Universidade Federal de Juiz de Fora
Programa de Pós-Graduação em
Engenharia Elétrica

Guilherme Sebastião Pinheiro Lopes

Impact of Filtering on CYGNO Experiment

Juiz de Fora
2022

Guilherme Sebastião Pinheiro Lopes

Impact of Filtering on CYGNO Experiment

Thesis presented to the Graduate Degree Program of Electrical Engineering from the Federal University of Juiz de Fora as a partial requirement for obtaining the Master's degree in Electrical Engineering.
Concentration Field: Electronic Systems.

Advisors: Prof. Rafael Antunes Nóbrega, D.Sc.
Prof. Igor Abritta Costa, D.Sc.

Juiz de Fora
2022

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF com os dados fornecidos pelo(a) autor(a)

Lopes, Guilherme Sebastião Pinheiro.

Impact of Filtering on CYGNO Experiment / Guilherme Sebastião Pinheiro Lopes. - 2022

86 f. : il.

Advisor: Rafael Antunes Nóbrega

Coadvisor: Igor Abritta Costa

Dissertation ((Master's Degree) - Federal University of Juiz de Fora, Engineering School. Postgraduate Program of Electrical Engineering, 2022.

1. CYGNO. 2. Image Processing. 3. Deep Learning. I. Nóbrega, Rafael, adv. II. Costa, Igor, coadv. III. Título

FEDERAL UNIVERSITY OF JUIZ DE FORA

RESEARCH AND GRADUATE PROGRAMS OFFICE

Guilherme Sebastião Pinheiro Lopes

Impact of filtering on CYGNO's reconstruction algorithm

Dissertation submitted to the Graduate Program in Electrical Engineering of the Federal University of Juiz de Fora as a partial requirement for obtaining a Master's degree in Electrical Engineering. Concentration area: Electronic Systems

Approved on 18 of March of 2022.

Juiz de Fora, 03/18/2022.

Dedico este trabalho a Deus, que me presenteia diariamente com a vida e saúde para alcançar meus objetivos.

ACKNOWLEDGMENTS

Agradeço primeiramente a Deus por todas as bençãos derramadas em minha vida e por sempre me amparar nos momentos em que mais precisei.

Aos meus pais, Maria e Anselmo, por acreditarem em mim e sempre estarem ao meu lado, me apoiando em todas as decisões. Às minhas irmãs, Ana Caroline e Tatiane, e aos meus sobrinhos, Rafael, Sarah e Thiago. Obrigado por todo apoio e carinho.

À minha esposa Tamira, pelos 12 anos de companherismo, e por todo amor e carinho que sempre teve por mim.

À minha filha Stella, motivo das minhas maiores alegrias e razão pela qual luto para ser uma pessoa melhor a cada dia.

Ao meu sogros Sérgio e Deise, por todo auxílio e carinho que sempre tiveram por mim.

Ao meu querido Professor Rafael, um parceiro de vida que sempre me motivou e acreditou no meu trabalho, me ajudando sempre a tomar as melhores decisões. Grande parte deste trabalho (e dos outros que virão!) é por causa das ricas contribuições trazidas por você. Obrigado por tudo.

Ao meu coorientador e amigo Igor, por sempre me dar os melhores conselhos e me apoiar nos momentos em que mais precisei.

Aos meus amigos do NIPS e, em especial, Alan, Amaro, Antônio, Mariana, David, Fhelipe, Tiago e Paulo, por todo companherismo durante minha jornada na UFJF.

To CYGNO collaboration for promoting this work and always believing in our potential.

Finally, I would like to thank CAPES (Coordination for the Improvement of Higher Education Personnel), the Federal University of Juiz de Fora and the Graduation Degree Program in Electrical Engineering for all the support and tools necessary to the development of this work.

RESUMO

Este trabalho visa avaliar o impacto da utilização de filtros digitais no processo de detecção de pixels em um detector de particulas do tipo TPC, com leitura ótica baseada em sensores de imagem de alta resolução.

Para esta análise, uma metodologia de avaliação baseada no uso de dados de simulação obtidos no âmbito do Experimento CYGNO também é proposta. Por fim, dados reais são analisados como forma de validar os resultados. Para fins comparativos, algumas técnicas clássicas de filtragem foram selecionadas, juntamente com a proposta de utilizar uma rede neural convolucional para realizar a seleção de pixels de interesse, com o objetivo de verificar as vantagens de se utilizar tais técnicas em uma etapa de pré-processamento dos dados, principalmente no que tange a estimação de energia e tempo de processamento. Os resultados obtidos demonstraram que uma rede convolucional tem potencial para melhorar o desempenho da etapa de processamento das imagens geradas pelo detector e que uma filtragem não-linear clássica consegue reproduzir um resultado similar ao do algoritmo utilizado pelo experimento em um tempo quatro vezes menor.

Palavras-chave: Filtragem, CYGNO, Redes Neurais Convolucionais

ABSTRACT

This work proposes to study the impact of classic digital filters and a convolutional neural network in the detection process of pixels with the presence of a signal formed from the release of energy produced by particles that interact inside a TPC detector which makes use of an optical readout based on high-resolution image sensors. For this analysis, an evaluation methodology based on the use of simulation data obtained within the scope of the CYGNO Experiment is also proposed. Finally, real data are analyzed as a way to validate the results. Some classical filtering techniques were selected for comparative purposes, together with a convolutional neural network trained to perform the selection of pixels of interest, with the objective of verifying the advantages of using such techniques in a pre-processing stage of the data, especially regarding energy estimation and processing time. The results obtained showed that a convolutional network has the potential to improve the performance of the processing stage of the images generated by the detector and that a classical non-linear filtering can reproduce a result similar to the algorithm used by the experiment in a time four times shorter.

Keywords: Filtering, Convolution Neural Network, CYGNO.

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CNN Convolutional Neural Networks

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DM Dark Matter

GEM Gas Electron Multiplier

FCN Fully Convolutional Network

iDBSCAN intensity-based DBSCAN

KDE Kernel Density Estimator

LEMOn Large Elliptical Module

MPGD Micro Pattern Gas Detector

NNC Nearest Neighbor Clustering

PMT Photomultiplier Tube

ReLU Rectified Linear Units

sCMOS	Scientific Complementary metal-oxide-semiconductor
TPC	Time Projection Chamber
WIMP	Weakly Interacting Massive Particles

CONTENTS

1 INTRODUCTION

From the 1960s to now, the image processing field has been growing rapidly. In the past, it was commonly applied to problems involving areas of medicine and space programs. Image processing techniques are now used in a broad range of applications, including experimental physics where they can be used to improve images of experiments in areas such as high-energy plasma and electron microscopy (GONZALEZ; WOODS et al., 2002).

Another example of application, related to the work of this thesis, is based on gas detectors that can be a choice for detecting particles with low energy release. The light produced by the de-excitation of the gas molecules during the multiplication process of electrons can be captured by a camera (MARAFINI et al., 2018).

The acquired images contain tracks from particles interaction inside the detector. They can be processed so it improves pictorial information and reaches a more effective human interpretation. Moreover, additional information can be extracted and processed by classifiers. These pieces of information can be used in applications in pattern recognition and by using machine learning techniques.

Like feature extraction, the pre-processing step might be mandatory for many image-based applications (HEMANTH; ANITHA, 2012), for improving the detection and classification of patterns in images, mainly on images that have a low signal-to-noise ratio. As in most applications, it is not straightforward to know, at the pixel level, the expected output information given a real dataset. Therefore, to evaluate the impact of any pre-processing technique, a simulated dataset may be essential to assess the performance of any proposed algorithm and once such an algorithm proves to be efficient in the simulation data, it should be applied in practice to validate the results.

The purpose of this research is to measure the impact of digital image filtering for the CYGNO experiment (PINCI et al., 2018) regarding the improvement of the signal-to-noise ratio of its captured images. In a first approach, we evaluate the efficiency of detection of straight tracks using spatial filtering techniques and a deep learning based technique, comparing them to the algorithm used by the experiment at the time this proposal was developed.

This work is organized as follows: Section 2 will present a literature review about dark matter detection and image processing techniques. The CYGNO experiment and the datasets used for this work will be explained in Section 3. In Section 4, the proposed methodology is defined. Section 5 shows the relevant results and Section 6 presents the conclusion.

2 LITERATURE REVIEW

2.1 DIRECT DARK MATTER DETECTION

During the last century, the standard model of particle physics has been gradually built up. There has been a great effort of different scientists and researchers to try to create a model that could explain the elementary particles and their interactions in order to understand everything that can be observed in the universe. The theory proved incredibly successful, accurately describing almost all empirical data; however, some exceptions were found (KIBBLE, 2015).

About a century ago, astronomers suggested the existence of a hypothetical type of matter called *dark matter*, that is invisible in the entire electromagnetic spectrum (LIU; CHEN; JI, 2017). From then on, the dominance of dark matter and its role in driving the evolution and landscape of our universe have become the standard paradigm in cosmology (BERTONE; HOOPER; SILK, 2005). Furthermore, dark matter is estimated to be five times more abundant than the so-called baryonic matter and has essentially shaped the universe that can be observed today.

Aiming to prove the existence of dark matter, several models have been created in attempt to explain it. The most accepted models are those based on the interactions of the so-called WIMP (Weakly Interacting Massive Particles).

2.1.1 WIMP EXPECTED INTERACTION

Microscopically, WIMP, with typical masses around 100 GeV/c^2, is a generic class of dark matter. It is a candidate favored by many theories beyond the Standard Model of Particle Physics (SMITH et al., 2007).

The possibility of direct detection of dark matter by observing the interactions of WIMP was first discussed in 1985 (GOODMAN; WITTEN, 1985). Since they do not have electrical charge, in most cases they do not interact with atomic electrons, but interact in the so-called elastic dispersion, an interaction with atomic nuclei. During the interaction with the atomic nucleus, momentum transfer gives rise to detectable atomic recoil energy (SCHUMANN, 2019). The total energy loss of recoil in a WIMP detector can be described by Equation 2.1.

$$\left(\frac{dE}{dx}\right)_{tot} = \left(\frac{dE}{dx}\right)_{elec} + \left(\frac{dE}{dx}\right)_{nucl} \tag{2.1}$$

Most part of the energy induced by the interaction between a WIMP and an atomic nucleus is dissipated as heat. The remaining energy is electronic energy losses that can ionize or excite atoms. This atomic excitation can produce scintillation light, and it is possible to detect such light by using photosensors. However, since a very small portion of the signal is generated, the number of photons available for this detection is too low, which makes the task of detectors that seek to identify these particles very difficult. The electrons and ions formed by ionization, in turn, can be captured to form a signal of greater amplitude, facilitating the detection of the event.

2.1.2 DETECTORS FOR WIMP SEARCHES

Currently, direct dark matter detection experiments seek to obtain information through collisions between known particles and those explained by the Standard Model and dark matter. Such a collision is illustrated in Figure 1, where the open circle represents the WIMP and the dark circle represents the target nucleus. The result of this collision is two particles, one WIMP and another from the standard model. After ionization of the material, when the electron fills the hole in the atomic electrosphere, the emitted photon is detected by a photosensor. Such ejected electron can be accelerated by a external field, interacting with other material and emitting more photons, which could be detected.

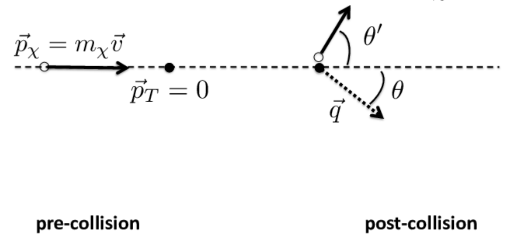


Figure 1: WIMP collision example (PETER et al., 2014)

The DAMA experiment (PETRIELLO; ZUREK, 2008) has tried to detect WIMPs from the so-called Annual Modulation (FREESE; LISANTI; SAVAGE, 2012). Due to the relative motion of the Earth around the Sun, it would be possible to detect in laboratory the most energetic event only once a year, when the greatest speed of the earth moving through the solar system is reached. In doing so, the relative speed between the target material (used in the detector) and the dark matter particle would be increased, as well as the energy released, consequently. Such effect was detected by DAMA experiment, but no other experiment has found similar results.

Other experiments are still trying to look for these dark matter signals, testing different detector designs. Some examples are DAMIC (COLLABORATION; NETO et al., 2016), MIMAC (IGUAZ et al., 2011), PICASSO (BEHNKE et al., 2017) and EDELWEISS (AHMED et al., 2011). These experiments look for low energy nuclear recoils (10 - 100 keV) due to the elastic scattering of WIMPs within the detector's active volume. However, due to the low probability of occurrence of such expected interactions, it is necessary to control, minimize or even reject any background source that is indistinguishable from the target signal. Much of this effort, apart from using underground facilities, is concentrated in the data analysis task.

The CYGNO experiment uses the directional detector concept with the goal of detecting dark matter. The principle of this detector concept is based on a Time Projection Chamber (TPC) gas detector being readout by a high-granularity sensor. The WIMP particles interact with gas, releasing electrons that are drifted by an electric field until reaching a Triple-GEM device. Then, the Triple-GEM device collects and multiplies those electrons by an avalanche process that will produce photons which, in turn, might be captured by an image sensor. By studying the energy profile of such events, it would be possible to estimate the direction of the incident particle. Therefore, the most important output provided by the CYGNO detector is given by images that can reveal the tracks produced by the interaction of the incident particles with the TPC gas.

2.2 DIGITAL IMAGE PROCESSING AND NOISE REDUCTION

Vision is one of the most advanced human senses, and images play a key role in human perception. Unlike humans, who can distinguish a limited visual range of the electromagnetic (EM) spectrum, imaging machines cover almost the entire EM spectrum, ranging from radio to gamma waves. They can operate on images generated by sources that humans are not used to associate with images, including ultrasound and electron microscopy (GONZALEZ; WOODS et al., 2002).

Experiments such as those discussed in this work produce a large volume of information that is not visible to the naked eye. However, when they are captured by sensors that reveal the interactions that occur inside a given detector, one can have interpretable quantities, like images, for example. The main components for performing tasks like this are shown in figure 2.

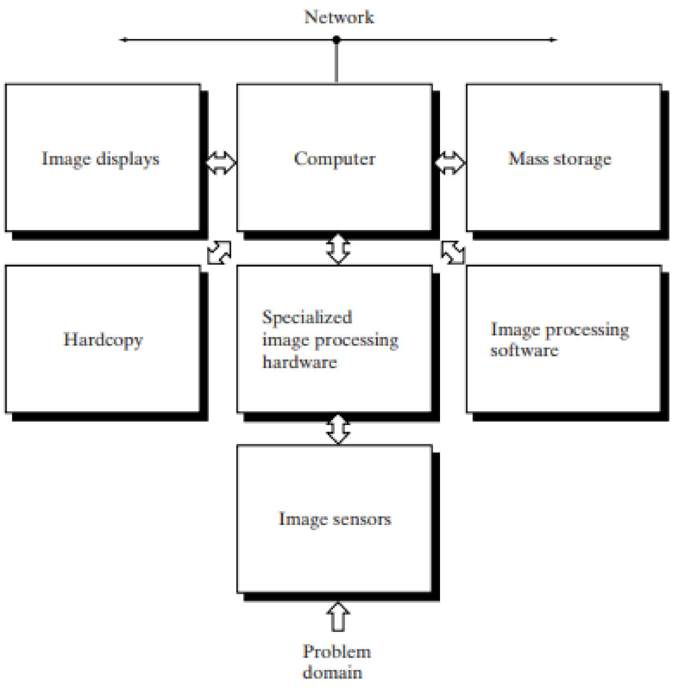


Figure 2: Components of a general-purpose image processing system (GONZALEZ; WOODS et al., 2002)

Not all components seen in figure 2 are required in an image processing system. The most important for the problem addressed in this work are detailed below:

- Image sensor: This component includes two elements to acquire digital images. The first is a physical device that is responsible to convert the radiated energy into images - the sensor itself. The second is a digitizer, which receives the output of the physical device and transforms it into a digital format.
- Specialized image processing hardware: It is a hardware that can perform in parallel arithmetic and logic operations on entire images. This step becomes extremely important if it is required fast image processing. An interesting appli-

cation would be to discard events that are not interesting, avoiding the execution of later steps, such as storage.

- Computer: Used for offline image processing tasks, as those that have been developed in this work.
- Mass storage: This element stores the images for a specific time before the analysis step, for example.

The present work deals with images from the image sensors, processing them in computers that belong to the offline sector of the acquisition system. In the next section, the main image processing concepts and techniques used in here will be defined.

2.2.1 DIGITAL IMAGE DEFINITION

An image can be described as a two-dimensional function, $f(x, y)$, where x and y represent the spatial plane, and the value of f, at each point, denotes the intensity of the image at that point. We have configuration of a digital image when x, y and f are finite and made of discrete values. Each combination composed by a location (x,y) is called pixel, that is the most widely used term to denote the elements of a digital image (GONZALEZ; WOODS et al., 2002); $f(x, y)$ represents its intensity.

Digital images can also be interpreted as a continuous image passed by two processes: sampling and quantization. Sampling will transform a spatial plane from continuous range to a discrete one; quantization will do the same for intensity, making $f(x, y)$ have a discrete integer value. It allows the representation of a digital image by a widely used mathematical class: matrices. Thus, a NxM image representation is shown in equation 2.2.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \cdots & f(0,M) \\ f(1,0) & f(1,1) & f(1,2) & \cdots & f(1,M) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ f(N,0) & f(N,1) & f(N,2) & \cdots & f(N,M) \end{bmatrix} \tag{2.2}$$

Color images commonly map a coordinates pair (x,y) in space to a set of values, represented in three dimensions, where each dimension represents the intensity level of the image in the chosen color scale. For the application discussed in the present work, a pair (x, y) leads to exactly one value $f(x, y)$. Images with this type of association are called gray scale images.

Based on the presented concepts concerning the definition of images and their components, the operations used in this work will be explained below, starting from the concept of spatial filtering. The concepts covered in spatial filtering will also serve as a basis for the description of more modern techniques, which were also used in this work, such as Deep Learning.

2.2.2 SPATIAL FILTERING

Spatial filtering is a neighboring procedure, where the value of any particular pixel in the output image is calculated by applying some algorithm to the values of the neighboring pixels of its corresponding pixel (CHAKI; PAREKH; BHATTACHARYA, 2016). Such algorithms can be a set of linear operations applied to the whole image, usually between a pixel and its neighbors, or non-linear operations, keeping the spatial concept.

The concept of spatial filtering is shown in the figure 3. This process consists of moving the matrix w, commonly called a mask, from point to point in the image; for each pair (x, y), the response of the filter at that point is calculated using some predefined relationship. If linear spatial filter is used, the response is given by a sum of the products of mask values and the corresponding image pixels in the area spanned by the filter mask (GONZALEZ; WOODS et al., 2002). An example of linear spatial filtering can be explained by equation 2.3, where the new pixel $f'(x, y)$ is obtained from the linear combination of the pixel $f(x, y)$ and its neighbors in the image with the values w of the applied mask, as it is shown in figure 3.

$$\begin{aligned} f'(x,y) = {} & f(x-1,y-1)w(-1,-1) + f(x-1,y)w(-1,0)\cdots \\ & + f(x,y)w(0,0) + \cdots + f(x+1,y)w(1,0) \\ & + f(x+1,y+1)w(1,1) \end{aligned} \tag{2.3}$$

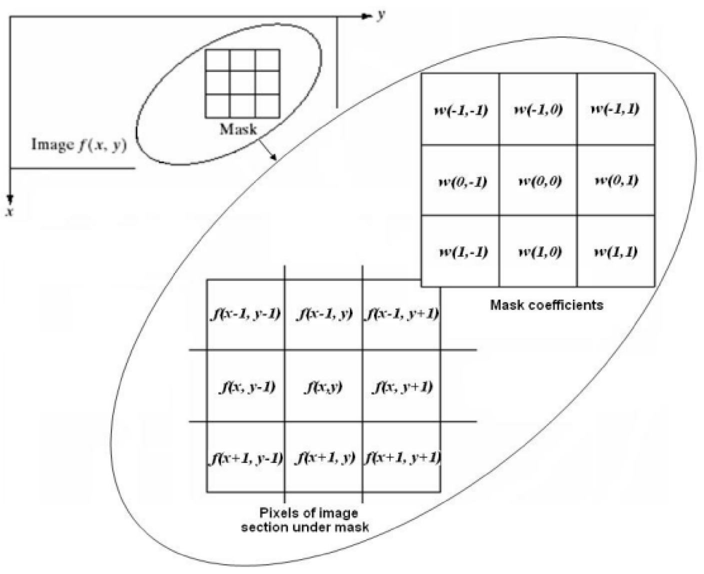


Figure 3: Convolution operator concept

In general, the process of linear filtering of an image f(x,y) of MxN size, submitted to a mask of size mxn, can be described by equation 2.4, where $f'(x,y)$ is the new image created by filtering $f(x,y)$, for x from 0 to $M-1$ and y from 0 to $N-1$.

$$f'(x,y) = \sum_{i=-\frac{m-1}{2}}^{\frac{m-1}{2}} \sum_{j=-\frac{n-1}{2}}^{\frac{n-1}{2}} w(i,j)f(x+i,y+j) \tag{2.4}$$

The operation described by equation 2.4 is also called convolution, and $f'(x,y)$ can be represented by $f(x,y) * w(x,y)$, where $*$ is the convolution operator.

Nonlinear spatial filters also operate on pixel and its neighbors by sliding a mask throughout a whole image, just as it has been explained concerning linear filters. However, the filtering operation is conditionally based on the values of the pixels in the considered neighborhood, and they do not explicitly use coefficients in the sum-of-products (GONZALEZ; WOODS et al., 2002).

The spatial filters used for problems involving noise reduction are known as smoothing filters. For the current work, three filters of this kind were used for the sake of comparison: two linear ones and one nonlinear. These filters will be discussed in the next sections.

2.2.2.1 SMOOTHING LINEAR FILTERS

The main concept behind smoothing linear filters is that this operation simply consists of the average of the pixels contained in the neighborhood of a defined mask.

The simplest smoothing filter used for tasks such as noise reduction is the mean filter. The main idea is simple: one must add all pixels in a defined neighborhood and then divide them by the number of pixels in the chosen neighborhood. The equation that defines a mask of size nxm that performs this operation in an image is defined in equation 2.5, where the ones matrix has nxm size.

For the present work, filters where n=m and n odd will be used. For a better understanding, this value will be called *window* (W). As a simple example, the statement window=3 corresponds to a mask given by a matrix of size 3x3. Hence, equation 2.5 becomes equation 2.6, where W is the window size.

$$w(x,y) = \frac{1}{n.m}\begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{bmatrix} \tag{2.5}$$

$$w(x,y) = \frac{1}{W^2}\begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{bmatrix} \tag{2.6}$$

Applying the mask defined by equation 2.6 in equation 2.4, a mean filter is applied to the input image $f(x,y)$.

Another important linear smoothing filter is the Gaussian filter. The idea behind the Gaussian filter consists of creating a mask $w(x,y)$ using a two dimensional Gaussian function, defined by equation 2.7. A visual example of this mask can be seen in figure 4.

$$w(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{2.7}$$

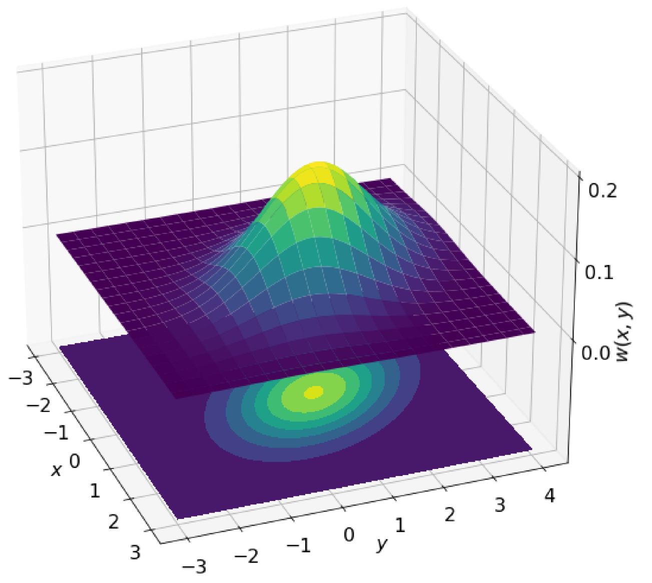


Figure 4: Gaussian kernel example

By observing figure 4 and equation 2.7, it is possible to see that for every pair (x, y), there is a corresponding $w(x, y)$. As (x, y) moves away from the center of the distribution $(0, 0)$, the value of $w(x, y)$ tends to 0. Therefore, the Gaussian filter can be considered an averaging filter, weighted by the distance from the neighbor to the central pixel. The further the neighbor is from the central pixel, the lower the weight of the pixel in the weighted sum. For a closed window size definition, which applies similarly to the mean filter, for the sake of simplicity, the approximation $W = 5\sigma$ was adopted (LOPES et al., 2019). Then, the equation for the Gaussian filter becomes 2.8.

$$w(x, y) = \frac{1}{2\pi \frac{W^2}{25}} e^{-\frac{(x^2+y^2)}{2\frac{W^2}{25}}} \tag{2.8}$$

2.2.2.2 SMOOTHING NONLINEAR FILTERS

For smoothing purposes, some filters based on Order-Statistics can be used. Among them, the main one is the median filter. The name itself is intuitive, as it replaces the value of pixel (x, y) with the median between the pixels involved around it by the mask. Median filters are quite popular because, for certain types of random noise, they provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size. It happens mainly for some specific types of noise, like *impulse noise*, also called *salt-and-pepper noise* (GONZALEZ; WOODS et al., 2002).

The median filter has a very simple implementation. First, the pixels enclosed by the mask are selected and their respective intensity values are ordered. As the windows used are odd sized, the nth largest value, that is the median value of the set of pixels obtained from mask, can be defined by equation 2.9. Practically, for a window of size 3, the median is the 5th largest value. When the window size is 5, the median is represented by the 13th and so on.

$$n = 1 + \frac{W^2 - 1}{2} \tag{2.9}$$

The basic concepts of filters applied in later sections were presented. The filtering definitions will also serve as a stimulus for the techniques that will be presented in the following section, which are based on Deep Learning techniques for solving problems involving images.

2.3 DEEP LEARNING APPROACHES

Machine-learning technology powers many aspects of modern society: from web searches to content filtering on social networks, recommendations on e-commerce websites, and so on. It is also increasingly present in consumer products, such as cameras, smartphones and automobiles (LECUN; BENGIO; HINTON, 2015).

Applying Machine Learning algorithms to tasks involving image processing is a widely used procedure. Preprocessing and extracting feature steps in the images are necessary before applying them to Machine Learning algorithms, though. It is important to notice that a specific knowledge is requested to apply those steps, since a deficiency of machine learning models may be a difficulty in working with data in its original format.

Feature learning, or representation learning, is a set of methods that allows a machine to learn the necessary features to perform a detection or classification task, obtaining this information directly from raw data. Deep Learning is based on multiple levels of representation learning collected by composing simple but non-linear modules that transform the representation at one level (starting with the raw input) into a representation at a higher and slightly more abstract level (LECUN; BENGIO; HINTON, 2015).

For images, the input layer is displayed in the form of a matrix with intensity values of each pixel. In the first layer, it is extracted representations of more general aspects, such as presence or absence of edges, orientation or components in the image. In later layers, the extracted information is combined and, thus, it generates more specific information, which can be related to identified patterns or the way objects in an image are correlated. This is a great advantage of Deep Learning: no specific knowledge is required to extract the features used for the machine learning process. These features are extracted directly from the data in its raw format.

The Convolutional Neural Network CNN is a type of technology based on deep learning networks of representation learning that uses a convolution operator to execute the feature extraction step. It was possible to see in section 2.2.2 that linear filtering applies a mask into images using a convolution operator in order to smooth images, reducing noise level in images. Other image transformations can also be applied using convolution operators, like edges detection using Laplacian operator (GONZALEZ; WOODS et al., 2002). The difference now is that several trainable layers of convolution can be applied to the image in several stages, selecting the ideal values of $w(x, y)$ using an optimization process, minimizing some error function defined in advance.

This type of technique is widely used in data processing and has been very successful in practical applications, such as object detection (ZOU et al., 2019), instance segmentation (BOLYA et al., 2019), face recognition (GUO; ZHANG, 2019) and others.

2.3.1 CONVOLUTIONAL NEURAL NETWORK COMPONENTS

In addition to the convolution layer, CNN has other types of layers that help in the process of extracting features. In figure 5, an example of a convolutional network is presented from the input of the image in its pixel format, represented in a matrix form, to the last step, which performs an operation of classification.

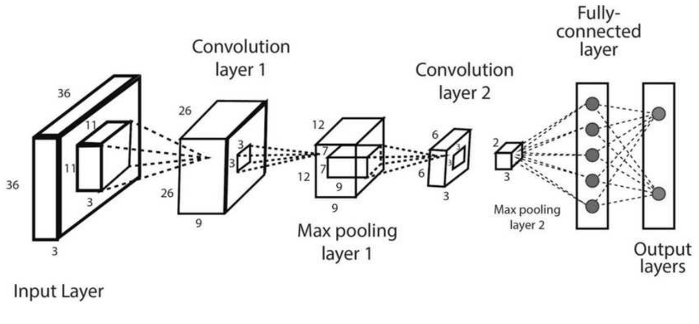


Figure 5: Convolutional Neural Network example (SARKER, 2021)

By looking into image of figure 5 it is possible to note that the image is first applied to a convolutional layer and, then, to a max-pooling layer. After max-pooling, the result components are again applied to a convolutional layer and, after that, by a max-pooling again. Finally, the result of the last one pooling is the input of an one-layer perceptron structure, with two neurons in the output.

Considering the architecture of the network shown in figure 5, the main components of a convolutional network will now be explained.

2.3.1.1 CONVOLUTIONAL LAYER

As mentioned in section 2.3, the convolutional layer is responsible for the extraction of image features by applying the convolutional operator. This layer involves the most part of the computer processing capacity and can be considered the most important layer of CNN.

The convolutional layer is composed by a series of trainable two-dimensional filters, which are applied to the image following the model given by equation 2.4. The main parameters that define a convolutional layer are number of masks, mask size and stride. The number of masks defines how many trainable filters will be applied on the image; each one is responsible for extracting a type of feature from the image. The mask size has been defined in section 2.2.2. Finally, stride defines the size of the displacement step that must be given by the mask along the sliding process applied to the input image in the convolution operation. When stride is 1, the mask is displaced by 1 pixel during the convolution process; when stride is 2, mask is displaced by 2 pixels and so on. An example of convolution layer application using a strides=2 is shown in figure 6. The colors indicate which pixels were considered by the convolution operator, associating the result of the output image. So, in this example, as the mask is 2x2 in size, each convolution operation acts in four different pixels and four operations are needed to complete the process.

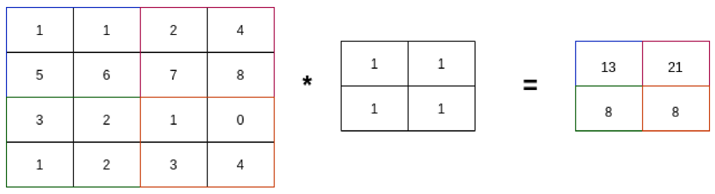


Figure 6: Convolutional layer application example

2.3.1.2 POOLING LAYER

The pooling layers are downsample layers in the data, reducing the spatial size of the arrays during the network learning process. With the application of pooling, the dimensionality of the data is reduced; therefore, the computational cost is also reduced. Another important point in the use of pooling is the reduction of overfitting (KARPATHY; LI; JOHNSON, 2017).

The pooling technique frequently used in CNN networks is max-pooling. Given the choice of kernel to be used, max-pooling consists of replacing the region values taken by the kernel by their maximum value. It is supposed to eliminate negligible values and reduce the size of the data representation (GIUSTI et al., 2013). An example of max-pooling with strides=2 is shown in the figure 7. As one can observe, the max-pooling operation preserves the maximum value of pixels, keeping important information by reducing image size. Other operations can also be used in pooling, such as average and minimum value.

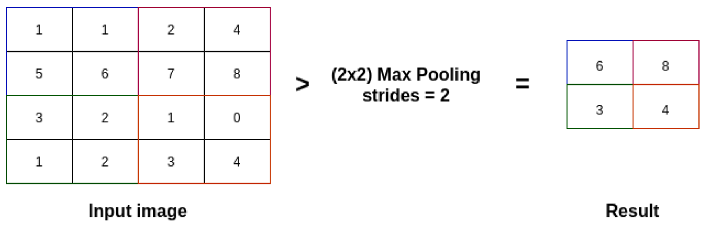


Figure 7: Pooling layer application example

2.3.1.3 FULLY CONNECTED LAYER

For a classification or regression task, the fully connected layer is the last step of a CNN, as it could be seen in figure 5. Its main objective is to use all the knowledge extracted through features by the previous layers to perform a classification or a regression task determined from the training data.

The meaning of the term ”fully connected” is due to the fact that each neuron in the previous layer is connected to all neurons in the next one, adding an output layer that brings a number of neurons equivalent to the number of classes present in the respective experiment. Final classification is performed according to some type of activation function (KARN, 2016). This approach is the same as one that is used by Multi-Layer Perceptron (RAMCHOUN et al., 2016). An example of fully connected layer is shown in figure 8. The learned features are taken to the fully connected layer. Its weighs shall be computed by a training process to make it useful to perform the intended classification or regression task.

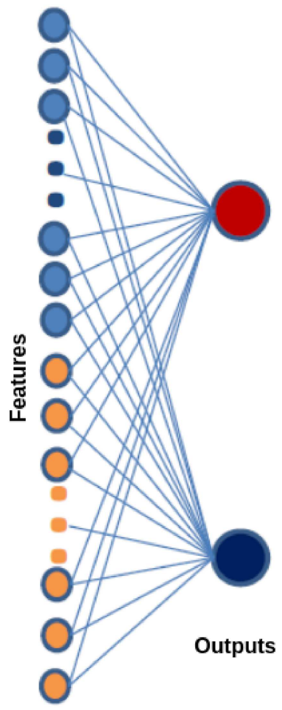


Figure 8: Fully connected layer example

2.3.1.4 ACTIVATION FUNCTIONS

Activation functions are used in neural networks to compute the weighted sum of inputs and biases which will define whether a neuron will be activated or not. When no activation function is used, a neural network is simply a series of linear operations applied on data, and for CNN this is not so different. Some of commonly activation functions used in CNN are Sigmoid, Hyperbolic tangent and Rectified Linear Units (ReLU), described by equations 2.10, 2.11 and 2.12, respectively. In figure 9 it is possible to see them together.

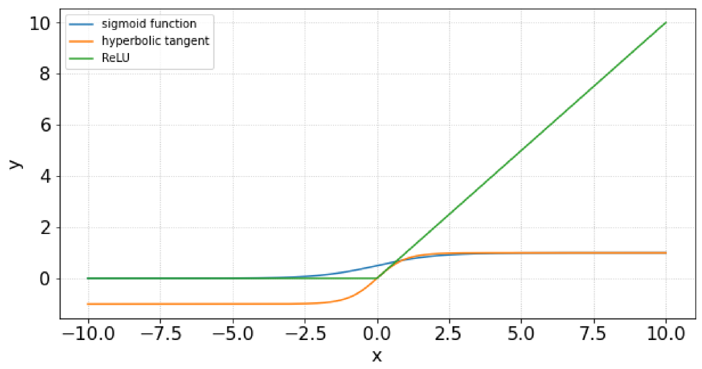


Figure 9: Activation functions example

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.10}$$

$$\tanh x = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.11}$$

$$f(x) = max(0, x) \tag{2.12}$$

For CNN, ReLU is widely used, since activation functions such as Sigmoid and Hyperbolic tangent can damage the learning process, causing known problems such as gradient explosion and gradient vanishing. The backpropagation process leads to very small or very large values of the derivatives. Once CNN has many layers, when applying these activation functions the gradient value tends to increase a lot (explosion) or decrease a lot (vanishing) (HANIN, 2018).

2.3.1.5 OTHER USED OPERATIONS

Some of the main layers and operations involving CNN were presented. This section aims to point out some processes that were also used in this work and help to optimize the learning process of networks as well.

- Dropout: Dropout (SRIVASTAVA et al., 2014) was proposed in order to reduce overfitting during the neural networks training process. This process consists in deactivating some neurons in the network during the learning process. Such deactivation is done from a probability p of a neuron being deactivated. The higher the value of p, the greater the number of neurons deactivated during training. This procedure forces the network not to be dependent on one neuron or on the combination of several neurons during the training process.

- Batch Normalization: Batch Normalization is a popular and effective technique that consistently accelerates CNN training convergence (IOFFE; SZEGEDY, 2015). Batch normalization is applied to individual layers and works as follows: In each training iteration, it first normalizes the inputs (from batch normalization) by subtracting their mean and dividing it by their standard deviation. Then it is applied a scale coefficient and a scale offset. One of parameters of this operation is the size of batches; so, in addition to applying the normalization in batches, one must also be concerned with the size of the batches.

The inputs needed to understand convolutional networks were shown in this section. The examples illustrate the simpler CNN operation. In the next section, a more complex class of convolutional networks will be discussed, which was used in this work: the Fully Convolutional Network (FCN), used for semantic segmentation.

2.3.2 THE FULLY CONVOLUTIONAL NETWORK

The FCN was created in 2015 (LONG; SHELHAMER; DARRELL, 2015) and is similar to CNN. The main difference is that in FCN the output layer, generally used for simple classification or regression tasks, is replaced by a convolution layer, which is responsible for classifying each pixel of the image individually. The idea of this approach is to determine which elements exist in the image in addition to their location. An example of this difference is shown in figure 10. For CNN application, the output returns the found object of an image. On the other hand, FCN returns, at the pixel level, information about the object, such as which class they belong to, and also which pixel groups make up a given object.

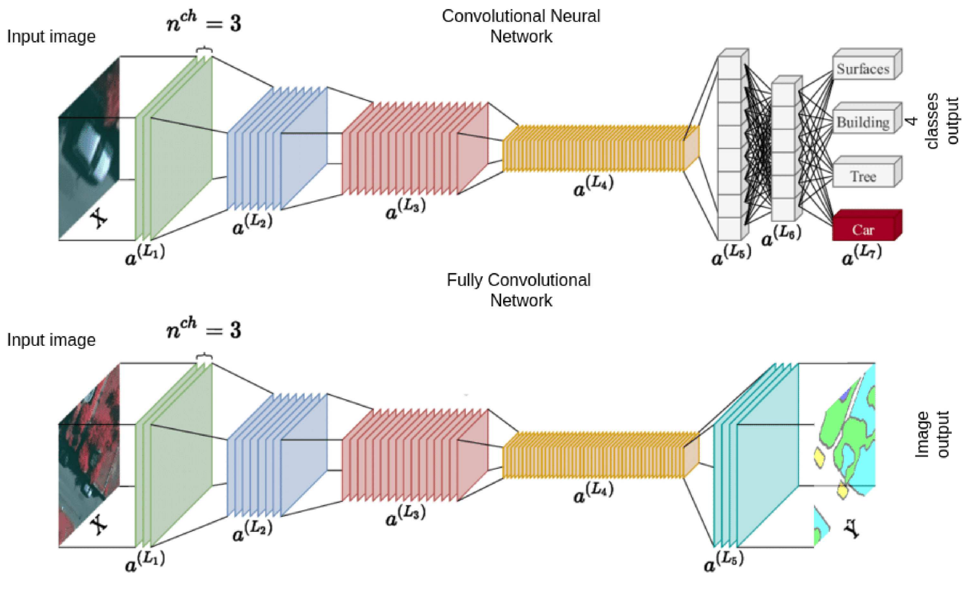


Figure 10: FCN and CNN comparison example (OLIVEIRA et al., 2021)

The process of turning a set of extracted representations into images for a classification at the pixel level is done in a very similar way to the process of transforming an image into a set of representations. Nonetheless, the operations presented in 2.3.1 will be applied in the opposite way, trying to create an image from a representation layer. This process is know as transposed convolution (ZHOU et al., 2020).

The main idea of transposed convolution is to enrich the smaller layers with information by upsampling them, increasing their size until reaching the same size as that of the input image.

For restoring values, some operations, like interpolation, nearest neighbors and max-unpooling operations can be applied. An example of upsampling nearest neighbors is shown in figure 11.

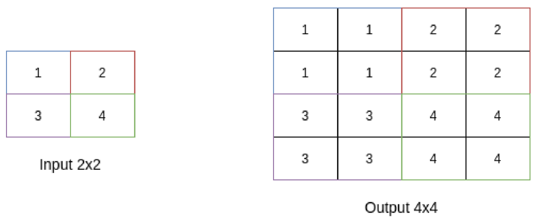


Figure 11: Nearest neighbors upsampling example

One of the problems with this type of network architecture is that when propagating through multiple convolutional layers and alternating pooling layers, the resolution of feature representation output is reduced. Therefore, the estimates obtained by the FCN will often be in low resolution, resulting in relatively imprecise object boundaries (NASR-ESFAHANI et al., 2019).

Knowing the basics of CNN and FCN, it is now possible to present the convolutional neural network used in work, the U-Net.

2.3.2.1 U-NET FULLY CONVOLUTIONAL NETWORK

U-Net (RONNEBERGER; FISCHER; BROX, 2015) is a FCN with the objective of performing the segmentation of biomedical images. The authors designed it to work with a small number of images of training and to obtain a high precision in the segmentation, a case that is very similar to the one approached in this work.

One of the ways to realize pixel-wise classification tasks was developed by training a network in a sliding-window setup, to predict the class label of each pixel by providing a local region (patch) around that pixel as input.

Some drawbacks of that approach can be punctuated: in addition to the large number of images required for training the network, running predictions for each patch in a sliding window slows the network down. It generates great redundancy involved in predicting the same pixel several times as well. In addition, it is also the point about the trade-off between accuracy of the location of objects in the image and the use of context. Larger patches require more max-pooling layers, which end up reducing location accuracy, while smaller patches decrease the context covered by the network (RONNEBERGER; FISCHER; BROX, 2015).

The U-Net proposed architecture is shown in figure 12. It is possible to see that U-Net do not use any fully connected layers, only convolutional ones. This kind of architecture is called Encoder-Decoder (SKEIKA et al., 2019) and is composed by two main steps. The first, named *contract*, builds the representations based on the input image by using convolutional and pooling layers. The second step, called *expand*, applies upsampling and concatenation, starting from the information of the correspondent layer in contract step in order to have an image output that allows pixel-wise comparison between input and output.

The name U-Net is given by the symmetry between the layers of the network that form a ”U” letter in its architecture, since the operations applied in the contraction stage are applied transposedly in the expansion stage.

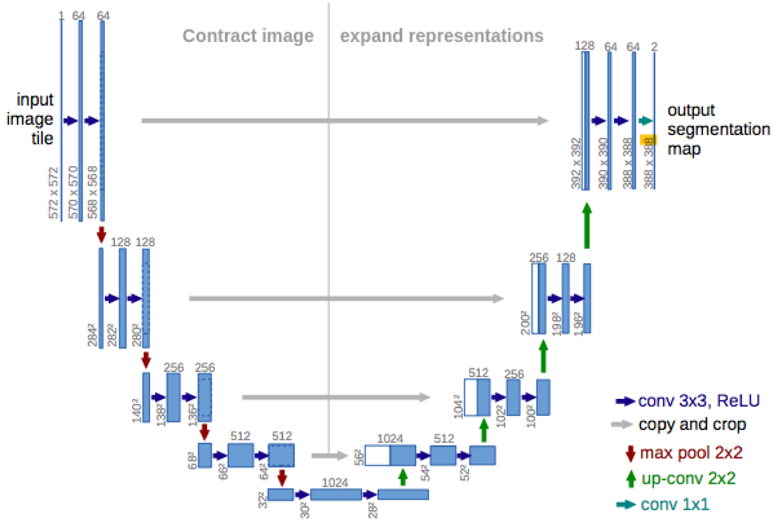


Figure 12: U-Net architecture (RONNEBERGER; FISCHER; BROX, 2015)

In the *contracting* path, a combination between convolution and max-pooling operations is applied, starting by the input image, and for each stage ($conv + max-pool$), the number of convolution layers is multiplied by 2. The inverse step is done for the *expansive* path by using a combination between convolution and transposed convolution (*upconv*) operations, concatenating them with information of the correspondent layer from the contracting step. The contracted layers have the high resolution information about image input, thus improving the learning of representations in the following layers. In the last step, a 1x1 convolutional layer is applied, preceded by a Softmax activation layer for predictions.

3 CYGNO

This section is dedicated to describe the CYGNO experiment, showing its investigation objectives and how its apparatus was used in this work. The main characteristics of the analyzed data will also be presented.

3.1 CYGNO EXPERIMENT

An essential requirement for experiments interested in understanding the mysteries regarding dark matter is to know in details their background radiation, both internal and external to the detector apparatus. For the latter, environment and cosmic radiations can produce low energy events, below 30 keV, making it a little complicated task to discriminate background noise from events of interest (BAUDIS, 2012). For this reason, techniques capable of discriminating nuclear recoils, generally related to the signal of interest, from electronic recoils, generated by background radiation, are of upmost importance for Dark Matter experiments.

In order to minimize this background effect, such experiments are usually located underground; they also make use of materials of excellent radio-purity. Some of them have the ability to discriminate a nuclear recoil from other interactions that may happen even underground.

3.2 DM SEARCHES WITH CYGNO

The Milky Way presents a rotation movement around its center in a clockwise direction. This motion presents irregularities when compared to that one which is predicted based on the total visible mass (through stars, gas, and other components). It was noted that the most distant regions of the galaxy's orbit rotate at higher speeds than predicted, based on Kepler's Law. Therefore, the conclusion is that the rotational speed does not necessarily decrease with distance, but remains constant from the innermost to the outermost of the Galactic disk. (SCHNEIDER, 2014).

The rotation curve describes the rotation speed of stars in the galaxy as a function of their distance from the center. Such velocity is related to the amount of matter found within this orbit, so it is possible to infer the mass of the galaxy through the movement of its components. Thus, the velocity in its external parts is greater than expected, implying the existence of a significant amount of matter beyond the unobservable part of the Galactic disk. Therefore, it is believed that this difference is due to the existence of dark matter, which is directly undetectable and whose nature is unknown. (PASACHOFF; FILIPPENKO, 2013).

The Sun orbits the galactic center with a speed of approximately 220 kms^{-1}, and its velocity vector is pointed to the Cygnus constellation. The signal derived from the interaction with WIMPs scattering expected in the experiment's detector comes from the relative motion of the Earth in relation to the galactic halo and Dark Matter (DM), apparently coming from the constellation Cygnus. Then, determining the direction from which the dark matter particle comes from space can provide a correlation with an astrophysical source that does not resemble any noise background; thus, it gives the necessary information for signal identification of dark matter. In addition, measuring the directionality of these particles can also help to discriminate different models of dark matter (KNIRCK et al., 2018; IRASTORZA; GARCÍA, 2012) and furnish more information about the properties of WIMPs, which would not be possible with non-directional detectors.

In this way, the CYGNO collaboration proposes a different approach, using a high resolution TPC detector, which is filled with Helium or Fluorine so it increases its sensitivity to WIMPs at the same time it preserves information about their directionality and reject background noise, even if it is a low energy event.

TCP is a particle detector composed of a cage filled with gas or liquid, surrounded by electric and magnetic fields. When a particle passes through the detector and has enough energy to ionize it, electrons and ions are carried to the anode and cathode by the action of this electric field, making it possible to infer the particle's released energy from the charge measured at the anode.

An advantage of using Helium to fill TPC is the possibility of working at atmospheric pressure, which, in addition to reducing costs with production of equipment that withstand different pressures, also guarantee a reasonable ratio between volume and mass. In order to increase the sensitivity of the detector, an amplification step is used, based on Micro Pattern Gas Detector (MPGD).

MPGD consists in microelectronic structures with very small distances (lower than 1 mm) between its cathode and anode. CYGNO experiment uses GEM technology (SAULI, 2016a), introduced as a pre-amplification step for the detector.

The aforementioned characteristics make the experiment able to explore new cases of particle physics that need a high ability of discriminating between nuclear recoil and other particles and collecting information about the direction of these particles.

This section is divided into two parts: the first one explains the construction of the detector, and its mainly used components. The second highlights some characteristics of the datasets used in this work, acquired with the LEMOn prototype.

3.3 CYGNO DETECTOR

The CYGNO experiment aims to develop a MPGD detector based on TPC with a triple Gas Electron Multiplier (GEM) and a Scientific Complementary metal-oxide-semiconductor (sCMOS) optical readout that delivers a high precision 3D tracking, sensitive to the direction of the recoiling nuclei and the electrons for Dark Matter searches at low (1-10GeV) WIMP masses down to the Neutrino Floor. Figure 13 shows a 3D drawing of the CYGNO detector.

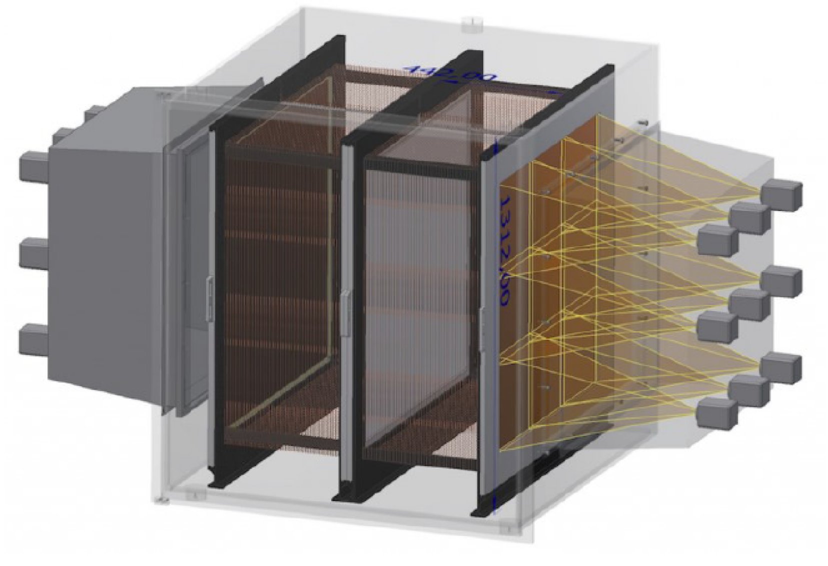


Figure 13: CYGNO detector

The project must go through some phases so the final objetive is achieved. They can be divided as it follows:

PHASE-0 is the current one and it is focused on the detector development, using prototypes as a test platform to understand the detector's characteristics; during this phase, many tools are under development in order to simulate and analyse the acquired data;

PHASE-1 aims to built the 1 m^3 demonstrator;

PHASE-2 is expected to develop a 30-100 m^3 detector.

The CYGNO collaboration has been testing different prototypes, such as ORANGE (MARAFINI et al., 2017; ANTOCHI et al., 2018), NITEC (BARACCHINI et al., 2018) and Large Elliptical Module (LEMOn) (PINCI et al., 2017; Mazzitelli et al., 2017; PINCI et al., 2018; ANTOCHI et al., 2020), by varying the radioactive sources and some of the general operating conditions to define the best adjustments for the project and for the development of the final detector's architecture.

The idea behind each one of the developed prototypes is the same: a gas filled acrylic box with at least one clear side to allow the camera to take pictures into the sensitive area, and a drift field to lead the electrons to the camera side in the box, where a Triple-GEM (SAULI, 2016b) is placed. Such components are used to amplify the signal produced by ionization process. The camera is put out of the box to take pictures of the produced light signal.

So far, the collaboration works with four prototypes, being them ORANGE, MANGO, LEMOn and LIME. The main difference between these prototypes is the volume drift. The prototype used in this work is LEMOn, which is discussed in the next section. The most recent prototype is LIME, which has a volume drift of about 1/9 of what the final detector is supposed to have ($1m^3$).

3.4 LEMON PROTOTYPE

One of the most recent CYGNO Experiment's prototypes is LEMOn, and the databases used in the present work are based on this detector, both simulated and extracted in practice. The LEMOn detector, as shown in figure 14, is composed of an elliptical field cage ($20 \times 20 \times 24$ cm^3) which has a 7 liter active drift volume inside and is closed by a 20×24 cm^2 Triple GEM structure that amplifies the signal coming from the sensitive volume. Then, the photons produced in the GEM are readout by an Orca Flash 4 CMOS-based camera placed at a distance of $52.5cm$ (i.e. 21 Focal Length, FL).

The gas used inside the detector is the mixture He/CF_4 in the proportion of 60/40. One electric field is applied to the TPC drift volume, and another one, between TPC drift volume and GEMs. They are called drift field (E_d) and transfer field (E_t), respectively.

The regular operational settings of the detector, as used in this work, are: $E_d =$ 500 V/cm, $E_t = 2.5$ kV/cm, and a voltage difference across the GEM sides (V_{GEM}) of 460V.

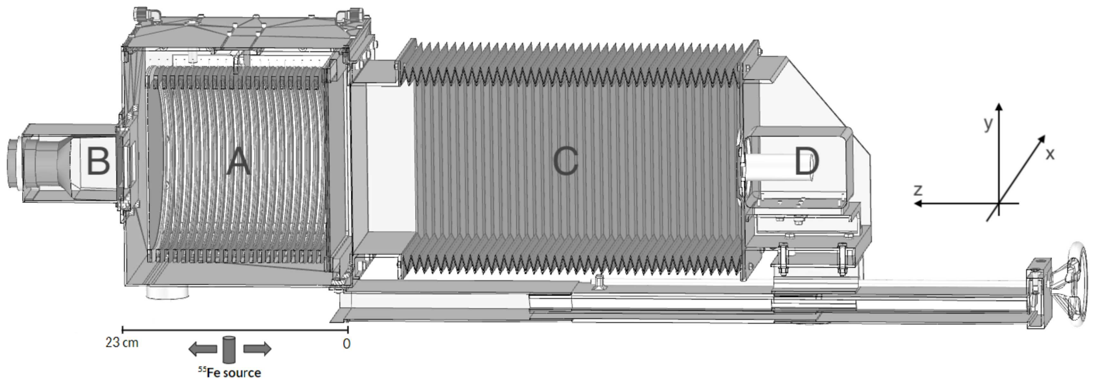


Figure 14: Drawing of the experimental setup. In particular, the elliptical field cage, closed on one side by the triple-GEM structure and on the other side by a semitransparent cathode (A); the Photomultiplier Tube (PMT) (B); dark volume of adaptive distance between the GEM and the CMOS camera (C); and the CMOS camera with its visible lens (D).

3.5 LEMON DATASETS

In the developed work, it was used two datasets acquired with the LEMOn detector: (1) images simulated by the collaboration, aiming to reproduce a supervised environment for the experiment and (2) real data obtained with the apparatus in operation, in two scenarios. The first one was prepared with the obstructed camera for the acquisition of noise images with simulation purposes; the second one was prepared with no camera obstruction, reflecting the reality of the detector.

3.5.1 SIMULATION DATASET

The simulation images were produced based on the LEMOn prototype, simulating the interactions of the particles in the detector using the Geant4 (INCERTI et al., 2010) software, commonly used for this purpose. As the experiment was developed to detect events for energies below 60 keV, the simulation generated events with the following energies: 1, 3, 6, 10, 30 and 60 keV. Each of the energy packs has 100 electronic recoil images and 100 nuclear recoil images, with a 2048x2048 size image, simulating the process performed by the photosensor, including digitization.

In figure 15, the probability density function of the intensity of the pixels for the simulated energy packets is displayed for each energy and particle, considering only activated pixels. A Kernel Density Estimator (KDE) was used in order to plot those distributions. Analyzing them, it is possible to see that, for energies below 10 keV, the two particles present very similar pixel intensity distributions. With the increase in energy, the nuclear recoils tend to present a long tail distribution, while the electronic recoils start to concentrate a good part of their activated pixels in lower intensity values. As they have the same energy, the track format should be different between them.

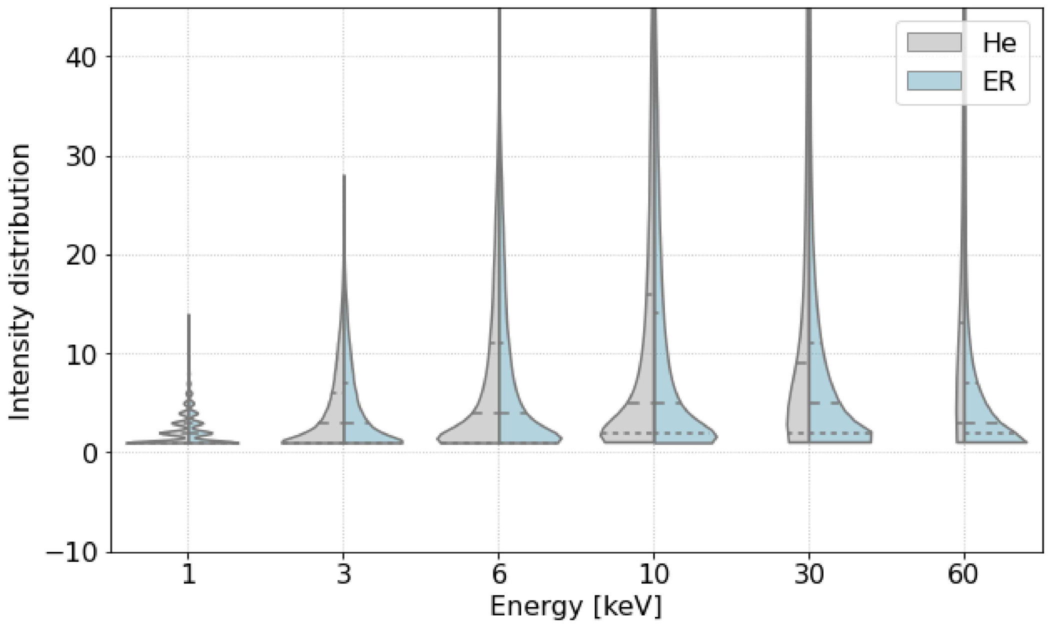


Figure 15: Probability density function of intensity, considering only activate pixels. For each energy (x-axis), the probability density function is shown along y-axis. It is considered Nuclear recoil (He), and Electron recoil (ER).

Such behavior can be shown through the figures 16, 17, 18, 19, 20, 21. In the first element, we have a sample of nuclear recoil, followed by a sample of electronic recoil of the same energy, with a zoom in the region of interest. The last element displays the distribution of the sum of all pixels for each energy, segmented by particle type.

Looking at the shape of the traces left by the particles in the detector, one can see that the nuclear recoils always have a rounded shape, regardless of the energy level. For the distribution of the sum of pixels of all images in the package, when compared to electronic recoils, it presents greater variance in its distribution. For the shape of the electronic recoils, from 10 keV onwards, it can be seen that the traces left by them in the detector have a different shape, but the distribution of the sum of the intensities of the pixels presents a very small variability, mainly for lower energies.

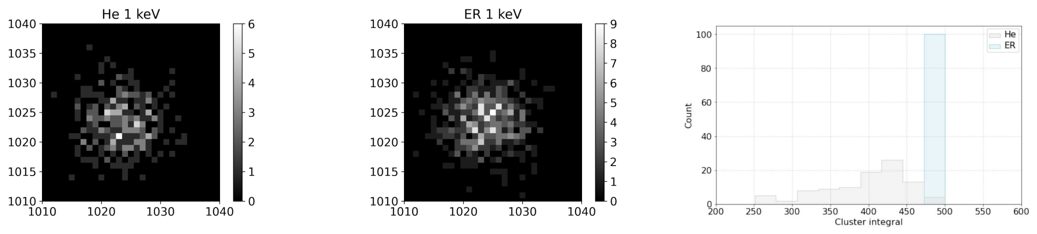


Figure 16: Example 1 keV simulation

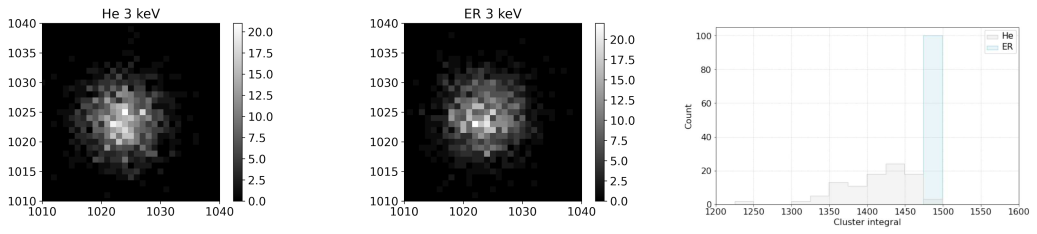


Figure 17: Example 3 keV simulation

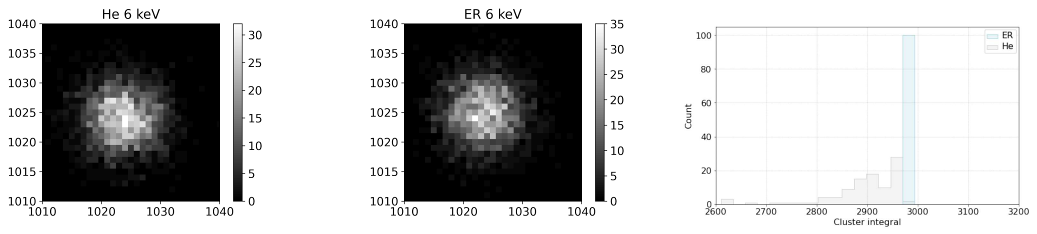


Figure 18: Example 6 keV simulation

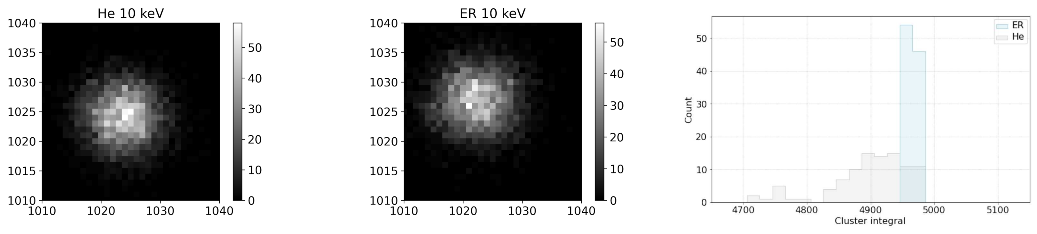


Figure 19: Example 10 keV simulation

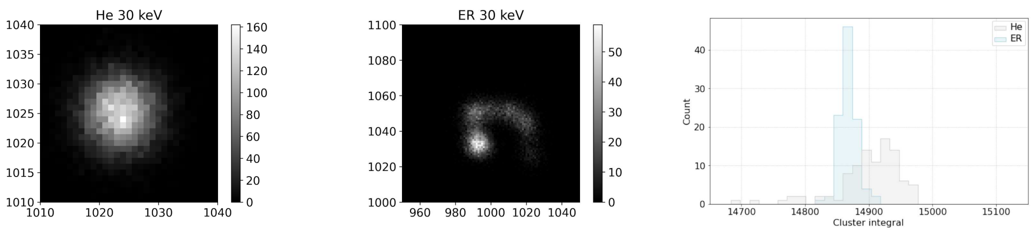


Figure 20: Example 30 keV simulation

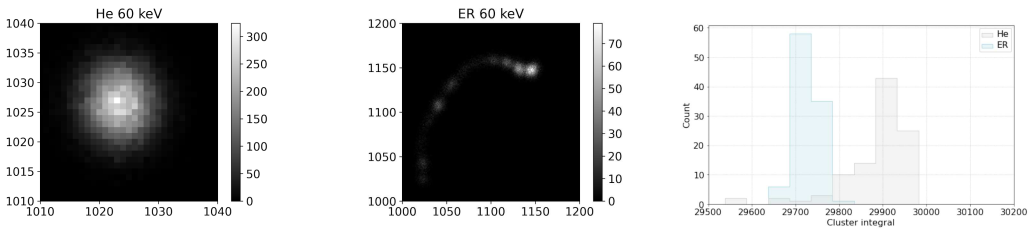


Figure 21: Example 60 keV simulation

Another important characteristic to take into consideration is given by the number of pixels activated by the simulated events. Figures 22 and 23 show the probability density function built using KDE, regarding the number of activated pixels on each image for NR and ER, respectively. In order to provide a better visualization, ER results were divided into two images.

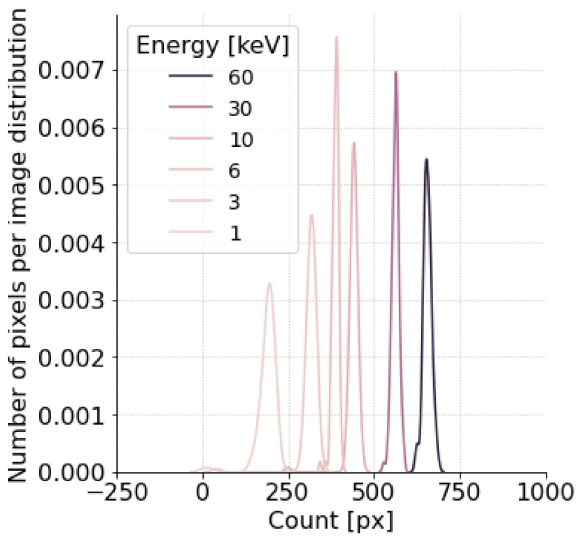


Figure 22: Probability density function of the number of activate pixels per image, considering nuclear recoil particles. It is possible to note that the expected value is increased along with energy, but the shape of the function is kept.

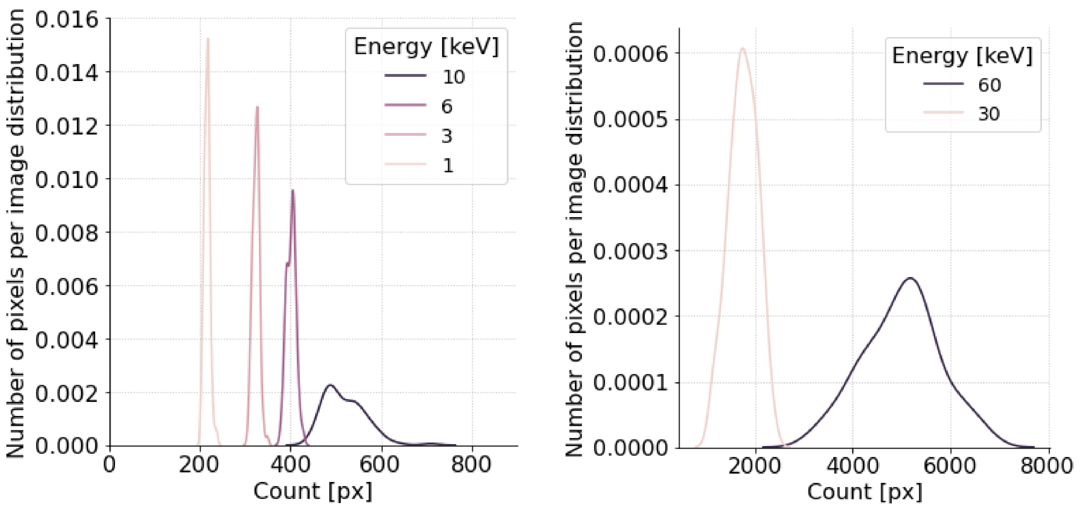


Figure 23: Probability density function of the number of activate pixels per image, considering electron recoil particles. It is possible to note that the shape changes when energy of events are increased.

For nuclear recoil, there is a small impact on the number of pixels that make up the particle tracks and, also, on their dispersion along the different simulated energies. For nuclear recoil, this effect is more significant than those, for example, 30 keV events that have a much higher number of pixels when compared to particles with lower energy; in addition, the dispersion of the distributions increases considerably for events with energy above 6 keV.

It is also important to point out that in the most optimistic case, the experiment images, with a size of 2048×2048, it would have around 5000 activated pixels. This represents about 0.11% of pixels that contain relevant information. Such information is important when defining the evaluation metrics, which are addressed in the next section.

3.5.2 REAL DATASET

For the data collected by the LEMOn prototype, two types of situation were used:

- Noise dataset acquisition: Formed by lowering down V_{GEM} to a value where there is no multiplication process, so it records only electronic noise. This noise will be used to create a noise simulator using Monte Carlo method. The number of samples acquired was 6478. A sample of this acquisition is shown in figure 24

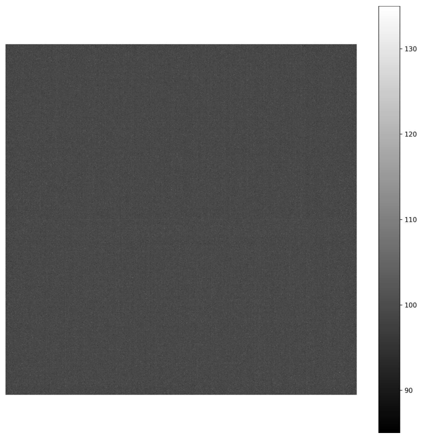


Figure 24: Noise dataset acquisition sample

- Real data acquisition: Formed by configuring the detector to work with its operational settings and inserting a ^{55}Fe source next to detector drift volume. This radioactive source is commonly used for low energy tests and calibrations (PHAN; LEE; LOOMBA, 2020; SANGIORGIO et al., 2013) due to the fact that it emits particles around 5.9 keV with low background events. Figure 25 shows an LEMOn acquired image in such conditions. On the right side, we have an example of a ^{55}Fe spot, which will be used in the evaluation section to analyze real data.

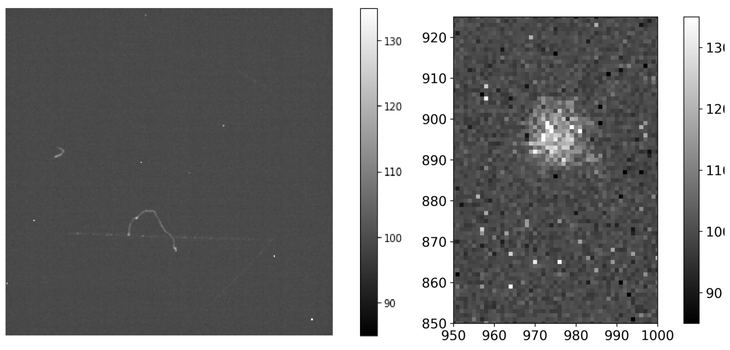


Figure 25: An example of image acquired by LEMOn prototype when ^{55}Fe is on. On the left side figure, a 2048x2048 image containing events identified by the detector. On the right side figure, a ^{55}Fe spot shown after zooming in the image.

4 METHODOLOGY

In this section, the methodology developed to evaluate the impact of filtering as a preprocessing technique on images from the LEMOn detector is discussed. This process can be divided into two parts. The main objective of the first part is to select optimized parameters for the proposed filters. Such selection was made using simulated images provided by the collaboration. Once the output is known in this case, it is possible to use an adequate efficiency evaluation metric to measure the performance of the algorithms. For the second part, considering the optimized filter parameters have been chosen, the filter performances in the reconstruction algorithm are evaluated, comparing output variables such as energy and processing time.

The main objective of the proposed analysis is to verify the impact of different filters that could be used to replace a computationally heavy part of the algorithm currently in use by the CYGNO Experiment.

4.1 *CYGNO EXPERIMENT RECONSTRUCTION ALGORITHM OVERVIEW*

The version of the reconstruction algorithm used in this work is shown in the flowchart of figure 26.

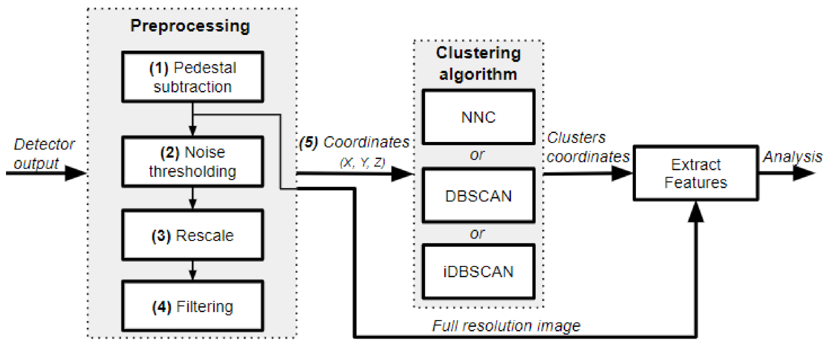


Figure 26: Reconstruction algorithm flowchart (BARACCHINI et al., 2020).

The detector output images are sent to a set of preprocessing algorithms that are described below:

- Pedestal subtraction: The output image of the detector can be described from the composition of the particle interaction result inside the detector and the electronic noise contribution. The latest is originated during the capture of events by the sCMOS sensor. The purpose of the pedestal removal is to subtract from the image the average effect of the electronic noise pixel by pixel. To make it possible, the noise acquisition described in section 3.5.2 was used in order to get the average value of each pixel value.
- Noise thresholding: Once the expected noise value has been removed for each pixel, the next step consists of a threshold on the same pixels based on the noise information extracted from images with the camera obstructed. This threshold is based on the camera electronic noise standard deviation. Its main objective is to discard the pixels with intensity values smaller than the standard deviation times n_σ, where n_σ is defined as described in section 1.3.
- Rescale: Also known as rebinning, this process consists of re-scaling the image in order to reduce its size. The scale factor used by the algorithm is 4, so an input image of size 2048x2048 becomes a 512x512 image. The main objective of this procedure is to reduce the number of pixels that will be sent to the clustering algorithm, aiming to reduce the computational cost of the whole process.
- Filtering: After rescaling, a median filter is applied. This process replaces the pixel value with a median calculated using its neighbors. The filter window is 4x4, which gives 15 neighbors. Input value of pixels are used to fill edges, in order to perform median operator.

  In this step, is also applied a noise reduction algorithm in order to remove sparse pixels in image.

After the preprocessing step, the filtered pixels, together with their respective intensity values, are sent to a clustering algorithm to form clusters of particles. The algorithm used in this work is the intensity-based DBSCAN (iDBSCAN), which presents better results compared to other algorithms, such as Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Nearest Neighbor Clustering (NNC) (BARACCHINI et al., 2020).

Finally, once the clusters are extracted, their features are calculated to understand the interactions that took place in the detector. The main feature used in this work is the energy, which is the sum of the pixels' intensities belonging to the cluster found in the clustering step. Later on, this value can be converted into keV using a calibration factor.

4.2 PROPOSED RECONSTRUCTION ALGORITHM

With a better understanding about the energy reconstruction process, it is possible to think of improving strategies for the algorithm. The purpose of the present work is, first, to propose a different way from the one used for pixel selection and, in addition, to develop a benchmark environment to verify the advantages and disadvantages of each approach.

The idea initially proposed is based on the optimization of the selection of pixels after the threshold step. It is expected that the track's real energy could be obtained only when pixels that are part of the particle are sent to the clustering algorithm. Therefore, improving this process impacts the reconstruction process.

When analyzing the image formation, from the particle interaction inside the detector to the output generated by the camera, the biggest problem of the pixel selection step is related to the electronic noise introduced by the image capture process. The inserted noise varies the intensity value of the pixels, directly affecting the value of the reconstructed energy. This variation can be characterized in the frequency domain by components located at high frequencies, which can be attenuated by image filtering techniques presented in section 2.2.

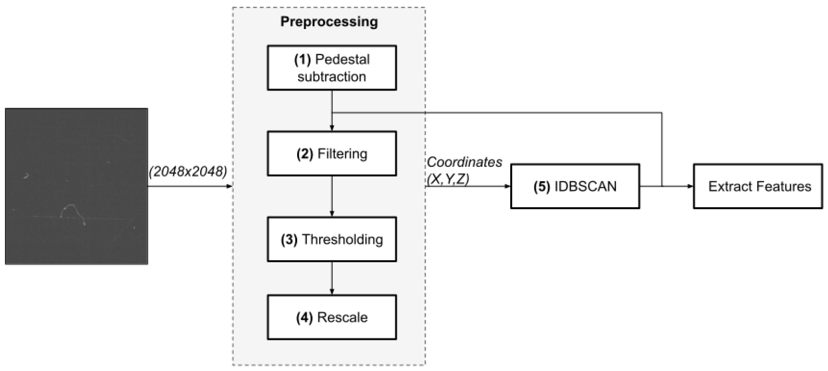


Figure 27: Proposed reconstruction algorithm flowchart

The proposed algorithm, including a filtering step to attenuate the electronic noise coming from the camera, is shown in figure 27. The main changes in relation to the version currently used by the experiment are:

- Filtering after pedestal subtraction: The pedestal subtraction was kept and a filtering step is applied to the image with mean noise value of pixels subtracted, instead of a cut based on standard deviation noise.
- Threshold after filtering: Filtering operation can change the image intensity value domain, so the threshold will be applied after the filtering step.
- All steps after the rescale were removed: after the threshold process, the last step of the preprocessing is the rescale.

In order to simplify the analysis process, none of the clustering algorithm parameters were changed, since the initial premise was that the optimization of the pixel selection process can generate improvements in the energy reconstruction process.

Once the use of filters to improve energy reconstruction has been proposed, it is necessary to define a coherent way to select the parameters of filters and evaluate the performance of these algorithms together with the algorithm currently used by the experiment.

In the next section, the development of the filter parameter selection environment and the evaluation of the algorithms are discussed.

4.3 A PIXEL SELECTION EVALUATION ENVIRONMENT

The proposed environment to compare pixel selection techniques for CYGNO reconstruction algorithm is shown in the figure 28. This setup will be used to perform algorithm evaluations and can also be used to choose optimal parameters for these algorithms.

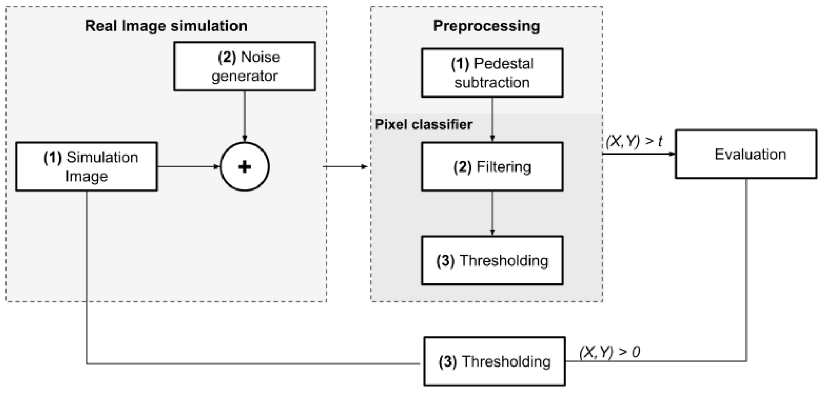


Figure 28: Simulation environment for comparison of pixel selection techniques.

The fundamental condition for this proposal to work is knowing the ground-truth pixels in the input image, since this answer is used to determine the efficiency. For this reason, simulation images, presented in 3.5.1, will be used to define the algorithms and their parameters.

4.3.1 REAL IMAGE SIMULATION

The track generator's main task is to convert the interactions of the detected particles into images like seen in reality. This step is very important because the more accurate the simulation is, the more precise the proposed algorithm will be. Some attempts to model these interactions have been developed in (LOPES et al., 2019). However, for this work, the simulation developed by LEMOn detector was used. For evaluation purposes, a threshold value of 0 is applied to the generated image in order to define which pixels should be detected in the case of a perfect selection. As an example, on the right side of figure 29 it is shown a 60 keV electron recoil image simulated by Geant4. On the right side of the same figure all pixels that have a threshold greater than 0, and that should be identified by the preprocessing step, are shown.

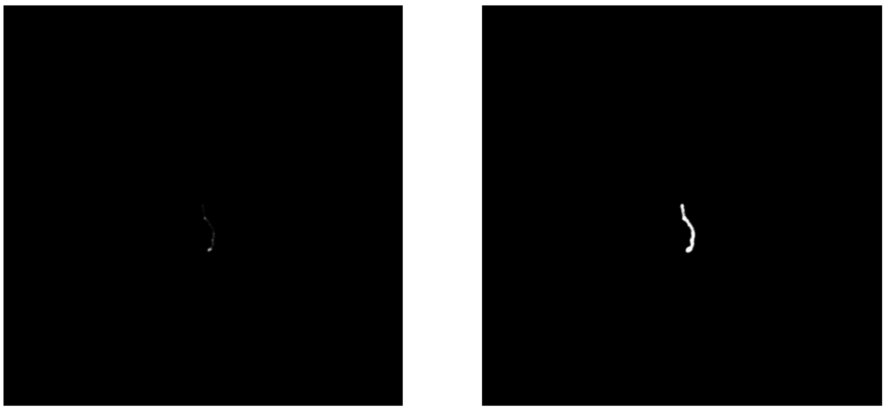


Figure 29: An example of image generated by Geant4. On the left side figure, it is shown a 2048x2048 image used as input in the proposed environment. On the right figure, it is shown the same image after applying a threshold equals to 0 to it.

4.3.2 NOISE GENERATION

In order to have an image closer to the reality of the detector, we inserted an electronic noise simulator in this environment, trying to obtain camera effects. A real image can be defined by equation 4.1, where $I_{real}(x, y)$ is the image after noise insertion, $I_{truth}(x, y)$ is the image after track generator and $\eta(x, y)$ is the noise image generated.

$$I_{real}(x, y) = I_{truth}(x, y) + \eta(x, y) \tag{4.1}$$

An image data-set was collected using the noise acquisition described in section 3.5.2. Assuming that the intensity of the pixel probability distribution for the noise is independent for each pixel, a Monte Carlo simulation was used to create a random noise generator with a probability distribution close to real.

An example of this step can be observed in the figure 30. On the left, it is possible to see the image obtained by the simulation step; and on the right, there is the same image after passing by noise insertion process.

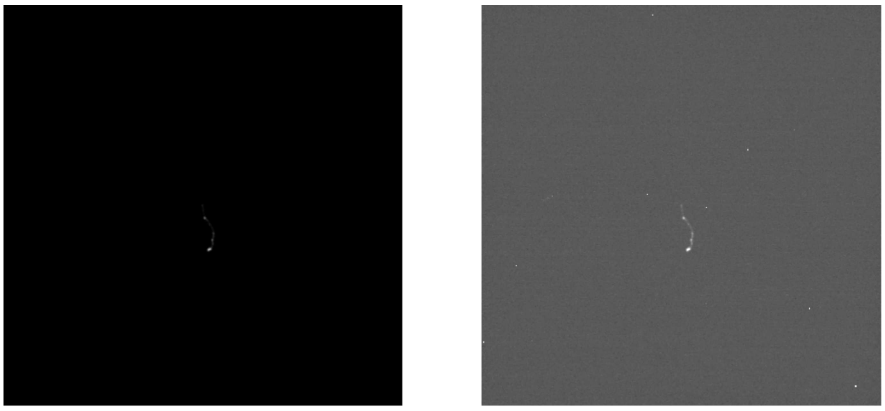


Figure 30: Example of 60 keV electron recoil event after noise insertion. On the left, an image obtained from simulation that, after being added to a noise image, results in the figure on the right

4.3.3 PEDESTAL SUBTRACTION

After inserting the simulated electronic noise in the images, the pedestal is removed using the mean value of noise for each pixel; it was calculated using a different noise sample. An image called *pedmap*, containing these values for each pixel, is used in this procedure.

The main idea behind this step is trying to take to 0 the value of the intensity on pixels that were not activated by particles in the detector. As for the pixels that were activated, it was removed the noise portion of its value, trying to keep it as close as possible to the value that it actually should have had if there was no noise.

The pedestal subtraction process can be observed in figure 31. On the left side, it is shown the histogram of an image before pedestal subtraction and on the right after the subtraction process. It is possible to notice that most pixels have an intensity value greater than 0. On the other hand, when the pedestal is subtracted, most of the pixels presents values closer to 0, removing the mean value of noise influence.

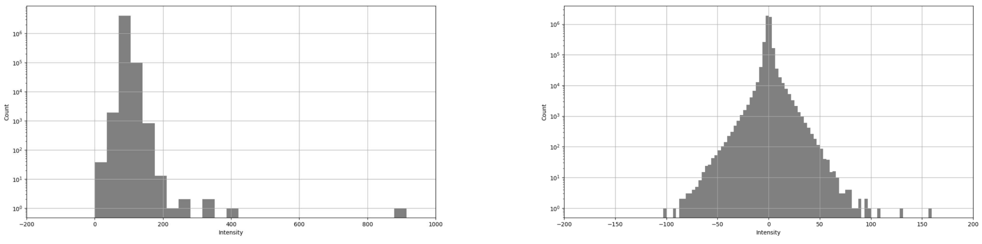


Figure 31: Intensity histogram of an image event. On the left, the image before pedestal subtraction and, on the right, after this operation. It is possible to note that most part of pixel values move to around 0.

An example of image after pedestal subtraction is shown on figure 32. When compared this image to the image of figure 30, it's possible to see that some pixel spikes disappear. It happens because this is a camera effect, bring some specific pixels to very high intensity value. When noise is acquired, these pixels presents these effect that can be got by mean value and removed after pedestal subtraction process.

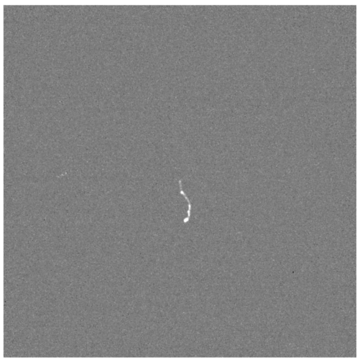


Figure 32: Example of image after pedestal subtraction

4.3.4 FILTERING

The filtering process consists of applying digital filters to the images after noise insertion, aiming to improve the pixel selection process, thus improving the separation of the distributions of pixels that are considered signal from the background pixels and facilitating the threshold step. The filters selected to evaluate the proposed work were *smoothing* filters and a deep learning based pixel selection, as described in section 2.2.

For smoothing, some examples of filter application effect can be seen in the figures 33, 34 and 35. It is possible to see the effect of filtering in simulation images by changing the window parameter for each used filter. The higher the window value, the more the image is blurred, removing the high variations (similar to high frequencies) in intensities caused by noise. On the other hand, the characteristics of the particles are also lost, since the tracks get somewhat blurred, and it is necessary to find an ideal point for this trade-off.

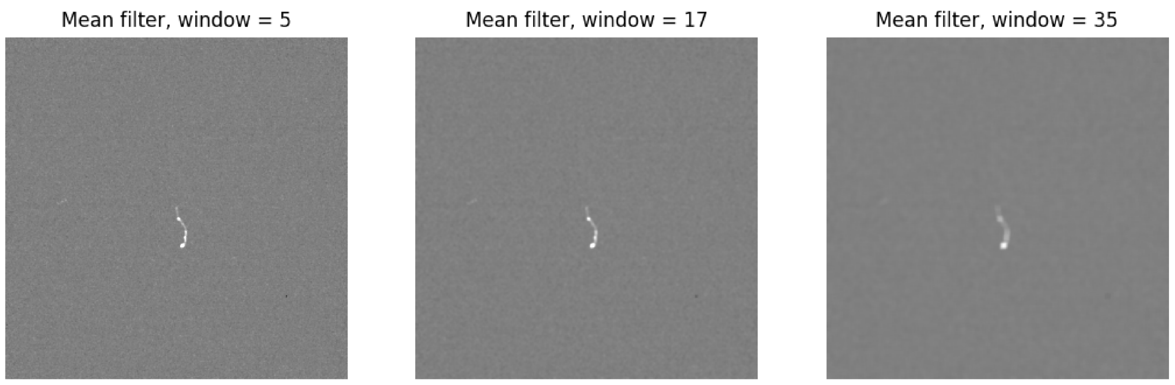


Figure 33: Example of mean filtering applied to simulated image. When the window size is increased, it increases the blurring of the image.

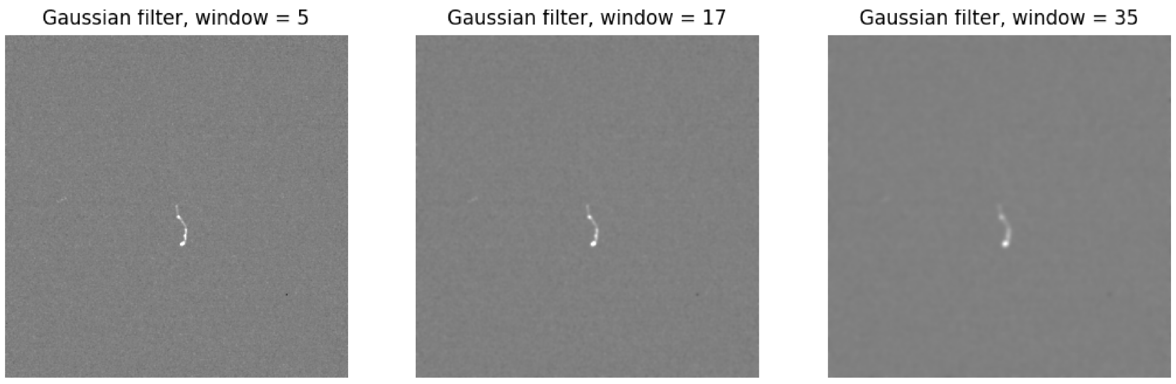

Gaussian filter, window = 5
Gaussian filter, window = 17
Gaussian filter, window = 35

Figure 34: Example of gaussian filtering applied to simulated image. When the window size is increased, it increases the blurring of the image.

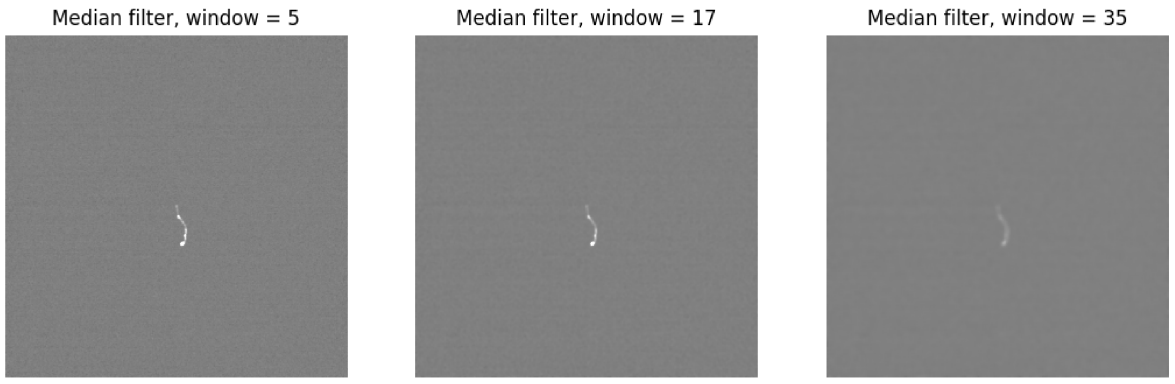

Median filter, window = 5
Median filter, window = 17
Median filter, window = 35

Figure 35: Example of median filtering applied to simulated image. When the window size is increased, it increases the blurring of the image.

For a deep learning based pixel selection, the U-Net explained in section 2.3.2.1 was used to perform the pixel selection task. The idea is that each pixel returns a value between 0 and 1, which reflects the probability of a pixel being considered a signal pixel. The closer to 1, the more likely such a pixel is a signal. On the other hand, the closer to 0, the greater the chance that the pixel is background.

The U-Net implementation has been based in the architecture proposed in section 2.3.2.1 by changing the input image size. Since the experiment images have a 2048x2048 size, the input and output layers of the neural network have changed from 572x572 to 2048x2048 pixels. The proposed network architecture is shown in Appendix A.

Figure 36 shows an example of input that was used for U-Net training along with an example of output presented to the neural network. All pixels that have their truth value greater than 0 are tagged as 1, otherwise as 0.

As the expected output of U-Net should be a probability of pixel with signal, the output layer was defined as Sigmoid function, that is commonly used for binary classification problems.

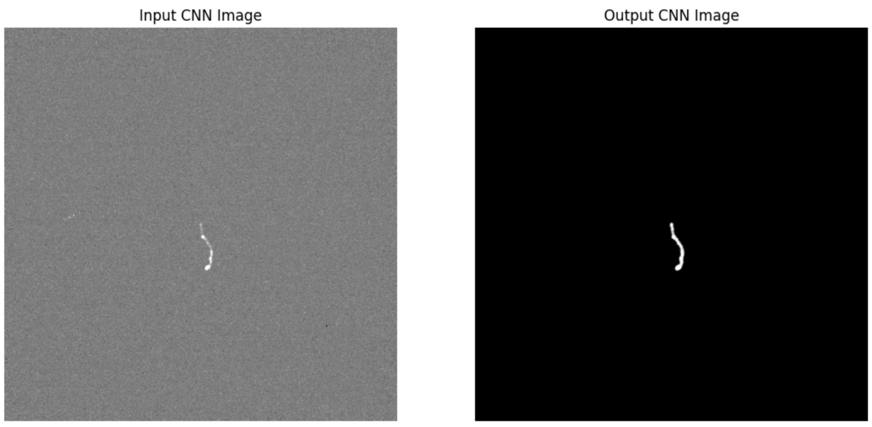


Figure 36: Example of U-Net input (left side) and output (right side).

Once defined the architecture and the images that will be used for U-Net development, the next step is the training. To make the U-Net training feasible, it is necessary to know the ground truth of each image in order to present it to the neural network, minimizing the chosen error function. In this case, the *binary cross entropy* was used, described by the equation 4.2, where y is the truth pixel label and p is the probability returned by U-Net.

$$L = -(y \log(p) + (1-y)\log(1-p)) \tag{4.2}$$

A crucial need for solving problems involving deep learning is the demand for samples in the training process, especially in long tail problems, when there is little representation for some classes or dataset characteristics (CUI et al., 2019). Since the available simulation dataset has only 100 events for each particle type, at each energy level, the strategy adopted was to create new samples and not using the original samples, leaving them for the U-Net evaluation stage. In order to do that, a known process for this type of task was used, the data augmentation defined by (SHORTEN; KHOSHGOFTAAR, 2019) as a data-space solution to the problem of limited data.

The main purpose of data augmentation in the present work is to increase the sample size without losing the main characteristics of the data used for training. In order to maintain these characteristics, such as format, energy of events, and also the electronic noise, only rotation and translation transformations were used to generate new images. The equation 4.3 was applied to the original image pixels, for x_0 and y_0 translation parameters and θ the rotation parameter, in order to generate new images based on the original. It is important to note that Data Augmentation process was applied before noise inserting, keeping noise characteristics for generated images. An example of $\theta = 25°$ rotation is shown in figure 37.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sen(\theta) & x_0 \\ sen(\theta) & cos(\theta) & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4.3}$$

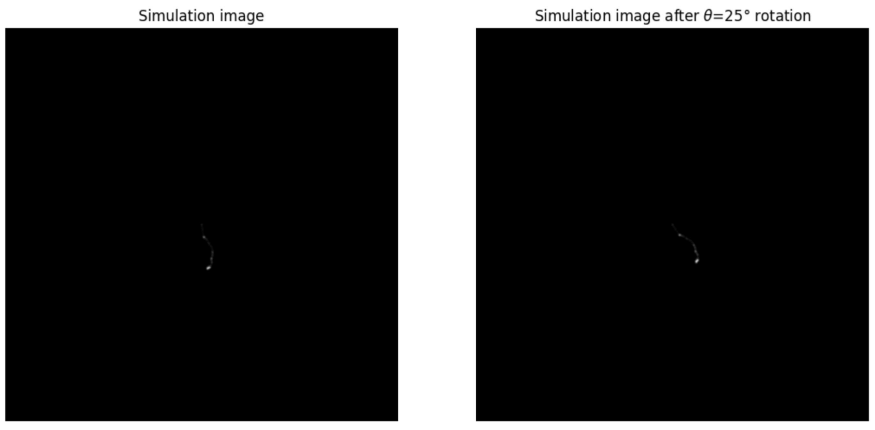


Figure 37: Data augmentation illustration. The right image has the same content of the left, but the activate pixel is rotated by 25°.

Based on the definitions previously made, the next step is to train the network to evaluate the results. By using the data augmentation process, 10000 images were generated for training the neural network, out of which 7000 (70%) have been used for training and 3000 (30%) for validation. The network was trained for 50 epochs, with 450 steps per epoch at a learning rate of 0.02. The optimizer used for this task is Adam Optimizer (DÉFOSSEZ et al., 2020), since it normally presents a fast convergence compared to the others. Figure 38 shows the training process evolution for the U-Net.

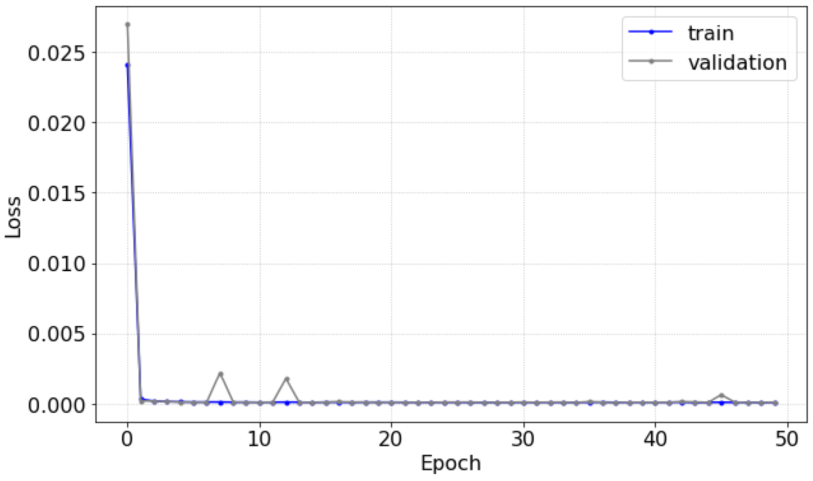


Figure 38: U-Net training step. After few epochs, the loss function reaches a very small value for both datasets (train and validation).

It is possible to notice that the loss function for training and validation data tends to a small value after a few epochs. At some epochs, the value of validation datasets presents some peaks. This can explained because some batch of validation images can be more complex to predict (Low energy, for example), but these peaks appear to decrease along the training epochs.

An example of U-Net output for a test image sample can be seen in figure 39. On the left side it is presented a simulated image used as input of U-Net; on the right side, the output after pixel-wise inference can be seen. A color bar was inserted in order to show that the output is an image of probability, obtained from the sigmoid activation function at the last layer.

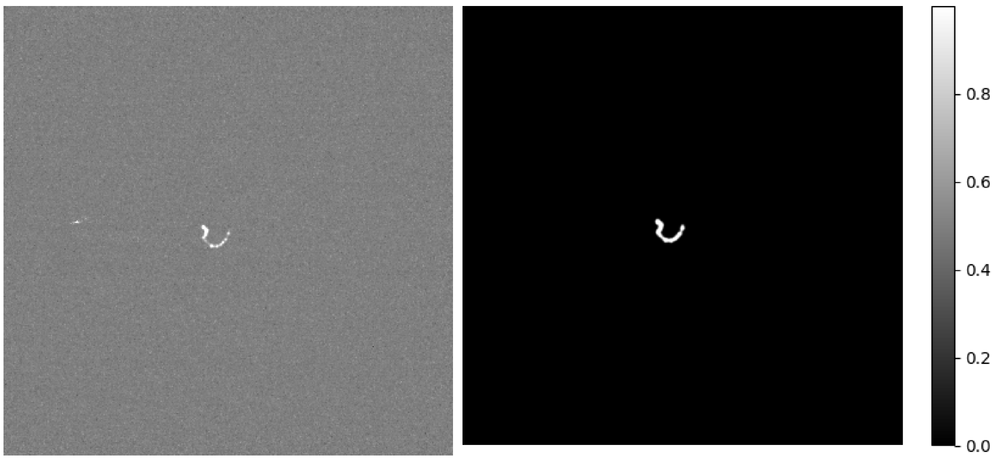


Figure 39: Example of U-Net output after training process. On the left side image, the input of U-Net; on the right side, the output of it after epoch 50. The color bar on the right side of the figure shows that the range of output pixels are between 0 and 1, as it is expected for Sigmoid activation function.

Once the filters and the U-Net were developed, the next step is to define a metric to evaluate the efficiency of those algorithms. The chosen metrics are defined in the next section.

4.3.5 EFFICIENCY

One of the most important steps of the reconstruction algorithm is the pixel selection step. The main goal of this step is to maximize the number of signal pixels that are signal in fact on the output, as well as to minimize the number of background pixels classified as signal by that step, also known by SNR (Signal to Noise Ratio).

There are several ways to measure how good a classifier is, once an algorithm is being used to infer whether a certain element (pixels, in this case) are part of a previously defined group (signal or background). Since each simulation image has only one event, the percentage of filled pixels in an image is extremely small, about 0.1%. Some metrics, as accuracy or ROC curve, may lead us to think that the classifiers are performing well when, in fact, they are not, because it is a problem of unbalanced classes. Because of that, the metrics that were chosen at this step are commonly used for this kind of problem, Precision and Recall (JUBA; LE, 2019), which can be defined by equations 4.4 and 4.5, respectively.

$$Precision = \frac{TP}{TP + FP} \tag{4.4}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.5}$$

Where TP are the signal pixels classified as signal pixels, FP are background pixels classified as signal pixels and FN is the signal pixels classified as background. By changing the threshold value, a Precision-Recall curve can be built in other to evaluate all possible operation points, as shown in Figure 40

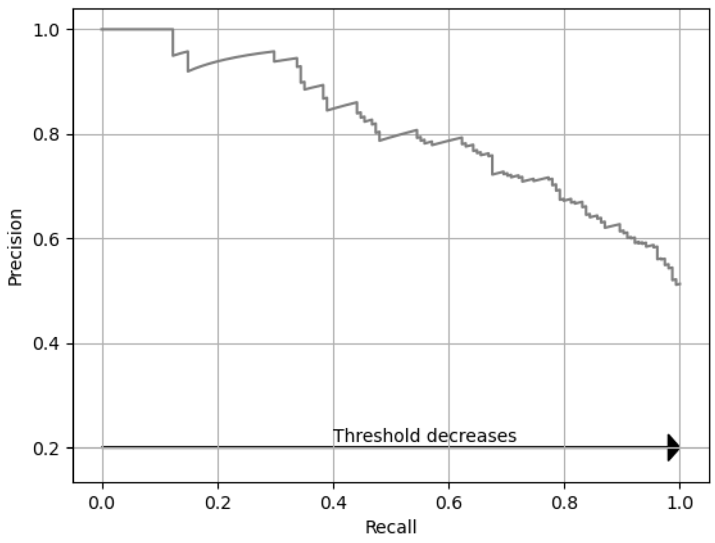


Figure 40: Precision-Recall curve and threshold relationship. When threshold value is high, the value of recall tends to maximum (1); on the other hand, when threshold decreases, precision value tends to minimum (0)

Thinking about images and threshold, when using a high threshold, it means that most of the pixels that exceed this value will feature high intensity and are probably part of a cluster of some particle. In other words, most of the pixels classified as signal, in fact are implying a high value of precision. For recall, as the threshold value increases, most of the lower intensity pixels will be rejected by the threshold, presenting a lower value.

Looking from the inverse perspective, when the threshold value is low, most of the pixels are classified as a signal, and therefore, probably most of the pixels that are actually signals will be identified, generating a high recall value. On the other hand, a low threshold value generates a large volume of background pixels classified as a signal, thus causing a decrease in the precision value.

Each pair (filter, parameters) or any filter selection technique will have their own

Precision-Recall curve, by changing threshold value and calculating Precision and Recall. Therefore, one way to choose the best parameter for a filter would be the best precision-recall curve.

To perform this task in a systemic way, a summary measure of the precision-recall curve can be used. For this work, the F1-score was used, and it can be defined by the equation 4.6.

The F1-score is the harmonic mean of precision and recall (FLACH; KULL, 2015), and it is a metric that has a lower and an upper bound that, in this case, are 0 and 1, respectively.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4.6}$$

Looking into equation 4.6, it is possible to notice that the maximum value of F1 score is when precision and recall are equals to 1. When precision or recall is zero, the F1-score will also be zero.

Each pair precision-recall corresponds to one F1-score value. To summarize each precision-recall curve, the maximum value will be chosen and it will represent how efficient a filter is comparing with the ground truth.

5 RESULTS

As mentioned before, the main objective of this work is to verify if digital filters have the potential to produce relevant improvement of the signal-to-noise ratio of events acquired with the LEMOn detector. In order to accomplish that, a set of measures should be defined and evaluated. The proposed evaluation is divided into three parts: (1) pixel selection performance; (2) assessment of the impact of filters on the reconstruction algorithm; and (3) real data analysis. In addition to being used for performance comparison between filters, the results of step (1) will also be used to select the best filters parameters based on simulated images, as shown in figure 28. Once filters parameters have been chosen, step (2) will evaluate their impact in the CYGNO reconstruction algorithm. The energy values provided by the simulation will be used to measure the filters' impact regarding energy estimation. Finally, step (3) proposes a qualitative analysis based on energy histograms constructed from the output of the reconstruction algorithm and on the processing time spent by each of the proposed filters. In all the steps, the version used by the CYGNO Experiment before the insertion of the changes proposed in this work will also be compared. All these steps will be presented in more detail in the next sections along with the achieved results.

5.1 PIXEL SELECTION PERFORMANCE

Based on the considered methodology, F1 score is used to evaluate the proposed filters, using all images from the pack. The main goal of this step is to choose the best parameter for each filter. For the window-based filters (smoothing filters), its dimension was scanned from 9 to 21. Figures 41, 42 and 43 show the best F1-score values found for each kind of particle.

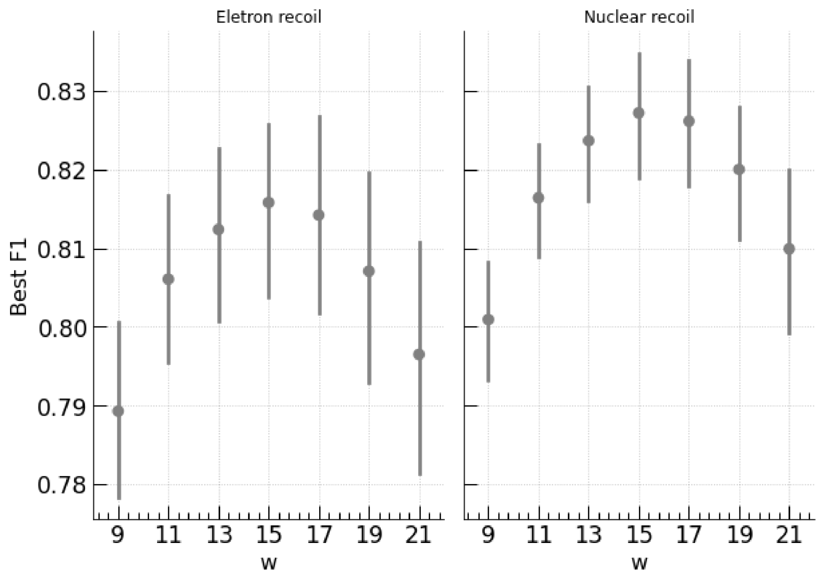


Figure 41: F1-Score evaluated for all images using a median filter. The error bar shows the standard deviation of this measure for each window value.

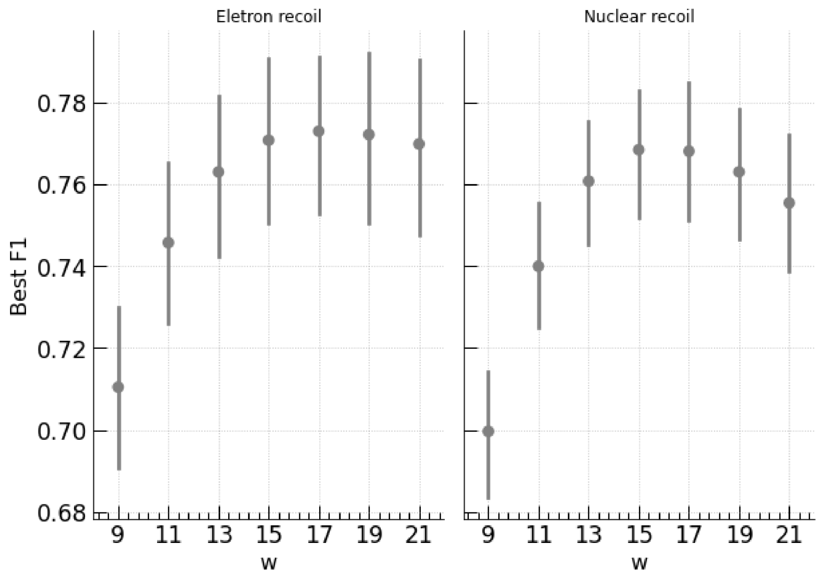


Figure 42: F1-Score evaluated for all images using a gaussian filter. The error bar shows the standard deviation of this measure for each window value.

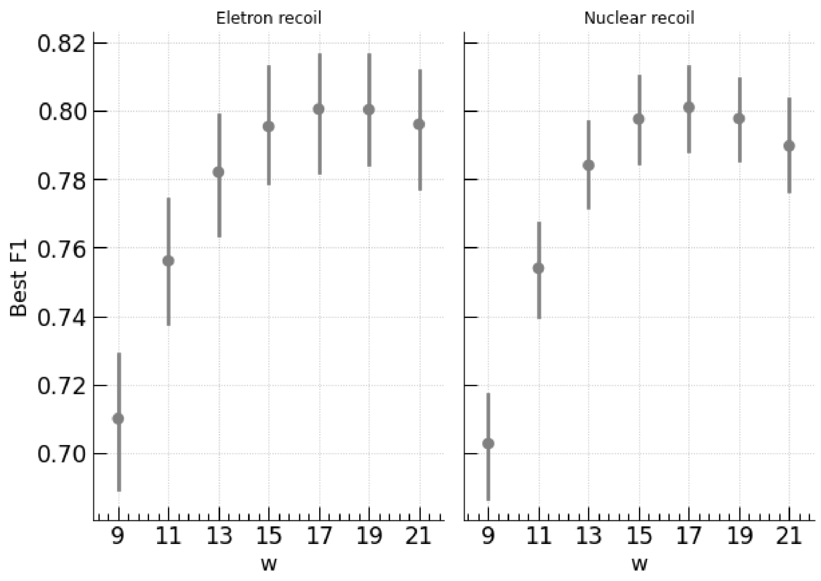


Figure 43: F1-Score evaluated for all images using a mean filter. The error bar shows the standard deviation of this measure for each window value.

For U-Net, the parameters have been defined from the training process described in 4.3.4. In figure 44 it is possible to see F1-Score values, for each epoch, for both training and validation datasets.

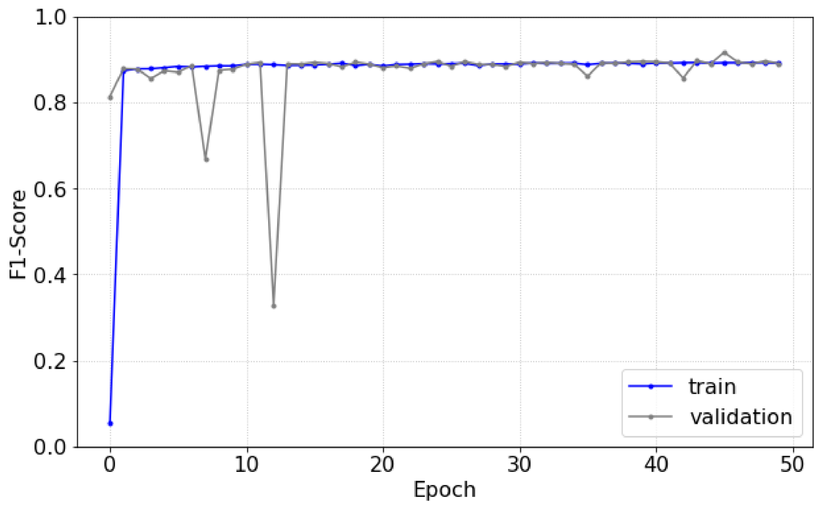


Figure 44: U-Net F1 Score along epochs evolution using train and validation datasets. A plateau can be noticed after epoch 15 for both datasets.

For filter selection method of reconstruction algorithm, as it uses a simple cut based on the standard deviation of each pixel noise, only one precision-recall curve is built, based on threshold value changes. Table 1 shows the value of F1-Score mean and standard deviation for all simulation data, when this pixel selection way is used.

It is possible to notice that the distribution of the best F1 scores present the best

Pixel selection method	Event	Best F1 ($\mu \pm \sigma$)
$n.\sigma_{noise}$	Electron recoil	0.172 ± 0.157
$n.\sigma_{noise}$	Nuclear recoil	0.102 ± 0.073

Table 1: Evaluation standard deviation pixel-based selection

results for when the algorithm uses filtering processes, remembering that the maximum (and best) value of F1-score that it is possible to obtain is 1.

Another important point regards the variance of the measured F1-score, which happens due to the intensity variation of the tracks in the images. Tracks with lower energy tend to have lower F1-score when compared to higher energy tracks. This can also occur in the case of electron recoils, where there is great variation in intensity between the tracks.

The best parameters for Smoothing filters can be chosen by looking at figures 41, 42 and 43. For the gaussian and mean filters, 17 is the best value for w, while for the median filters, w=15 should be chosen. The U-Net parameters have been defined by training process. A summary of these values and the respective descriptive statistics for best F1-score is given in Table 2, when test dataset was used to evaluate the chosen parameters.

Algorithm	Parameter	Best F1 ($\mu \pm \sigma$)
U-Net	Many	0.873 ± 0.060
Median	$w = 15$	0.820 ± 0.127
Mean	$w = 17$	0.800 ± 0.184
Gaussian	$w = 17$	0.771 ± 0.225
Cygno	None	0.137 ± 0.127

Table 2: Filters evaluation simulation data resume

5.2 *EVALUATING FILTERS IMPACT ON RECONSTRUCTION ALGORITHM USING SIMULATION DATA*

Once the parameters are chosen according to the established metric, the next step is to insert the filtering processes into the CYGNO algorithm structure. Such measure must be taken to check the impact of the filtering processes considering the entire reconstruction chain. It is important to mention that the CYGNO collaboration has developed a post-processing code which aims at reducing the number of points sent to the clustering algorithm. This code has a high computational cost and also uses filtering processes which we intend to replace by the filters proposed in this work. Therefore, besides the inclusion of such filters, our proposal also incorporates a change in the structure of the CYGNO algorithm.

The most important variables to measure at this step are the reconstructed energy at the output of algorithm and its processing time. The main objective is to verify if the use of the proposed filters provides an improvement in the energy estimation of the events and/or reduces the processing time. This becomes possible from the simulation images, once the energy of the events is known. Each simulated image has only one cluster. Linear regression is used to relate the input E_i energy, given by the simulation software, and the output or estimated energy E_o by means of equation 5.1.

$$E_i = \alpha E_o + \beta \tag{5.1}$$

At the best scenario, each pair (E_o, E_i) will form a scatter plot, and a linear model could be used to fit those points. In the ideal case, the intercept β should be equal to 0, and the slope α, equal to 1. In this situation, the value of the reconstructed energy is the same, in average, of the input energy. However, α and β alone are not enough, since it is possible to have this ideal conditions met but with a linear model that does not fit well the data. Hence, the quality of the linear fit must also be analyzed.

In order to do that, two other measures will computed: R^2 and Shapiro-Wilk normality test. R^2 measures the percentage of the variance of the data explained by the fit model. Its equation is shown in equation 5.2, where σ_{res} is the total variance around the linear model and σ_{tot} is the total variance of the data. When σ_{res} is small in relation to σ_{tot} (good fit), R^2 tend to be close to 1. On the contrary, when σ_{res} is close to σ_{tot} (poor fit), R^2 gets close to 0.

$$R^2 = 1 - \frac{\sigma_{res}}{\sigma_{tot}} \tag{5.2}$$

R^2 is an important indicator of fit goodness but it is not sufficient. This can be better explained by Anscombe's quartet (REVELL et al., 2018), shown in figure 45. Although all datasets are completely different, they have the same mean, standard deviation and R^2. For regression purposes, just the first fit (I) is reasonable, because there is a symmetrical behavior of the points along the fit curve. Taking that into account for this work, the fit outcome will be considered only when the distribution of data points around the fit model follows a Normal distribution. For validation of this condition, a normality test will be applied - the Shapiro-Wilk normality test (RAZALI; WAH et al., 2011). This test is indicated for datasets with reduced number of samples (< 5000). For this test, the null hypothesis H_0 is that the tested distribution is not Gaussian, so when the fit *p-value* is higher than a certain threshold, the null hypothesis must be rejected and the tested distribution is considered to be Gaussian. The threshold value defined to reject or keep H_0 is 0.05 (5%).

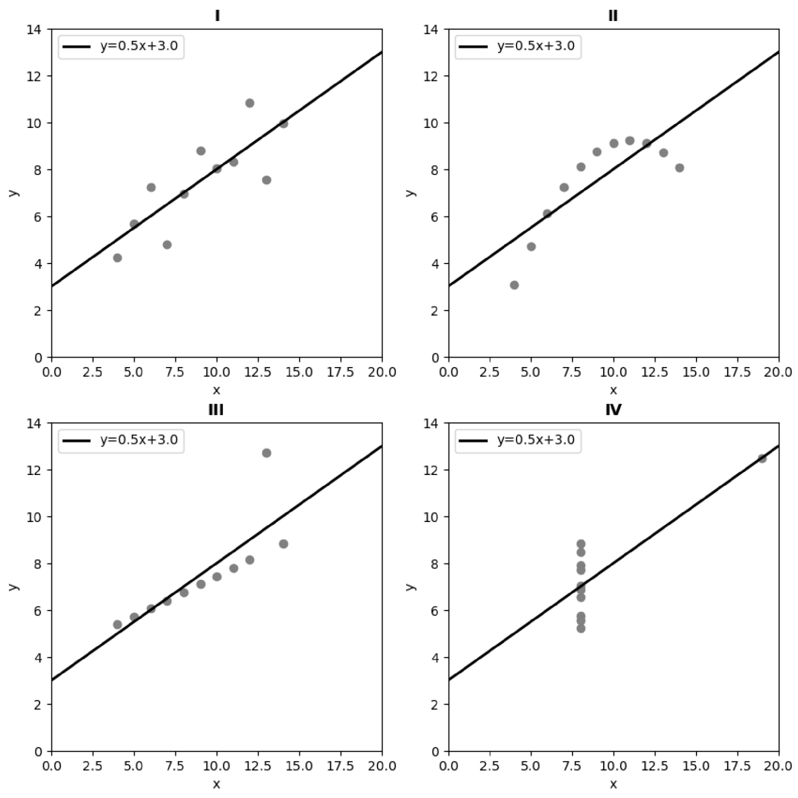


Figure 45: Anscombe's quartet

Figure 46 shows an example of a linear model not fitted to the data. In addition to the low value of R^2, the error distribution between the points that make up the dataset and the fit curve is not Gaussian, failing the Shapiro-Wilk test, with a *p-value* lower than 0.05. On the other hand, in figure 47, the linear model, according to the *p-value* showed in the right plot, fits well the data, and the null hypothesis is rejected.

The metrics described above are used as a basis for evaluating the impact of filters in the CYGNO Experiment. The diagram of figure 48 summarizes the methodology used to measure the performance of filters in simulated images.

The flowchart is similar to the one used for evaluation and selection of filter pa-

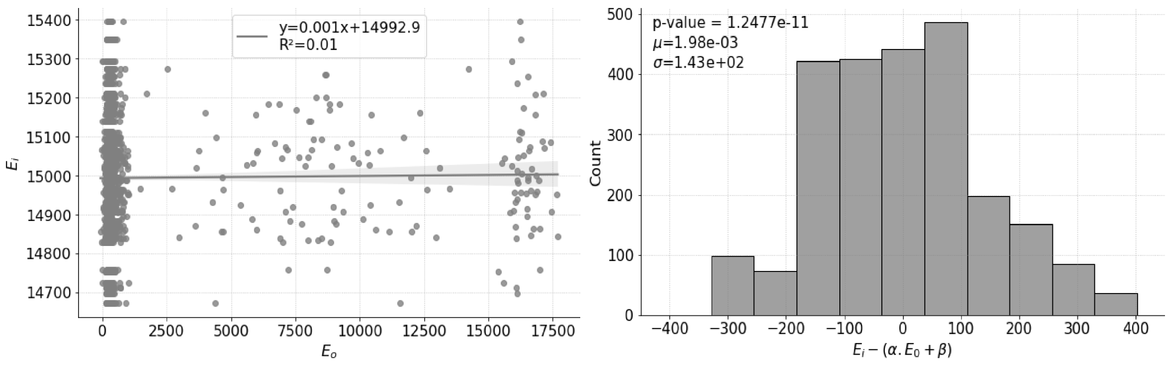


Figure 46: Fit example

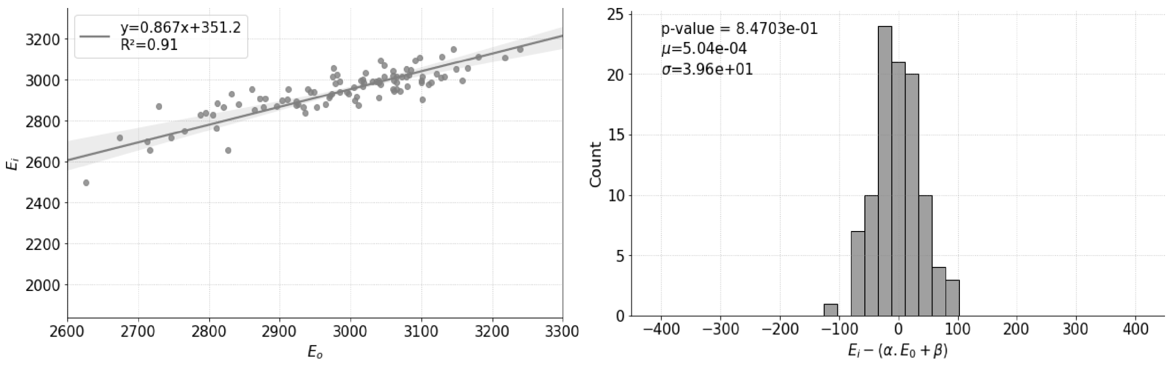


Figure 47: Fit example

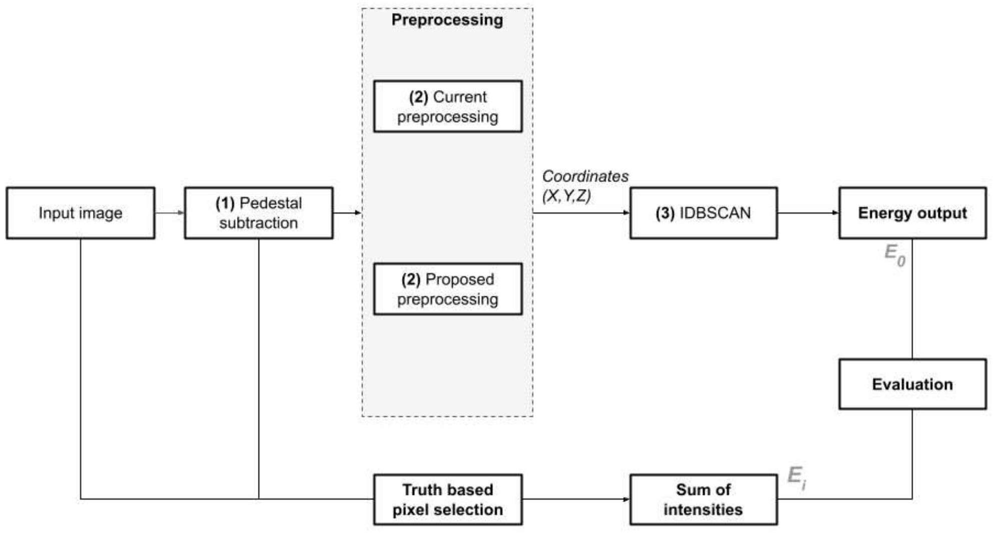


Figure 48: Reconstruction algorithm evaluation flowchart diagram.

rameters, but now the evaluated variable is energy. The reference energy (E_i) used to evaluate the proposed filters will be calculated from an ideal clustering algorithm, where all signal pixels are identified and all noise pixels rejected. The final energy is obtained from the sum of the intensities of all pixels belonging to the reconstructed cluster.

Before the event energy estimation process, noise is added to the simulated images, forming the input image shown in flowchart of Figure 48. These images are sent to be processed by the different proposed versions of the reconstruction algorithm. The output of this algorithm defines the clusters, from which the energy E_o can be estimated by the sum of the intensities present in each of its pixels. In the end, the closer the response of a proposed algorithm is to that of an ideal algorithm, the better is its performance.

It is important to notice that, for the proposed reconstruction algorithm, the filters should be applied before the threshold cut (see Figure 28), used to select the pixels to be sent to the clustering algorithm. Therefore, threshold is not a global parameter anymore, and its value should be chosen according to the applied filter in such a way that threshold cannot be used as a common reference in the filters' performance analysis, for comparison purposes. The common reference used will be the number of elements sent to the clustering algorithm, which depends on the threshold but which can be considered significant for the proposed analysis, since one of the main objectives of the filters is to improve the signal-to-noise ratio of the images, making them able to perform better while sending a smaller number of pixels to the clustering phase. Therefore, if two algorithms, for example, have the same performance in relation to energy estimation, but one of them achieves this result by sending fewer pixels to the clustering process, this would be considered the filter with the best overall performance. In this section, the threshold (or the number of elements sent to clustering) was chosen in order to obtain the best performance from the filters for each energy. That is, for each filter and each event energy, a different threshold was used. It will be possible, however, to correlate the achieved performance regarding energy estimation with the number of elements passed to the clustering algorithm, providing the means to detect the most efficient filters in terms of these two parameters. Section 5.3 will complement the results shown here by comparing the performance of filters for the same number of elements and computing their processing time.

Figure 49 shows the R^2 values for each filter, energy and interaction type. It is possible to note that R^2 is lower for low energy values, below 6 keV. For such energy range, U-Net or the median filter achieved the best results for both, ER and NR events. When event energy increases, R^2 reaches values close to one for NR events and all the proposed filters, except for the collaboration algorithm. For ER events, the U-Net and Median filters were also those that achieved the best results, while the Gaussian and Mean filters obtained the worst results.

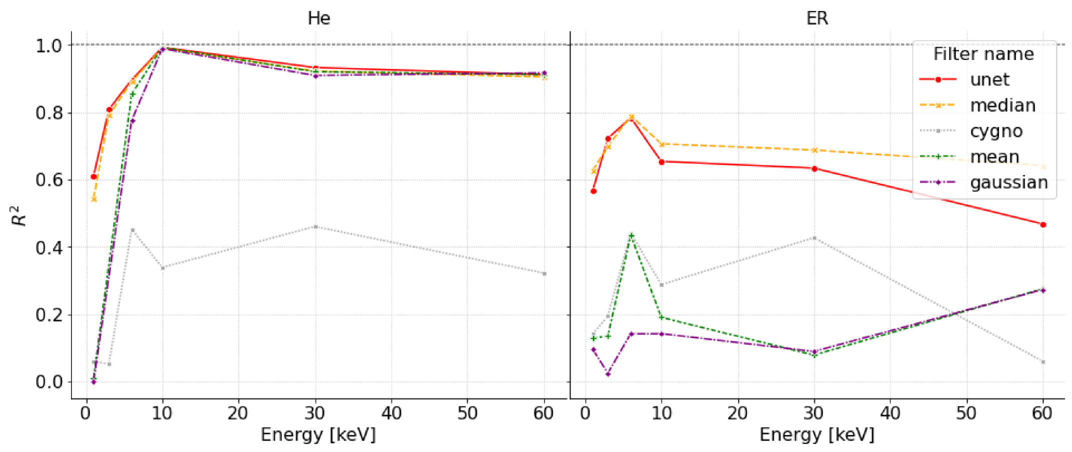


Figure 49: Evaluation of R^2 for NR particle (He represents Helium gas that the simulation detector was filled with) and ER.

Figures 50 and 51 show a measure of the linear model parameters α and β_{st}, respectively. Since the bias β is not immune to scale, a normalized bias β_{st} is used, being computed from the ratio between bias β and the expected energy. From the graphics it is possible to see that U-Net and median filters show the best results for NR events, getting close to α=1 and β_{st}=0, while the collaboration algorithm is the worst. Note that those results are highly consistent with the R^2 results, as it is for the ER events.

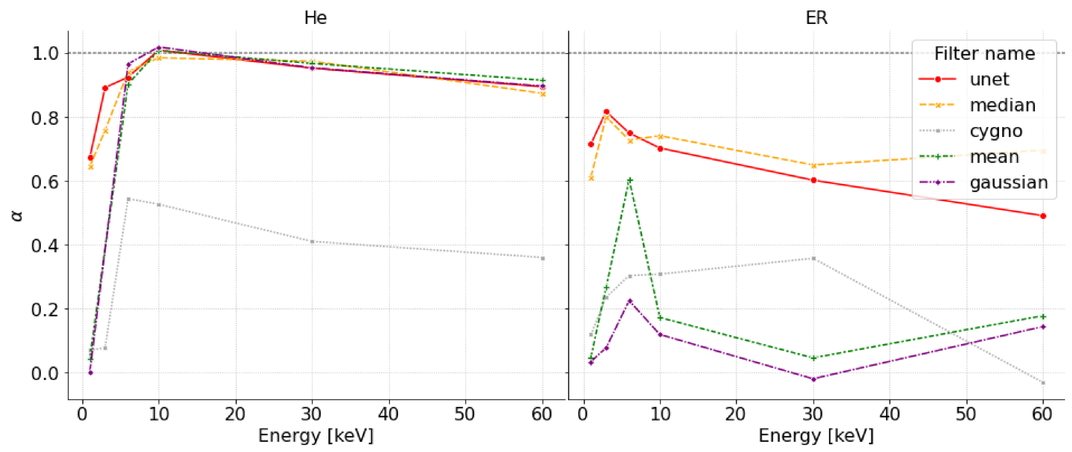


Figure 50: Evaluation of slope (α) for NR particle (He) and ER.

Another important measure is the number of detected clusters. Figure 52 shows the efficiency of the proposed algorithms. When efficiency is higher than 100%, it means that the algorithm found more clusters than the number of events. This can happen due to fake clusters, formed from noise pixels, and due to separation of the event tracks into more than one cluster. It is important to notice that for energies above 10 keV, the ER events loose their characteristic to form point-like tracks, allowing them to be divided into more than one cluster more easily. As it can be seen, for energies below 6

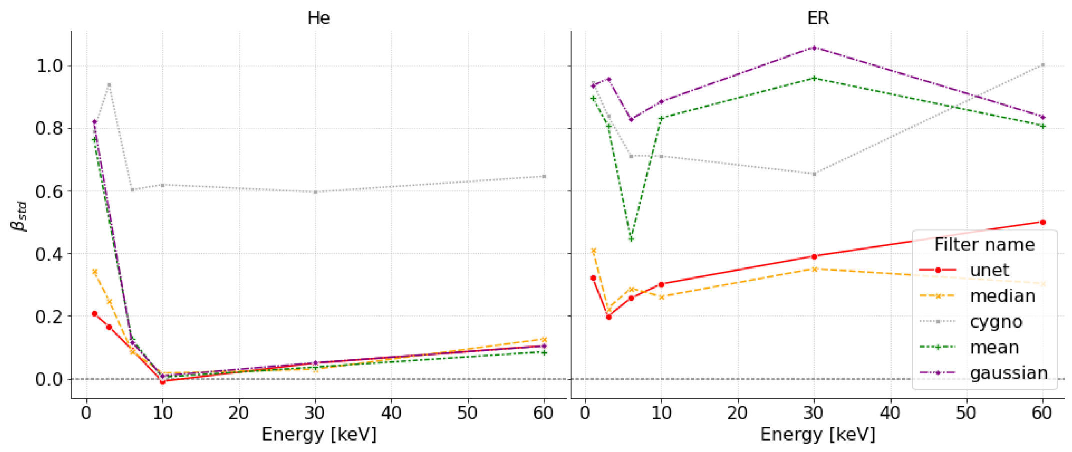


Figure 51: Evaluation of intercept (β) for NR particle (He) and ER. The intercept is standardized to make possible the comparison between different energies.

keV, not all filters can identify all clusters provided by the images. However, with the increase in energy, this task becomes simpler and their efficiency increases considerably. However, for ER events and energy above 10 keV, all the filters tend to work with an efficiency higher than 100% due to the reasons mentioned above. Again, it is important to note the consistency of these results with the previous ones.

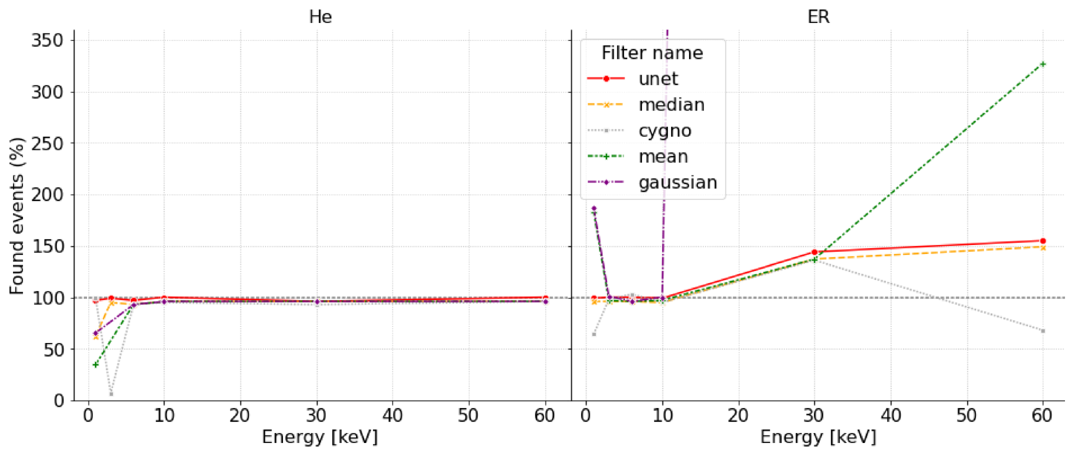


Figure 52: Evaluation of cluster detection for NR particle (He) and ER

Figure 53 shows the standard deviation of distributions composed of values computed from the difference between output and input energies, for NR (right) and NR (left), respectively. From these plots it is possible to observe that the U-Net and Median methods stood out throughout the whole energy range under evaluation for ER events. For NR, these same filters performed better than the others, with emphasis on the low energy region below 6 keV.

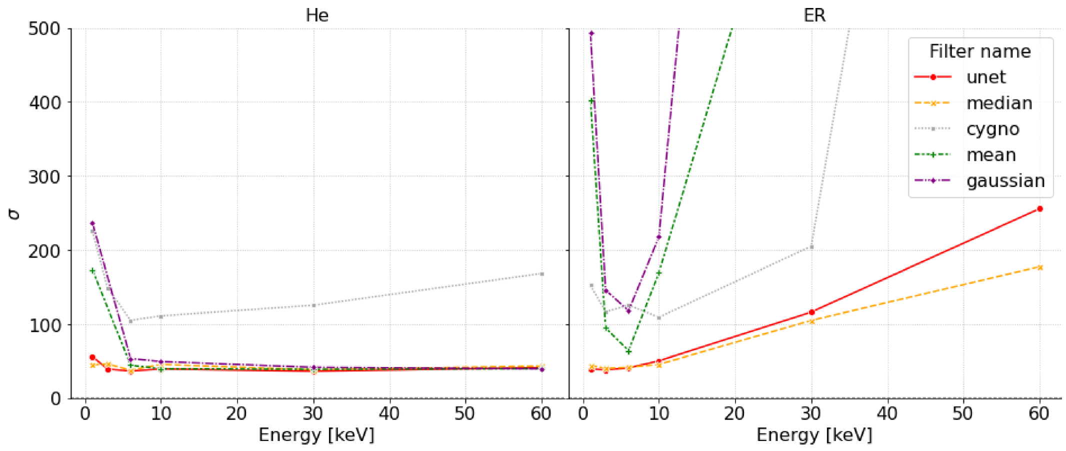


Figure 53: Standard deviation of the difference between output and input energies for NR (He) and ER events

Finally, table 3 shows the average of number of pixels sent to the clustering process, after threshold cut, for each tested algorithm. As it can be seen, the collaboration algorithm needs to send around 50000 pixels to achieve its best performance, while all other algorithms need about two orders of magnitude fewer events.

Filter name	Number of pixels after threshold
U-Net	450
Mean	500
Gaussian	500
Median	700
Cygno	50000

Table 3: Average number of points after threshold

From the results presented in this section, we should highlight the performance of the U-Net and Median filters, which showed that it is possible to improve the performance of the algorithm used by the collaboration.

5.3 *EVALUATING FILTERS ON RECONSTRUCTION ALGORITHM USING REAL DATA*

This section intends to complement and validate the previous results by analyzing the impact of filters when applied to real data. For the sake of simplicity, only the two filters that achieved the best results with simulated data (median and U-Net) and the collaboration algorithm will be considered. An important remark is that the same methodology cannot be applied for real data since there is no presumptive knowledge about the expected output. Therefore, in this section the analysis is divided into two parts: (1) evaluation of the reconstructed energy distribution, aiming to compare the filters output with the output coming from the collaboration algorithm and, (2) measurement of the processing time.

5.3.1 RECONSTRUCTED ENERGY

The energy estimated by the filters based algorithms will be compared to the collaboration algorithm. Since the selected dataset has around 900 images, the reconstruction process is time consuming and, consequently, only four threshold values are considered, based on the number of elements sent to the clustering process: 10k, 30k, 100k and 300k. It is important to note that the collaboration algorithm uses a threshold of 1.3σ, which sends 300k elements for clustering.

Figures 54 and 55 show a comparison between the filters' output energy distributions (median and U-Net filters respectively) and the collaboration energy distribution for each operation point.

As can be seen in figure 54, when threshold is lower, the low energy region is populated with events which eventually come from a process of segmenting a track into more than one cluster. When compared to the collaboration results, for a threshold of 1.3σ (300.000 points), the median filter approaches its distribution for a operation point of 300k, both in shape and in its descriptive measures.

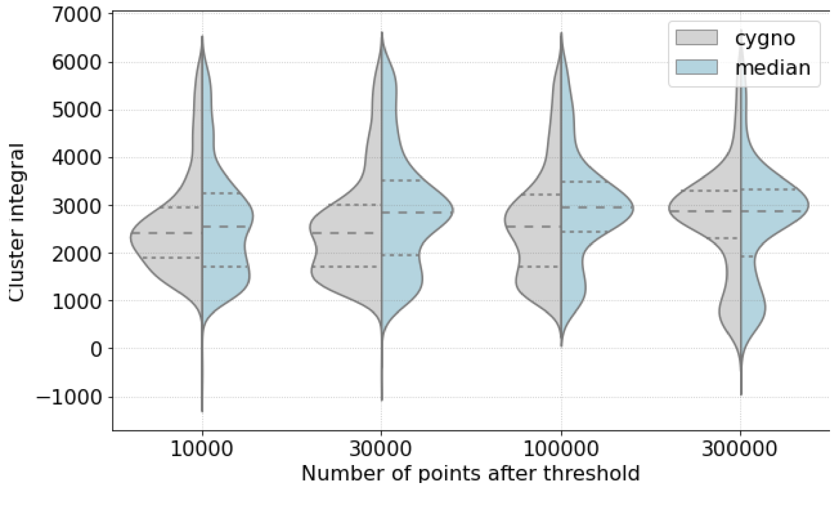


Figure 54: Median filter comparison

For U-Net, a similar behavior can be seen, but at 300k points, the density within the low energy region becomes very high. The reason for that to happen is the formation of low energy clusters due to the large volume of background events sent to the clustering algorithm. The best scenario for the U-Net algorithm seems to occur somewhere between 30k and 100k points.

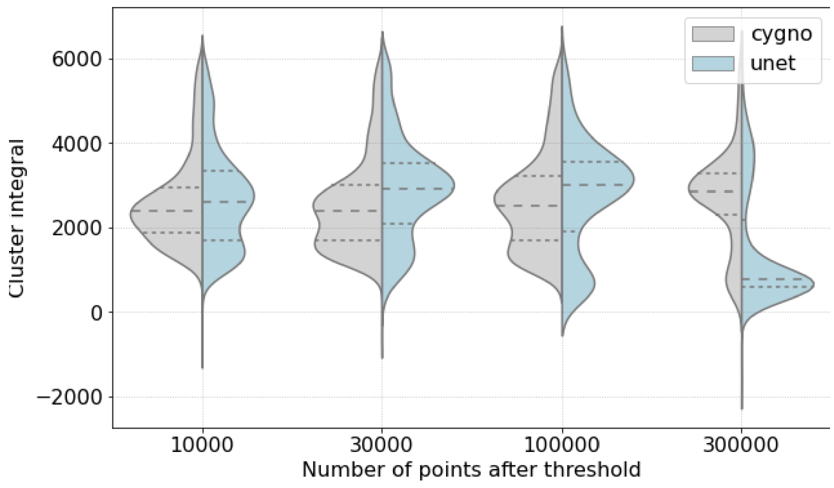


Figure 55: U-Net filter comparison

Considering both comparisons above, it is worth mentioning that the collaboration algorithm was massively tested during its usage time, causing its operating point (threshold) to be found. For the proposed algorithms, due to the high processing time required, only four operating points were tested. For U-Net, as mentioned, there is a possibility that the points tested are still far from the best operating point it can work with. For the median filter, the result obtained with 300k points is very similar to that obtained with the collaboration algorithm. In Figure 56 it is possible to see the energy distributions for the median and collaboration algorithms, for the ^{55}Fe acquisition. These distributions appear to be very similar, which can also be verified from their descriptive statistics shown in Table 4. An important detail is that for the median filter, the value of the first quantile occurs at a lower energy value. This implies that such a configuration has a higher percentage of elements at higher energies. This may imply cluster union, which is partitioned into lower energy clusters when using the collaboration algorithm. For the other quantiles, the statistics present approximate values.

Filter	Q_1	Q_2	Q_3
Cygno	2298.40	2857.92	3291.80
Median	1932.65	2872.45	3328.62

Table 4: Quantile comparison

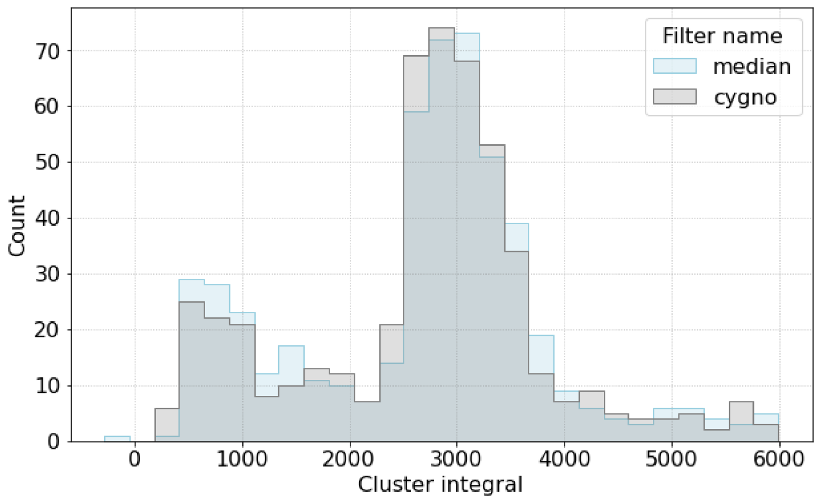


Figure 56: Cluster integral comparison between median filter and current collaboration algorithm. Cluster integral represents the sum of intensity of all pixels that are in each cluster. This value is proportional to energy of events.

In order to statistically confirm whether the two distributions are similar, the Kolmogorov-Smirnov test (BERGER; ZHOU, 2014) was applied. For such test, the null hypothesis is that the distributions are not similar. The KS value found was 0.050532, with a *p-value* = 0.514255. As the *p-value* is greater than 0.05, the null hypothesis can be rejected.

5.3.2 PROCESSING TIME

From the previous analysis, it was possible to realize that the U-Net and Median based algorithms can select more efficiently the signal pixels of the events (section 5.1), providing better quality in their energy estimation (section 5.2) and, at least, reproduce the same energy distributions measured with real data when compared to the algorithm used by the collaboration (section 5.3.1). In this section, the processing time of such algorithms shall be measured. For this purpose, for each image, the processing time, from the beginning of the pre-processing part to the step where the event energy is estimated, was measured.

Measuring processing time is a complicated task in terms of computation, since other processes may be running, interfering with the measurement of this quantity. Trying to mitigate this effect, all algorithms were executed using cloud computing, ensuring that only these processes were running at the time of measurement.

Figure 57 shows the processing time measurements in box-plot format for all the evaluated operating points. As can be seen, U-Net obtained the best result among the tested algorithms. One of the reasons for this is due to the use of GPU for image inference, reducing the time to perform the filtering. The median filter took about twice as long as the U-Net, while the collaboration algorithm took almost an order of magnitude longer. There are two reasons that can justify this effect. One is that the CYGNO algorithm uses a computationally heavy process after threshold cut to eliminate isolated noise pixels, called *noise-reductor*. The other reason is that the use of filters allows to have a better signal detection performance, selecting more signal pixels while rejecting, at the same time, more noise pixels. It consequently reduces the number of pixels after the rebin process, as shown in figure 58.

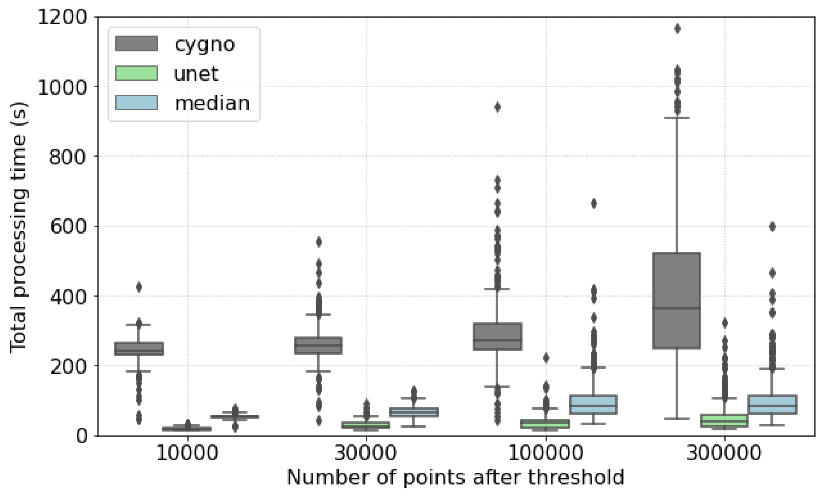


Figure 57: Processing time (All steps)

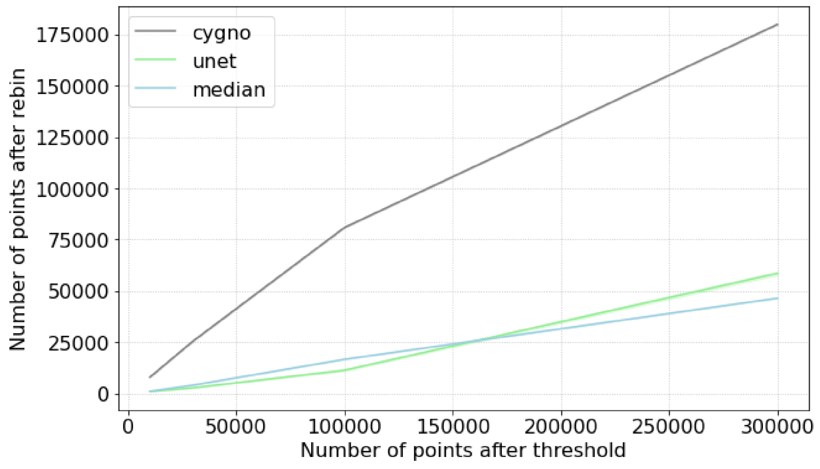


Figure 58: Number of points after rebin

6 CONCLUSIONS

This work proposed to study the impact of classic digital filters and a CNN in the detection process of pixels with presence of signal from the release of energy produced by particles that interact inside a TPC detector within the framework of the CYGNO experiment. Simulated data were used to optimize the proposed filters and the CNN, as well as to measure their performance regarding energy estimation. At the same time, real data were analyzed as a way to validate the results and to measure the processing time of the concurrent algorithms.

The analysis performed on the simulation data demonstrated the potential of filters to improve the energy estimation of the experiment while, with the real data, it was possible to conclude that the filters can also provide a great reduction of the processing time needed to run the reconstruction algorithm of the experiment.

In future work, the segmented regions provided by the CNN can also be used for energy estimation, eliminating the need for a clustering algorithm, which is responsible for overloading a large part of the image processing time. This proposal has already being implemented, and the results based on the simulated data are considered auspicious. Besides the need to deepen this analysis, CNN could also be used for classifying the type of particle, performing the already known instance segmentation. These subjects will be addressed in the near future.

REFERENCES

AHMED, Z. et al. Combined limits on wimps from the cdms and edelweiss experiments. *Physical Review D*, APS, v. 84, n. 1, p. 011102, 2011.

ANTOCHI, V. et al. A gem-based optically readout time projection chamber for charged particle tracking. *arXiv preprint arXiv:2005.12272*, 2020.

ANTOCHI, V. C. et al. Combined readout of a triple-GEM detector. *JINST*, v. 13, n. 05, p. P05001, 2018.

BARACCHINI, E. et al. A density-based clustering algorithm for the cygno data analysis. *Journal of Instrumentation*, IOP Publishing, v. 15, n. 12, p. T12003, 2020.

BARACCHINI, E. et al. Negative Ion Time Projection Chamber operation with SF_6 at nearly atmospheric pressure. *JINST*, v. 13, n. 04, p. P04022, 2018.

BAUDIS, L. Direct dark matter detection: the next decade. *Physics of the Dark Universe*, Elsevier, v. 1, n. 1-2, p. 94–108, 2012.

BEHNKE, E. et al. Final results of the picasso dark matter search experiment. *Astroparticle Physics*, Elsevier, v. 90, p. 85–92, 2017.

BERGER, V. W.; ZHOU, Y. Kolmogorov–smirnov test: Overview. *Wiley statsref: Statistics reference online*, Wiley Online Library, 2014.

BERTONE, G.; HOOPER, D.; SILK, J. Particle dark matter: Evidence, candidates and constraints. *Physics reports*, Elsevier, v. 405, n. 5-6, p. 279–390, 2005.

BOLYA, D. et al. Yolact: Real-time instance segmentation. In: *Proceedings of the IEEE/CVF international conference on computer vision.* [S.l.: s.n.], 2019. p. 9157–9166.

CHAKI, J.; PAREKH, R.; BHATTACHARYA, S. Plant leaf recognition using ridge filter and curvelet transform with neuro-fuzzy classifier. In: SPRINGER. *Proceedings of 3rd International Conference on Advanced Computing, Networking and Informatics.* [S.l.], 2016. p. 37–44.

COLLABORATION, D.; NETO, J. de M. et al. The damic dark matter experiment. *PoS (ICRC2015)*, v. 1221, 2016.

CUI, Y. et al. Class-balanced loss based on effective number of samples. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* [S.l.: s.n.], 2019. p. 9268–9277.

DÉFOSSEZ, A. et al. On the convergence of adam and adagrad. 2020.

FLACH, P.; KULL, M. Precision-recall-gain curves: Pr analysis done right. *Advances in neural information processing systems*, v. 28, 2015.

FREESE, K.; LISANTI, M.; SAVAGE, C. Annual modulation of dark matter: a review. *arXiv preprint arXiv:1209.3339*, 2012.

GIUSTI, A. et al. Fast image scanning with deep max-pooling convolutional neural networks. In: IEEE. *2013 IEEE International Conference on Image Processing.* [S.l.], 2013. p. 4034–4038.

GONZALEZ, R. C.; WOODS, R. E. et al. Digital image processing [m]. *Publishing house of electronics industry*, v. 141, n. 7, 2002.

GOODMAN, M. W.; WITTEN, E. Detectability of certain dark-matter candidates. *Physical Review D*, APS, v. 31, n. 12, p. 3059, 1985.

GUO, G.; ZHANG, N. A survey on deep learning based face recognition. *Computer vision and image understanding*, Elsevier, v. 189, p. 102805, 2019.

HANIN, B. Which neural net architectures give rise to exploding and vanishing gradients? *Advances in neural information processing systems*, v. 31, 2018.

HEMANTH, D. J.; ANITHA, J. Image pre-processing and feature extraction techniques for magnetic resonance brain image analysis. In: SPRINGER. *International Conference on Future Generation Communication and Networking.* [S.l.], 2012. p. 349–356.

IGUAZ, F. et al. Micromegas detector developments for dark matter directional detection with mimac. *Journal of Instrumentation*, IOP Publishing, v. 6, n. 07, p. P07002, 2011.

INCERTI, S. et al. The geant4-dna project. *International Journal of Modeling, Simulation, and Scientific Computing*, World Scientific, v. 1, n. 02, p. 157–178, 2010.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: PMLR. *International conference on machine learning.* [S.l.], 2015. p. 448–456.

IRASTORZA, I. G.; GARCÍA, J. A. Direct detection of dark matter axions with directional sensitivity. *Journal of Cosmology and Astroparticle Physics*, IOP Publishing, v. 2012, n. 10, p. 022, 2012.

JUBA, B.; LE, H. S. Precision-recall versus accuracy and the role of large data sets. In: *Proceedings of the AAAI conference on artificial intelligence.* [S.l.: s.n.], 2019. v. 33, n. 01, p. 4039–4048.

KARN, U. An intuitive explanation of convolutional neural networks. *The data science blog*, 2016.

KARPATHY, A.; LI, F.; JOHNSON, J. Cs231n convolutional neural networks for visual recognition. 2016. *URL http://cs231n. github. io*, v. 50, 2017.

KIBBLE, T. W. The standard model of particle physics. *European Review*, Cambridge University Press, v. 23, n. 1, p. 36–44, 2015.

KNIRCK, S. et al. Directional axion detection. *Journal of Cosmology and Astroparticle Physics*, IOP Publishing, v. 2018, n. 11, p. 051, 2018.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.

LIU, J.; CHEN, X.; JI, X. Current status of direct dark matter detection experiments. *Nature Physics*, Nature Publishing Group, v. 13, n. 3, p. 212–216, 2017.

LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 3431–3440.

LOPES, G. et al. Study of the impact of pre-processing applied to images acquired by the cygno experiment. In: SPRINGER. *Iberian Conference on Pattern Recognition and Image Analysis*. [S.l.], 2019. p. 520–530.

MARAFINI, M. et al. ORANGE: A high sensitivity particle tracker based on optically read out GEM. *Nucl. Instrum. Meth.*, A845, p. 285–288, 2017.

MARAFINI, M. et al. Study of the performance of an optically readout triple-gem. *IEEE Transactions on Nuclear Science*, IEEE, v. 65, n. 1, p. 604–608, 2018.

Mazzitelli, G. et al. A high resolution tpc based on gem optical readout. In: *2017 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*. [S.l.: s.n.], 2017. p. 1–4. ISSN 2577-0829.

NASR-ESFAHANI, E. et al. Dense pooling layers in fully convolutional network for skin lesion segmentation. *Computerized Medical Imaging and Graphics*, Elsevier, v. 78, p. 101658, 2019.

OLIVEIRA, H. et al. Fully convolutional open set segmentation. *Machine Learning*, Springer, p. 1–52, 2021.

PASACHOFF, J. M.; FILIPPENKO, A. *The cosmos: Astronomy in the new millennium*. [S.l.]: Cambridge University Press, 2013.

PETER, A. H. et al. Wimp physics with ensembles of direct-detection experiments. *Physics of the Dark Universe*, Elsevier, v. 5, p. 45–74, 2014.

PETRIELLO, F. J.; ZUREK, K. M. Dama and wimp dark matter. *Journal of High Energy Physics*, IOP Publishing, v. 2008, n. 09, p. 047, 2008.

PHAN, N.; LEE, E.; LOOMBA, D. Imaging 55fe electron tracks in a gem-based tpc using a ccd readout. *Journal of Instrumentation*, IOP Publishing, v. 15, n. 05, p. P05012, 2020.

PINCI, D. et al. High resolution tpc based on optically readout gem. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 2018. ISSN 0168-9002. Disponível em: <http://www.sciencedirect.com/science/article/pii/S016890021831711X>.

PINCI, D. et al. Cygnus: development of a high resolution TPC for rare events. *PoS*, EPS-HEP2017, p. 077, 2017.

PINCI, D. et al. Cygnus: development of a high resolution tpc for rare events. 2018.

RAMCHOUN, H. et al. Multilayer perceptron: Architecture optimization and training. International Journal of Interactive Multimedia and Artificial Intelligence . . . , 2016.

RAZALI, N. M.; WAH, Y. B. et al. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*, v. 2, n. 1, p. 21–33, 2011.

REVELL, L. J. et al. Graphs in phylogenetic comparative analysis: Anscombe's quartet revisited. *Methods in Ecology and Evolution*, Wiley Online Library, v. 9, n. 10, p. 2145–2154, 2018.

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: SPRINGER. *International Conference on Medical image computing and computer-assisted intervention.* [S.l.], 2015. p. 234–241.

SANGIORGIO, S. et al. First demonstration of a sub-kev electron recoil energy threshold in a liquid argon ionization chamber. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, Elsevier, v. 728, p. 69–72, 2013.

SARKER, I. H. Deep cybersecurity: a comprehensive overview from neural network and deep learning perspective. *SN Computer Science*, Springer, v. 2, n. 3, p. 1–16, 2021.

SAULI, F. The gas electron multiplier (gem): Operating principles and applications. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, Elsevier, v. 805, p. 2–24, 2016.

SAULI, F. The gas electron multiplier (gem): Operating principles and applications. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, Elsevier, v. 805, p. 2–24, 2016.

SCHNEIDER, P. *Extragalactic astronomy and cosmology: an introduction.* [S.l.]: Springer, 2014.

SCHUMANN, M. Direct detection of wimp dark matter: concepts and status. *Journal of Physics G: Nuclear and Particle Physics*, IOP Publishing, v. 46, n. 10, p. 103003, 2019.

SHORTEN, C.; KHOSHGOFTAAR, T. M. A survey on image data augmentation for deep learning. *Journal of big data*, Springer, v. 6, n. 1, p. 1–48, 2019.

SKEIKA, E. L. et al. *Utilização de redes neurais completamente convolucionais para identificação e medição de crânios fetais.* Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, 2019.

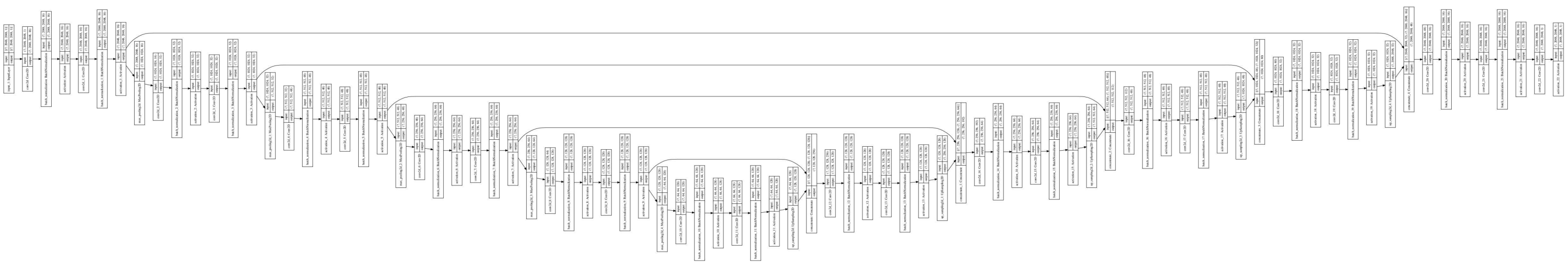
SMITH, M. C. et al. The rave survey: constraining the local galactic escape speed. *Monthly Notices of the Royal Astronomical Society*, Blackwell Publishing Ltd Oxford, UK, v. 379, n. 2, p. 755–772, 2007.

SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014.

ZHOU, J. et al. Multisignal vgg19 network with transposed convolution for rotating machinery fault diagnosis based on deep transfer learning. *Shock and Vibration*, Hindawi, v. 2020, 2020.

ZOU, Z. et al. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*, 2019.

APPENDIX A – U-NET ARCHITECTURE



APPENDIX B – PUBLICATION LIST

B.1 BOOK CHAPTER PUBLICATION

1. Lopes G.S.P. et al. (2019) Study of the Impact of Pre-processing Applied to Images Acquired by the Cygno Experiment. In: Morales A., Fierrez J., Sánchez J., Ribeiro B. (eds) Pattern Recognition and Image Analysis. IbPRIA 2019. Lecture Notes in Computer Science, vol 11868. Springer, Cham.

   This work proposes to evaluate the effect of digital filters when applied to images acquired by the ORANGE prototype of the CYGNO experiment. A preliminary analysis is presented in order to understand if filtering techniques can produce results that justify investing efforts in the pre-processing stage of those images. Such images come from a camera sensor based on CMOS technology installed in an appropriate gas detector. To perform the proposed work, a simulation environment was created and used to evaluate some of the classical filtering techniques known in the literature. The results showed that the signal-to-noise ratio of the images can be considerably improved, which may help in subsequent processing steps, such as clustering and particle identification.