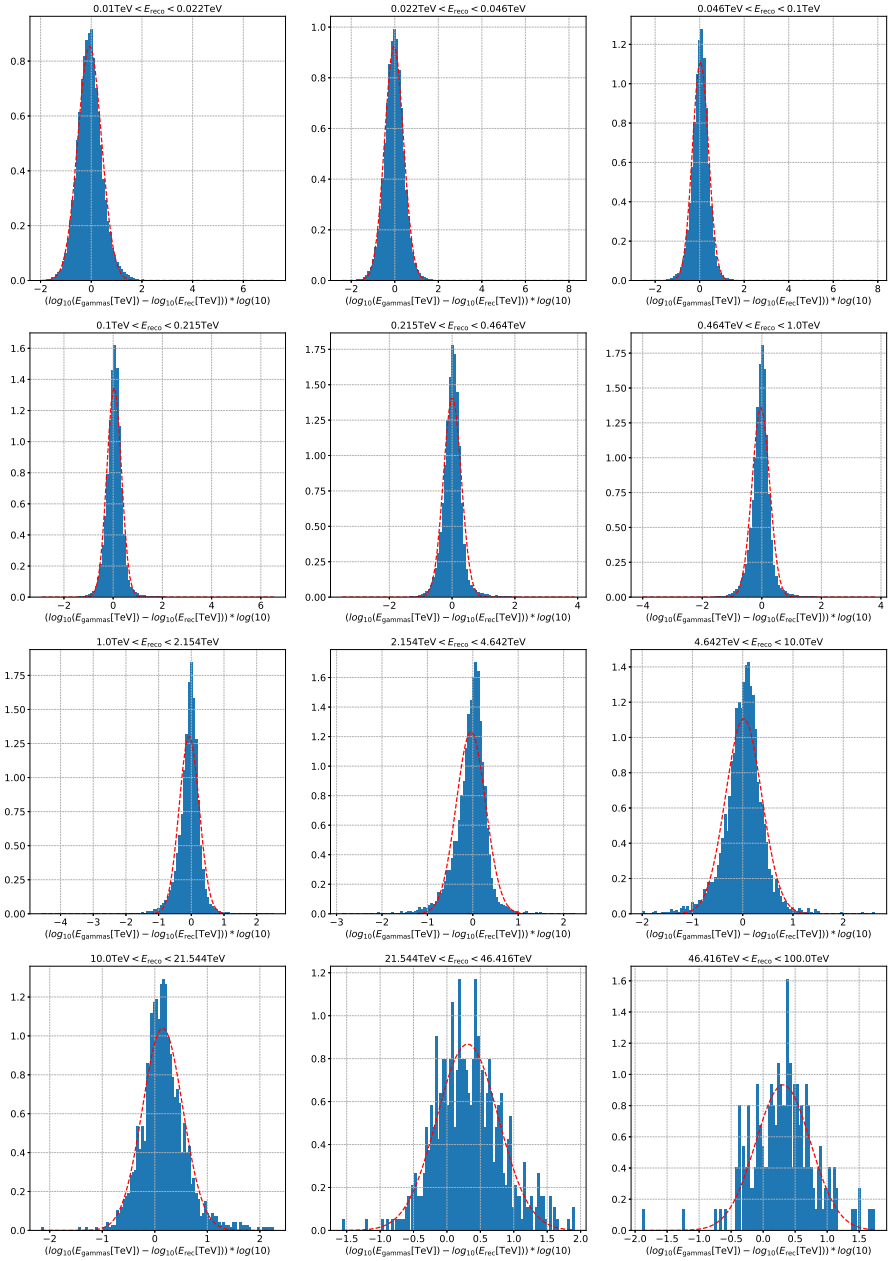

Figure 4.12: Energy biases based on the predictions of the model VGG-9 on the test set.

5 CONCLUSIONS

In this thesis, we designed a new full single-telescope CNN analysis chain for the LST of CTA. We designed models to perform γ/hadron separation, energy and direction reconstruction. For each task we explored different handcrafted architectures based on different frameworks, trying to derive the best possible models. In addition, each model has been compared with the current standard analysis chain for the LST based on RF.

On the γ/hadron separation task, the proposed models, significantly outperform the RF. The VGG-9 baseline model already overtakes the accuracy the RF by 11.8% and the AUC score by 11.6%. Additionally, the ResNetFSE, based on the residual learning framework and SE block, overtakes the accuracy of the RF by 14.5% and the AUC score by 13.5%, while keeping much lower the amount of training and inference time required when compared to the DenseNet. The results show that the CNN performs particularly well on events between 0.1 TeV to 100 TeV, increasing the accuracy as the event energy increases, while refolding at the highest energies. The performances on the highest energies were not meeting the expectations because the dataset provided for the CNN reconstruction used a non-optimal charge extraction method, discussed in Section 3.3.

On direction reconstruction, all the models that we explored overtake the performances achieved by the RF. They provide a good angular resolution having very similar performances. The VGG-9 baseline model overtakes the RF angular resolution up to $\sim$ 182% while the ResNetFSE performed the best in this task as well, with an improvement up to $\sim$ 271%. Although the angular resolution was expected to improve at the highest energies, this did not occur due to the low statistics and the gain-related dataset issue mentioned in Section 3.3.

We explored different architectures for the energy reconstruction task as well. All the models have very good performances as they perform much better than the RF in terms of energy resolution, while most of them are also able to keep the bias low. They overtake the performances across the entire energy spectrum and the ResNetFSE improved the energy resolution up to $\sim$ 161% with a MAE_e improvement of 45.33%. The models perform especially well at the low and middle energies where they achieve the best energy resolution and the lowest bias. This behaviour is mainly induced by the better quality of the events. Even in this task the low statistics and the gain-related dataset issue discussed in Section 3.3 have clearly affected the performances at the highest energies where the model resolutions drops.

The VGG-9 expressed good performances on energy resolution at the cost of a higher bias which is, indeed, an expected behaviour from models with less parameters. According to these results, increasing the complexity of the network did not improve its performance in terms of energy resolution, yet reduced the bias predictions.

Finally, we analyzed the impact of the peak times image on the performances for each task. We have seen that the peak times channel included in the input tensor is beneficial for all the tasks: it improves the performances, having a dramatical impact on the energy and direction reconstruction tasks.

The results derived in this thesis have shown that, to obtain better performances than RF, it is not necessary to employ very complex and state-of-the-art models. Indeed, the VGG-9 model is a very simple and shallow model which provides good performances across all the tasks, nevertheless state-of-the-art CNNs, as the ResNetFSE, are required to achieve the highest performances. The SE block, applied to residual networks, has proven to be effective in the context of IACT data analysis, while only slightly increasing the computational complexity. On the other hand, we did not find convenient the application of densely connected networks in this context, as they achieve comparable performances to other architectures, while being by far the most computationally demanding, in terms of memory requirements as well. Indeed, the amount of time necessary to train them and to use them for inference is visibly higher compared to other models based on different frameworks.

Given the above, CNN reconstruction applied to the single-telescope LST looks very promising. The presented analysis chain achieves a new state-of-the-art performances across all the tasks needed to perform a full single-telescope reconstruction of the LST, at the cost of a higher computational complexity with respect to the RF. It has been shown that CNNs are able to exploit the additional pixel-wise information brought by the event image to overtake the RF. Future developments can include the usage of the SE block on other architectures, the exploration of architectures based on different frameworks and the application of the models derived in this thesis on a new fixed dataset, that is not affected by the gain-related issue. To overcome the issue that affected the final results at the highest energies, it is necessary to produce a new dataset for CNN studies where the signal extraction has the proper gain selection. Another possible future trial could be to apply a custom hexagonal convolution and hexagonal pooling operations that are directly applied on the hexagonal grid image, as the one presented in (Steppa and Holch, 2019). This preserves the original event image and reduces the computational complexity. To conclude, the next natural step would be to apply the models on real data to investigate whether the performances are in line with the results derived in this thesis. The risk is that the models have

learnt well the simulation setup characteristics and are looking for features that are not present in images acquired with the real telescope, with consequent performance degradation. This performance can be checked using real data when the LST1 finishes its commissioning in the next months.

APPENDIX

A.1 CONVOLUTIONAL NEURAL NETWORKS ARCHITECTURES

The network architectures are kept constant across the three different tasks, except for the last fully connected layer. When the task performed by the CNN is γ/hadron separation the last layer is made of a single neuron with the sigmoid activation function, when the task is energy or direction reconstruction the last layer is made of one or two linear nodes respectively. All models were implemented using the order Conv - BN - ReLU on convolutional layers.

A.1.1 *VGG-9*

The VGG-9 is a 9-layers CNN based on the work of (Simonyan and Zisserman, 2015). It consists of a stacked convolutional and pooling layers and it is used as baseline model in this thesis.

layer name	VGG-9 layer
conv_1	3×3, 32, stride 1
conv_2	3×3, 32, stride 1
MaxPooling	2×2, stride 2
conv_3	3×3, 64, stride 1
conv_4	3×3, 64, stride 1
MaxPooling	2×2, stride 2
conv_5	3×3, 128, stride 1
conv_6	3×3, 128, stride 1
MaxPooling	2×2, stride 2
FC	32
FC	128
FC	1-d (2-d) sigmoid (linear)
Params	$\sim 0.625 \times 10^6$

Table A.1: VGG-9 architecture.

A.1.2 *ResNetF*

The ResNetF is a CNN based on the work of He et al., 2016. It is made of 5 convolutional blocks which contain the ResNet layers and the shortcut connections.

layer name	26-layer
conv1	3×3, 16, stride 1
conv2	$\begin{bmatrix} 3\times3, 16 \\ 3\times3, 16 \end{bmatrix} \times 3$
conv3	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 3$
conv4	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 3$
conv5	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 3$
FC	avg pool, 1-d (2-d) fc, sigmoid (linear)
Params	$\sim 1.1 \times 10^6$

Table A.2: ResNetF architecture.

A.1.3 *ResNetFSE*

The ResNetFSE is a 50-layers CNN inspired by the work of (Hu, Shen, and Sun, 2018). It has the same structure as the ResNetF described in Table A.2 with a SE block at the end of each convolutional block. Each SE block has the same ratio used in the original paper $r = 16$.

layer name	50-layer
conv1	3×3, 16, stride 1
conv2	$\begin{bmatrix} 3\times3, 16 \\ 3\times3, 16 \\ SE-fc(r=16) \end{bmatrix} \times 3$
conv3	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \\ SE-fc(r=16) \end{bmatrix} \times 3$
conv4	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \\ SE-fc(r=16) \end{bmatrix} \times 3$
conv5	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \\ SE-fc(r=16) \end{bmatrix} \times 3$
FC	avg pool, 1-d (2-d) fc, sigmoid (linear)
Params	$\sim 1.1 \times 10^6$

Table A.3: ResNetFSE architecture. The ratio parameter of the SE block is indicated with r.

A.1.4 *ResNetH*

The ResNetH is a 58-layers CNN based on the work of He et al., 2016. It is made of 5 convolutional blocks which contain the ResNet layers and the shortcut connections.

layer name	58-layer
conv1	3×3, 16, stride 1
conv2	$\begin{bmatrix} 3\times3, 8 \\ 3\times3, 8 \end{bmatrix} \times 3$
conv3	$\begin{bmatrix} 3\times3, 16 \\ 3\times3, 16 \end{bmatrix} \times 4$
conv4	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 6$
conv5	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 6$
conv6	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 9$
FC	avg pool, 1-d (2-d) fc, sigmoid (linear)
Params	$\sim 1.1 \times 10^6$

Table A.4: ResNetH architecture.

A.1.5 *DenseNet-BC* $(L = 64, k = 12)$

The DenseNet-BC $(L = 64, k = 12)$ is a 64-layers CNN base on the concept introduced by Huang et al., 2017. It makes use of *bottleneck layers* which reduce the computational load and *compression* toreduce the number of feature maps at transition layers.

layer name	64-layer
conv1	3×3, 24, stride 1
Dense block	$\begin{bmatrix} 1\times1, \text{conv} \\ 3\times3, \text{conv} \end{bmatrix} \times 10$
Transition layer	1×1 conv
	2×2 average pool, stride 2
Dense block	$\begin{bmatrix} 1\times1, \text{conv} \\ 3\times3, \text{conv} \end{bmatrix} \times 10$
Transition layer	1×1 conv
	2×2 average pool, stride 2
Dense block	$\begin{bmatrix} 1\times1, \text{conv} \\ 3\times3, \text{conv} \end{bmatrix} \times 10$
fully connected	global average pooling 2×2
	1-d (2-d) fc, sigmoid (linear)
Params	$\sim 0.377 \times 10^6$

Table A.5: DenseNet-BC $(L = 64, k = 12)$ architecture. The network is 64-layer deep, with growth rate $k = 12$, bootleneck layers and compression $= 0.5$

A.2 DATA GENERATOR

Listing A.1: Classifier data generator

```
class DataGeneratorC(keras.utils.Sequence):

    def __init__(self, h5files, batch_size=32, arrival_time=False
        , shuffle=True):
        self.batch_size = batch_size
        self.h5files = h5files
        self.indexes = np.array([], dtype=np.int64).reshape(0, 4)
        self.shuffle = shuffle
        self.generate_indexes()
        self.arrival_time = arrival_time
        self.on_epoch_end()

    def __len__(self):
        # Denotes the number of batches per epoch
        return int(np.floor(self.indexes.shape[0] / self.
            batch_size))

    def __getitem__(self, index):
        # Generate one batch of data
        indexes = self.indexes[index * self.batch_size:(index +
            1) * self.batch_size]

        # Generate data
        x, y = self.__data_generation(indexes)

        return x, y

    def get_indexes(self):
        return self.indexes[0:self.__len__() * self.batch_size]

    def chunkit(self, seq, num):
        avg = len(seq) / float(num)
        out = []
        last = 0.0

        while last < len(seq):
            out.append(seq[int(last):int(last + avg)])
            last += avg

        return out

    def worker(self, h5files, positions, i, return_dict):
        idx = np.array([], dtype=np.int64).reshape(0, 4)
        for l, f in enumerate(h5files):
            h5f = h5py.File(f, 'r')
            lst_idx = h5f['LST/LST_event_index'][:]
            h5f.close()
            r = np.arange(len(lst_idx))
```

```
            fn_basename = os.path.basename(os.path.normpath(f))
            clas = np.zeros(len(r))  # class: proton by default
            if fn_basename.startswith('g'):
                clas = np.ones(len(r))

            cp = np.dstack(([positions[l]] * len(r), r, clas,
                lst_idx)).reshape(-1, 4)

            idx = np.append(idx, cp, axis=0)
        return_dict[i] = idx

    def generate_indexes(self):
        cpu_n = multiprocessing.cpu_count()
        pos = self.chunkit(np.arange(len(self.h5files)), cpu_n)
        h5f = self.chunkit(self.h5files, cpu_n)

        manager = multiprocessing.Manager()
        return_dict = manager.dict()
        processes = []

        if cpu_n >= len(self.h5files):
            # print('ncpus >= num_files')
            for i, f in enumerate(self.h5files):
                p = multiprocessing.Process(target=self.worker,
                    args=([f], [i], i, return_dict))
                p.start()
                processes.append(p)
        else:
            # print('ncpus < num_files')
            for i in range(cpu_n):
                p = multiprocessing.Process(target=self.worker,
                    args=(h5f[i], pos[i], i, return_dict))
                p.start()
                processes.append(p)

        for p in processes:
            p.join()

        for key, value in return_dict.items():
            self.indexes = np.append(self.indexes, value, axis=0)

    def on_epoch_end(self):
        # Updates indexes after each epoch
        if self.shuffle:
            np.random.shuffle(self.indexes)

    def __data_generation(self, indexes):
        # Initialization
        x = np.empty([self.batch_size, 100, 100, self.
            arrival_time + 1])
        y = np.empty([self.batch_size], dtype=int)
```

```
        # Generate data
        for i, row in enumerate(indexes):

            filename = self.h5files[int(row[0])]

            h5f = h5py.File(filename, 'r')
            # Store image
            x[i, :, :, 0] = h5f['LST/LST_image_charge_interp'][
                int(row[1])]
            if self.arrival_time:
                x[i, :, :, 1] = h5f['LST/
                    LST_image_peak_times_interp'][int(row[1])]
            # Store class
            y[i] = int(row[2])

            h5f.close()

        return x, y
```

A.3 RANDOM FOREST INPUT PARAMETERS

Listing A.2: Random Forest input parameters

```
features = ['intensity',
            'time_gradient',
            'width',
            'length',
            'wl',
            'phi',
            'psi',
            'x',
            'y',
            'skewness',
            'kurtosis',
            'r',
            'leakage2_intensity',
            'n_islands',
            'intercept']
```

BIBLIOGRAPHY

Abadi, Martín et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: https://www.tensorflow.org/.

Abdel-Hamid, O. et al. (2014). "Convolutional Neural Networks for Speech Recognition." In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22.10, pp. 1533–1545. ISSN: 2329-9290. DOI: 10.1109/TASLP.2014.2339736.

Acharya, B. S. et al. (Mar. 2013). "Introducing the CTA concept." In: *Astroparticle Physics* 43, pp. 3–18. DOI: 10.1016/j.astropartphys.2013.01.007.

Aharonian, F. et al. (2006). "Observations of the Crab nebula with HESS." In: *Astronomy and Astrophysics* 457.3, pp. 899–915. DOI: 10.1051/0004-6361:20065351. arXiv: astro-ph/0607333 [astro-ph].

Albert, J. et al. (2008). "Implementation of the Random Forest method for the Imaging Atmospheric Cherenkov Telescope MAGIC." In: *Nuclear Instruments and Methods in Physics Research A* 588.3, pp. 424–432. DOI: 10.1016/j.nima.2007.11.068. arXiv: 0709.3719 [astro-ph].

Cherenkov, P. A. (1934). "Visible emission of clean liquids by action of γ radiation." In: *Doklady Akademii Nauk SSSR* 2, pp. 451+.

Chollet, François et al. (2015). *Keras*. https://keras.io.

De Angelis, A. et al. (2018). "Gamma-ray astrophysics." In: *The European Physical Journal Plus* 133.8, p. 324. ISSN: 2190-5444. DOI: 10.1140/epjp/i2018-12181-0. URL: https://doi.org/10.1140/epjp/i2018-12181-0.

Feng, Qi et al. (2017). "The analysis of VERITAS muon images using convolutional neural networks." In: *Astroinformatics*. Ed. by Massimo Brescia et al. Vol. 325. IAU Symposium, pp. 173–179. DOI: 10.1017/S1743921316012734. arXiv: 1611.09832 [astro-ph.IM].

Glorot, Xavier et al. (2011). "Deep Sparse Rectifier Neural Networks." In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Geoffrey Gordon et al. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, pp. 315–323. URL: http://proceedings.mlr.press/v15/glorot11a.html.

Hanlon, William (2019). *Cosmic Ray Spectra of Various Experiments*. URL: http://www.physics.utah.edu/~whanlon/spectrum.html (visited on 01/27/2019).

He, K. et al. (2016). "Deep Residual Learning for Image Recognition." In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society,

pp. 770–778. DOI: 10.1109/CVPR.2016.90. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.90.

Hillas, A. M. (Aug. 1985). "Cerenkov light images of EAS produced by primary gamma." In: *International Cosmic Ray Conference* 3.

Hu, J. et al. (2018). "Squeeze-and-Excitation Networks." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141. DOI: 10.1109/CVPR.2018.00745.

Huang, G. et al. (2017). "Densely Connected Convolutional Networks." In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269. DOI: 10.1109/CVPR.2017.243.

Ioffe, Sergey et al. (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." In: pp. 448–456. URL: http://jmlr.org/proceedings/papers/v37/ioffe15.pdf.

Jarrett, Kevin et al. (2009). "What is the best multi-stage architecture for object recognition?" In: *ICCV*. IEEE, pp. 2146–2153. URL: http://dblp.uni-trier.de/db/conf/iccv/iccv2009.html#JarrettKRL09.

Jiang, H. et al. (2018). "Super SloMo: High Quality Estimation of Multiple Intermediate Frames for Video Interpolation." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9000–9008. DOI: 10.1109/CVPR.2018.00938.

Kingma, Diederik P. et al. (2014). *Adam: A Method for Stochastic Optimization*. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. URL: http://arxiv.org/abs/1412.6980.

Krizhevsky, Alex et al. (2012). "ImageNet Classification with Deep Convolutional Neural Networks." In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

LeCun, Y. et al. (Dec. 1989). "Backpropagation Applied to Handwritten Zip Code Recognition." In: *Neural Computation.* 1.4, pp. 541–551. ISSN: 0899-7667. DOI: 10.1162/neco.1989.1.4.541. URL: http://dx.doi.org/10.1162/neco.1989.1.4.541.

Lecun, Y. et al. (1998). "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE* 86.11, pp. 2278–2324. ISSN: 0018-9219. DOI: 10.1109/5.726791.

Liu, Wei et al. (2016). "SSD: Single Shot MultiBox Detector." In: To appear. URL: http://arxiv.org/abs/1512.02325.

Lukas Holch, Tim et al. (2017). "Probing Convolutional Neural Networks for Event Reconstruction in γ-Ray Astronomy with Cherenkov Telescopes." In: *arXiv e-prints*, arXiv:1711.06298, arXiv:1711.06298. arXiv: 1711.06298 [astro-ph.IM].

Mangano, S. et al. (2018). "Extracting gamma-ray information from images with convolutional neural network methods on simulated

Cherenkov Telescope Array data." In: *arXiv e-prints*, arXiv:1810.00592, arXiv:1810.00592. arXiv: 1810.00592 [astro-ph.IM].

Murach, Thomas et al. (2015). "A Neural Network-Based Monoscopic Reconstruction Algorithm for H.E.S.S. II." In: *arXiv e-prints*, arXiv:1509.00794, arXiv:1509.00794. arXiv: 1509.00794 [astro-ph.IM].

Nair, Vinod et al. (2010). "Rectified Linear Units Improve Restricted Boltzmann Machines." In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pp. 807–814. URL: https://icml.cc/Conferences/2010/papers/432.pdf.

Nieto, D. et al. (2017). "Exploring deep learning as an event classification method for the Cherenkov Telescope Array." In: *arXiv e-prints*, arXiv:1709.05889, arXiv:1709.05889. arXiv: 1709.05889 [astro-ph.IM].

Parsons, R. D. et al. (2014). "A Monte Carlo template based analysis for air-Cherenkov arrays." In: *Astroparticle Physics* 56, pp. 26–34. DOI: 10.1016/j.astropartphys.2014.03.002. arXiv: 1403.2993 [astro-ph.IM].

Shilon, I. et al. (2019). "Application of deep learning methods to analysis of imaging atmospheric Cherenkov telescopes data." In: *Astroparticle Physics* 105, pp. 44–53. DOI: 10.1016/j.astropartphys.2018.10.003. arXiv: 1803.10698 [astro-ph.IM].

Simonyan, K. et al. (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition." In: *International Conference on Learning Representations*.

Srivastava, Nitish et al. (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." In: *Journal of Machine Learning Research* 15, pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

Steppa, Constantin et al. (2019). "HexagDLy—Processing hexagonally sampled data with CNNs in PyTorch." In: *SoftwareX* 9, pp. 193 –198. ISSN: 2352-7110. DOI: https://doi.org/10.1016/j.softx.2019.02.010. URL: https://www.sciencedirect.com/science/article/pii/S2352711018302723.

Wagner, Robert Marcus (2007). "Measurement of Very High Energy Gamma-Ray Emission from Four Blazars Using the MAGIC Telescope and a Comparative Blazar Study." In: *Publications of the Astronomical Society of the Pacific* 119.860, pp. 1201–1203. DOI: 10.1086/522380. URL: https://doi.org/10.1086%2F522380.

cta-observatory (2018). *cta-lstchain*. https://github.com/cta-observatory/cta-lstchain.

de Naurois, Mathieu et al. (2009). "A high performance likelihood reconstruction of γ-rays for imaging atmospheric Cherenkov telescopes." In: *Astroparticle Physics* 32, pp. 231–252. DOI: 10.1016/j.astropartphys.2009.09.001. arXiv: 0907.2610 [astro-ph.IM].