UNIVERSIDADE FEDERAL DE JUIZ DE FORA
FACULDADE DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Igor Fonseca Pains

Desenvolvimento de Algoritmo de Trigger para Detecção de Sinais no Experimento CYGNO

Juiz de Fora
2024

Igor Fonseca Pains

Desenvolvimento de Algoritmo de Trigger para Detecção de Sinais no Experimento CYGNO

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétricada Universidade Federal de Juiz de Foracomo requisito parcial à obtenção do título de Mestre em Engenharia Elétrica. Área: Sistemas Eletrônicos.

Orientador: Prof. Dr. Rafael Antunes Nóbrega

Juiz de Fora

2024

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

Pains, Igor Fonseca.
Desenvolvimento de Algoritmo de Trigger para Detecção de Sinais no Experimento CYGNO / Igor Fonseca Pains. – 2024.
90 f. : il.

Orientador: Rafael Antunes Nóbrega
Tese (Mestrado) – Universidade Federal de Juiz de Fora, Faculdade de Engenharia. Programa de Pós-Graduação em Engenharia Elétrica, 2024.

1. Trigger. 2. CNNs. 3. Filtragem. 4. Processamento de imagem I. Nóbrega, Rafael Antunes, adv. III. Título.

Igor Fonseca Paíns

Desenvolvimento de Algoritmo de Trigger para Detecção de Sinais no Experimento CYGNO

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: Sistemas Eletrônicos

Aprovada em 04 de setembro de 2024.

BANCA EXAMINADORA

Prof. Dr. Rafael Antunes Nóbrega - Orientador
Universidade Federal de Juiz de Fora

Prof. Dr. Augusto Santiago Cerqueira
Universidade Federal de Juiz de Fora

Prof. Dr. Luciano Manhães de Andrade Filho
Universidade Federal de Juiz de Fora

Prof. Dr. Danton Diego Ferreira
Universidade Federal de Lavras

Juiz de Fora, 15/08/2024.

Documento assinado eletronicamente por **Rafael Antunes Nobrega**, **Professor(a)**, em 04/09/2024, às 14:58, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Augusto Santiago Cerqueira**, **Professor(a)**, em 04/09/2024, às 15:08, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Danton Diego Ferreira**, **Usuário Externo**, em 04/09/2024, às 15:09, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Luciano Manhaes de Andrade Filho**, **Professor(a)**, em 05/09/2024, às 10:13, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no Portal do SEI-Ufjf (www2.ufjf.br/SEI) através do ícone Conferência de Documentos, informando o código verificador **1926030** e o código CRC **98D5CB7B**.

AGRADECIMENTOS

Agradeço primeiramente a Deus pela saúde e força de vontade durante este caminho que escolhi trilhar.

Aos meus pais, Adilson e Rosângela, pelo afeto, carinho, apoio e incentivo durante todos estes anos.

Ao meu irmão, Lucas, pela amizade e companheirismo.

Ao meu orientador, Rafael, pelos ensinamentos, profissionalismo, dedicação e paciência que contribuíram para minha formação intelectual.

Aos demais companheiros do laboratório LAPTEL, pelos ensinamentos neste início de caminhada na área de pesquisas.

Ao CNPq, à CAPES, à Universidade Federal de Juiz de Fora e à Faculdade de Engenharia pelo suporte financeiro e por prover as ferramentas necessárias para o desenvolvimento deste trabalho.

RESUMO

O desenvolvimento de algoritmos de *trigger*, com tempos de processamento rápidos, é essencial para experimentos que processam grandes volumes de dados e buscam eventos raros com uma alta taxa de aquisição. O experimento CYGNO, projetado para a detecção direta de matéria escura por meio de um detector a gás, enfrenta o desafio de processar aproximadamente 80.000 imagens diariamente, muitas das quais contêm apenas ruído eletrônico gerado pelo sistema de leitura do detector. Para enfrentar esse desafio, este trabalho propõe dois algoritmos de *trigger*: um algoritmo utiliza filtragem com filtros padrões, como média, mediana e gaussiano, além do filtro casado; o outro é baseado em redes neurais convolucionais (CNNs). Ambos os métodos alcançaram uma taxa de detecção de aproximadamente 80% para sinais de 0.25 keV, com o algoritmo de filtragem apresentando uma taxa de falso alarme de 10%, enquanto a CNNs reduziu esse índice para 1%. Os algoritmos propostos foram capazes de detectar todos os sinais identificados pelo algoritmo de reconstrução do experimento, além de operar com tempos de processamento significativamente menores: o filtro gaussiano levou cerca de 200 milissegundos em CPU e 20 milissegundos em GPU, enquanto a CNN exigiu aproximadamente 550 milissegundos em CPU e 200 milissegundos em GPU. Esses avanços não apenas melhoram a eficiência da análise dos dados, mas também otimizam os recursos do experimento CYGNO, possibilitando uma gestão mais eficaz das imagens adquiridas.

Palavras-chave: Processamento de imagens. Trigger. Redes neurais convolucionais. Filtragem. Detecção de sinais. Eficiência computacional.

ABSTRACT

The development of trigger algorithms with fast processing times is essential for experiments that process large volumes of data and seek rare events with a high acquisition rate. The CYGNO experiment, designed for the direct detection of dark matter through a gas detector, faces the challenge of processing approximately 80,000 images daily, many of which contain only electronic noise generated by the camera and detector. To address this challenge, this work proposes two trigger algorithms: one algorithm uses filtering with standard filters such as mean, median, and gaussian, as well as the matched filter; the other is based on convolutional neural networks (CNNs). Both methods achieved a detection rate of approximately 80% for 0.25 keV signals, with the filtering algorithm yielding a false alarm rate of 10%, while the CNNs reduced this rate to 1%. The proposed algorithms were able to detect all signals identified by the experiment's reconstruction algorithm while operating with significantly lower processing times: the Gaussian filter took around 200 milliseconds on a CPU and 20 milliseconds on a GPU, whereas the CNN required approximately 550 milliseconds on a CPU and 200 milliseconds on a GPU. These advancements not only improve the efficiency of data analysis but also optimize the resources of the CYGNO experiment, enabling more effective management of the acquired images.

Keywords: Image processing. Trigger. Convolutional neural networks. Filtering. Signal detection. Computational efficiency.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ALU	Unidade Lógica Aritmética (do inglês, *Arithmetic Logic Unit*)
AUC	*Area Under Curve*
BCE	Cross-Entropia Binária (do inglês, *Binary Cross-Entropy*)
CNN	Rede Neural Convolucional (do inglês, *Convolutional Neural Network*)
DBSCAN	*Density-Based Spatial Clustering Algorithm with Noise*
DFT	Transformada Discreta de Fourier (do inglês, *Discrete Fourier Transform*)
DM	Matéria Escura (do inglês, *Dark Matter*)
EFTE	Etileno Tetrafluoretileno
EM	Eletromagnético
ER	Recuo Eletrônico (do inglês, *Electron Recoil*)
FFT	Transformada Rápida de Fourier (do inglês, *Fast Fourier Transform*)
FN	Falsos Negativos (do inglês, *False Negatives*)
FP	Falsos Positivos (do inglês, *False Positives*)
FPR	Taxa de Falsos Positivos (do inglês, *False Positive Ratio*)
GEM	*Gas Electron Multiplier*
GPU	Unidade de Processamento Gráfico (do inglês, *Graphic Processing Unit*)
iDDBSCAN	*intensity-based Directional DBSCAN*
IDFT	Transformada Discreta de Fourier Inversa (do inglês, *Inverse Discrete Fourier Transform*)
IFFT	Transformada Rápida de Fourier Inversa (do inglês, *Inverse Fast Fourier Transform*)
ILSVRC	*ImageNet Large Scale Visual Recognition Challenge*
INFN	*Istituto Nazionale di Fisica Nucleare*
LIME	*Long Imaging ModulE*
LNGS	*Laboratori Nazionali del Gran Sasso*
MPGD	*Micro-Pattern Gas Detectores*
MSGC	*Micro-Strip Gas Chamber*
MWPC	*Multi-Wire Proportional Chmaber*)
NR	Recuo Nuclear, (do inglês, *Nuclear Recoil*)
ReLU	Unidade Linear Retificada (do inglês, *Rectified Linear Unit*)
ROC	*Receiver Operating Characteristic*
sCMOS	*scientific Complementary Metal-Oxide-Semiconductor*
SNR	Relação Sinal-Ruído (do inglês, *Signal-to-noise Ratio*)
TN	Verdadeiros Negativos (do inglês, *True Negatives*)
TP	Verdadeiros Positivos (do inglês, *True Positives*)
TPC	Câmara de Projeção Temporal (do inglês, *Time Projection Chamber*)
TPR	Taxa de Verdadeiros Positivos (do inglês, *True Positive Ratio*)
WIMP	Partícula Massiva de Interação Fraca (do inglês, *Weekly Interacting Massive Particle*)

SUMÁRIO

1 INTRODUÇÃO

O processamento digital de imagens vem desempenhando um papel fundamental na análise e interpretação visuais em uma ampla gama de aplicações. Inicialmente foi muito utilizado em problemas envolvendo a medicina e astronomia, mas com os avanços tecnológicos das últimas décadas, principalmente com o desenvolvimento do computador digital, também passou a ser usado em outras áreas como na física, em experimentos de plasmas de alta energia e microscopia eletrônica (1).

Outro exemplo de uma aplicação mais recente na área da física que utiliza tais técnicas é a detecção de partículas de altas e baixas energias através de detectores a gás. No processo de interação partícula-gás, fótons provenientes da excitação das moléculas de gás presentes no detector são gerados e podem ser registrados em imagens através de uma câmera (2). Tais imagens contêm os rastros produzidos por essa interação, os quais podem ser de difícil detecção à medida que a energia da partícula incidente diminui.

Alguns casos como o experimento CYGNO (3, 4) têm como objetivo detectar eventos raros e de baixa energia e, portanto, precisam realizar a aquisição de imagens de alta resolução e baixo ruído. Técnicas de processamento de imagem e detecção de sinal podem ser empregadas para que esses dados sejam devidamente analisados, como métodos de filtragem e clusterização (5, 6).

No entanto, antes mesmo de realizar a análise desses dados, é de suma importância que a aquisição dos mesmos seja realizada de maneira adequada, considerando a ampla variabilidade de cenários do detector e do ambiente em que este está inserido. Entretanto, uma única imagem pode conter milhões de pixels, exigindo, assim, recursos significativos de memória para armazená-las em larga escala.

Um dos objetivos do experimento CYGNO é realizar a aquisição contínua de dados em um futuro próximo, com uma taxa próxima a 1 Hertz, resultando em aproximadamente 80.000 imagens por dia. Adicionalmente, o projeto prevê a instalação do detector em um ambiente com baixo nível de ruído de fundo além de um sistema de blindagem feito com água e chumbo (7). Assim, é esperado que muitas dessas imagens não apresentem nenhum tipo de sinal relevante, sendo dominadas exclusivamente pelo ruído eletrônico proveniente do sistema de leitura do detector.

Neste contexto, o objetivo deste trabalho é desenvolver um algoritmo de *trigger online* para o experimento CYGNO, cujo propósito é selecionar, em um curto intervalo de tempo, as imagens que contenham algum tipo de sinal relevante, enquanto descarta aquelas que apresentam apenas ruído eletrônico. Para isso, serão abordados dois métodos: o primeiro utiliza técnicas de filtragem, incluindo filtros clássicos de suavização como média, mediana e gaussiano, além do filtro casado. O filtro casado é amplamente utilizado na literatura para detecção de sinal e emprega o próprio sinal esperado para sua detecção

na imagem. A escolha do filtro gaussiano é motivada tanto pela semelhança do formato do sinal de baixa energia com uma gaussiana 2D, aproximando-se de um filtro casado, quanto por ser um filtro de suavização, assim como os filtros de média e mediana, que possuem a capacidade de suavizar o ruído e potencialmente realçar o sinal. A segunda abordagem explora uma alternativa baseada em aprendizado de máquina, implementando uma rede neural convolucional (CNN). Ambas as abordagens serão comparadas entre elas e com o algoritmo de reconstrução atualmente utilizado no experimento para análises *offline*, considerando a eficácia na detecção de eventos reais e simulados, bem como o tempo de processamento.

1.1 ESTRUTURA DA DISSERTAÇÃO

Este trabalho está organizado da seguinte forma: o capítulo 2 apresentará uma revisão bibliográfica sobre técnicas de processamento de imagem e CNNs. A detecção de matéria escura, o experimento CYGNO e os bancos de dados serão detalhados no capítulo 3. A metodologia adotada neste trabalho será definida no capítulo 4 e, por fim, os resultados finais e conclusão serão mostrados nos capítulos 5 e 6, respectivamente.

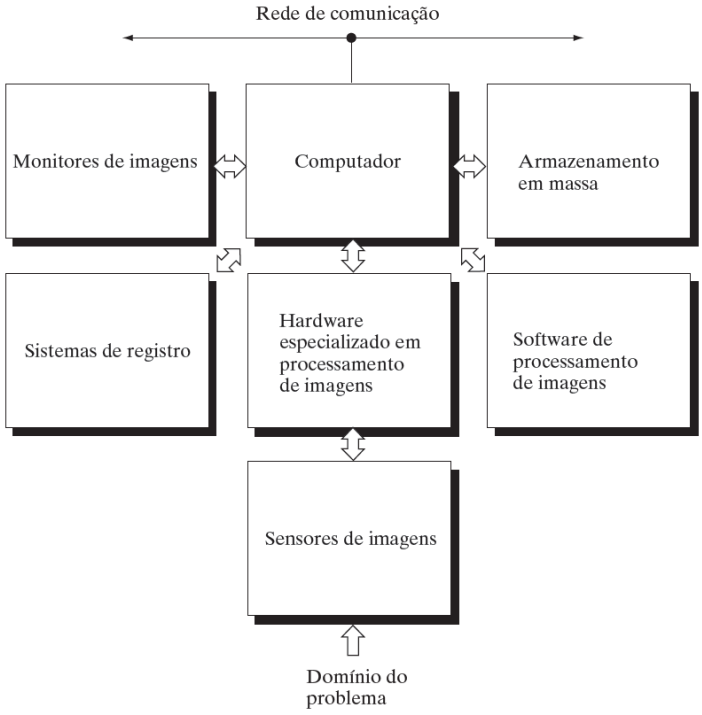
2 REVISÃO BIBLIOGRÁFICA

2.1 PROCESSAMENTO DIGITAL DE IMAGENS

A visão humana é considerada o sentido mais avançado e desempenha um papel fundamental na percepção, ao passo de que não é de se surpreender que as imagens tenham um papel significativo nesta. No entanto, ao contrário dos seres humanos, cuja visão está limitada à faixa visual do espectro eletromagnético (EM) com comprimentos de onda entre 350 e 780 nanômetros (nm), os dispositivos de captação de imagens têm a capacidade de explorar uma ampla gama do espectro EM, abrangendo desde ondas gama até ondas de rádio (1).

A ampla cobertura espectral fornecida por esses aparelhos faz com que seja possível lidar com dados provenientes de fontes que os humanos não estão acostumados a associar com imagens. Exemplos dessas fontes incluem ultrassom, microscopia eletrônica, imagens geradas por computador e, especificamente neste trabalho, por detectores de partículas. Neste contexto, a Figura 1 mostra os componentes básicos que constituem um sistema de uso geral típico para o processamento digital de imagens. Os componentes mais importantes para a aplicação apresentada neste trabalho estão definidos abaixo:

Figura 1 – Componentes de um sistema de processamento de imagens de uso geral.



Fonte: Extraída de (1)

- **Sensores de imagens:** Existem dois elementos necessários para a aquisição de imagens digitais. O primeiro consiste em um dispositivo físico que seja sensível à

energia irradiada pelo objeto ou fenômeno que se pretende capturar. Já o segundo, conhecido como digitalizador, desempenha o papel de converter o sinal elétrico gerado pelo sensor em um formato digital.

- **Hardware especializado em processamento de imagens:** É composto essencialmente pelo digitalizador, mencionado anteriormente, e por um hardware que possui a capacidade de executar operações aritméticas e lógicas básicas em paralelo em toda a imagem, chamada ALU - *Arithmetic Logic Unit* (Unidade Lógica Aritmética). Essa unidade desempenha funções que requerem um processamento de alta velocidade, o que pode ser desafiador para um computador convencional.
- **Computador:** Um computador de uso geral que deve realizar as tarefas de processamento de imagens. Dependendo da aplicação, um computador pessoal bem equipado é capaz de realizar tais operações *offline*.
- **Armazenamento em massa:** É indispensável para este tipo de aplicação. Uma imagem que possui, por exemplo, 2304x2304 pixels, na qual cada pixel requer 8 bits, necessita de cerca de 5 megabytes para ser armazenada caso não seja comprimida. Dessa forma, a gestão adequada do armazenamento em larga escala desses dados pode representar um desafio em diversas aplicações.

De forma geral, as técnicas de processamento de imagens podem ser divididas em duas categorias mais abrangentes: aquelas em que as entradas e saídas são imagens e aquelas em que as entradas são imagens, mas as saídas consistem em atributos ou características derivadas delas. A filtragem é um exemplo da primeira categoria, enquanto a segunda categoria pode englobar algoritmos de seleção baseados nas imagens originais ou filtradas. Nas próximas seções, serão discutidas as definições de uma imagem digital, técnicas de filtragem espacial e em frequência, além da detecção de sinal por meio de filtros casados e o uso de CNNs para análise e detecção de sinais em imagens.

2.1.1 IMAGEM DIGITAL

No projeto e análise de sistemas de processamento de imagens, é conveniente e muitas vezes necessário caracterizar matematicamente a imagem a ser processada. Esta pode ser definida como uma função bidimensional, *f(x,y)*, em que x e y são coordenadas espaciais (plano), e a amplitude de f em cada par dessas coordenadas (pixel) é chamada de intensidade (1).

Quando uma imagem é produzida a partir de um fenômeno físico, seus valores de intensidade são proporcionais à energia irradiada por uma fonte real (por exemplo, ondas eletromagnéticas). Além disso, o próprio sistema físico impõe restrições para um valor máximo de intensidade em uma imagem (8). Consequentemente, *f(x,y)* deve assumir valores diferentes de zero e finitos 2.1:

$$0 < f(x, y) < \infty \tag{2.1}$$

Neste contexto, a saída da maioria dos sensores consiste de uma forma de onda de tensão contínua pelo espaço, ou seja, as variáveis x, y e *f(x,y)* são contínuas. Para se criar uma imagem digital, é necessário converter todos os dados captados para o formato digital. Os principais processos utilizados nesta conversão são a amostragem e quantização, para discretizar os valores das coordenadas espaciais e de intensidade, respectivamente. Desta forma, as imagens digitais podem ser representadas matematicamente através de matrizes, como a imagem MxN descrita na equação a seguir.

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

Imagens coloridas comumente associam cada pixel a um conjunto de valores, que denotam o nível de intensidade em cada escala de cor escolhida. Nas imagens utilizadas neste trabalho e em diversas outras aplicações, cada pixel tem intensidade associada a apenas uma escala de cor. Estas normalmente são chamadas de imagens em escala de cinza.

Outro conceito importante no âmbito do processamento de imagens é a chamada vizinhança. A partir da Equação 2.2, pode-se observar que os pixels representados por *f(0,0)* e *f(0,1)* estão lado a lado no sentido horizontal, e, portanto, são considerados vizinhos horizontais. Já o pixel p, representado por *f(1,1)*, possui 8 pixels ao seu redor, considerando os sentidos horizontal, vertical e diagonal. Tal região é chamada de *vizinhança-8* de p, que faz parte de um quadrado de lado 3 centrado em tal pixel.

Estas definições são de extrema importância e servem como base para as técnicas que serão descritas no decorrer deste trabalho. A próxima seção discutirá o tópico de filtragem espacial.

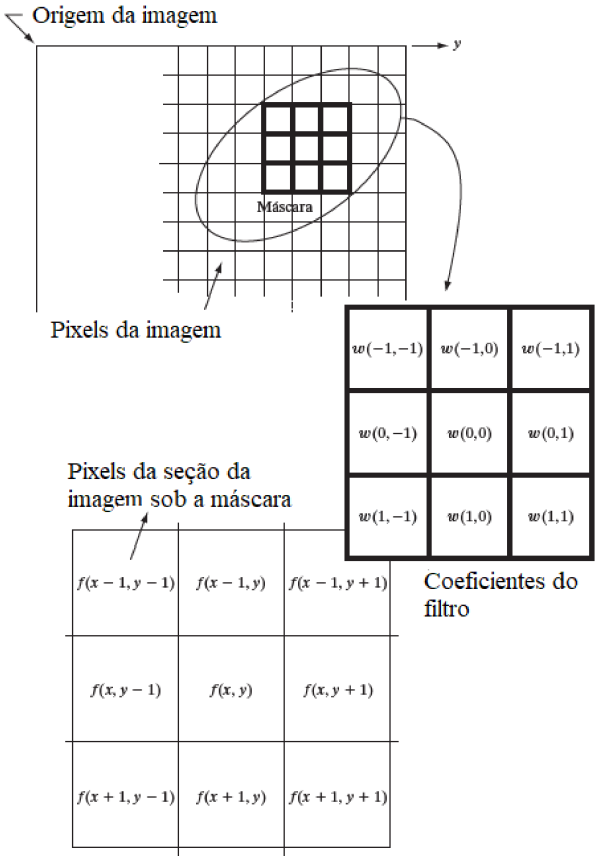
2.1.2 FILTRAGEM ESPACIAL

A filtragem espacial, também conhecida como processamento de vizinhança, consiste na substituição de cada pixel em uma imagem pelo resultado de uma operação que leva em consideração tanto a sua própria intensidade quanto a dos seus vizinhos (9). Algoritmos deste tipo podem ser classificados em duas categorias de acordo com o tipo de operação realizada: lineares e não-lineares. Nos filtros lineares, são empregadas operações como

média ou ponderação, enquanto os filtros não-lineares utilizam operações mais complexas, como mediana ou moda.

O conceito de filtragem espacial linear pode ser exemplificado na Figura 2 e na Equação 2.3. O filtro, também conhecido como máscara, percorre todas as posições da imagem de entrada. Cada pixel da imagem de saída, denominada *f'(x,y)*, possui uma intensidade calculada como uma soma ponderada entre a máscara e a vizinhança ao redor do pixel. Observa-se que o pixel representado por *f'(x,y)* é o resultado dessa operação quando o centro da máscara, *w(0,0)*, coincide com *f(x,y)*. Por essa razão, as máscaras de tamanho ímpar são mais comumente utilizadas, uma vez que proporcionam uma interpretabilidade maior.

Figura 2 – O conceito de filtragem espacial linear utilizando uma máscara w.



Fonte: Extraída de (1)

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

O procedimento ilustrado anteriormente pode ser representado de forma elegante pela Equação 2.4, amplamente utilizada na literatura (1). Nessa equação, o símbolo

$\circledast$ denota a correlação espacial entre a máscara *w(x,y)* de tamanho MxN e a imagem *f(x,y)*. Os somatórios abrangem todas as possíveis coordenadas da máscara *w(x,y)*, com as variáveis i e j variando no intervalo de $-(M-1)/2$ a $(M-1)/2$.

$$f'(x,y) = w(x,y) \circledast f(x,y) = \sum_{i=-(M-1)/2}^{(M-1)/2} \sum_{j=-(N-1)/2}^{(N-1)/2} w(i,j) f(x+i, y+j) \tag{2.4}$$

Outra forma de representar a filtragem espacial é por meio da operação de convolução, como ilustrado na Equação 2.5. Essa operação é semelhante à correlação, com a diferença de que a máscara w é rotacionada em 180° antes de ser aplicada. Enquanto alguns teoremas fundamentais da filtragem em frequência utilizam a convolução, a correlação é frequentemente empregada na abordagem de detecção como ocorre com o filtro casado. Ambos os tópicos serão abordados em detalhes nas próximas seções.

$$g(x,y) = w(x,y) * f(x,y) = \sum_{i=-(M-1)/2}^{(M-1)/2} \sum_{j=-(N-1)/2}^{(N-1)/2} w(i,j) f(x-i, y-j) \tag{2.5}$$

Ainda no contexto de filtragem espacial, os filtros não-lineares seguem o mesmo princípio demonstrado na Figura 2. No entanto, ao contrário dos filtros lineares, sua operação não pode ser representada por meio de somatórios. Em vez disso, emprega-se operações não-lineares, como a moda, que busca o valor de intensidade mais frequente, ou o máximo, que busca o maior valor de intensidade, dentro da região de sobreposição entre a máscara e a imagem.

O objetivo principal deste trabalho com a utilização destes filtros é realizar a redução do ruído presente nas imagens. Os filtros que possuem essa característica são denominados filtros de suavização. Para uma análise comparativa, foram selecionados dois filtros lineares e um filtro não-linear, que serão apresentados nas seções seguintes.

2.1.2.1 FILTROS LINEARES DE SUAVIZAÇÃO

Os filtros lineares de suavização, também conhecidos como filtros de média ou filtros passa-baixa, desempenham um papel crucial no processamento de imagens para redução de ruído. Suas outras denominações vêm do fato da saída desses filtros ser constituída, basicamente, por uma média da vizinhança de cada pixel da imagem de entrada, e da sua habilidade em rejeitar componentes de alta frequência (1).

Um exemplo simples de filtro linear de suavização é o filtro de média, cuja máscara correspondente, de tamanho MxN, é mostrada na Equação 2.6. A ideia por trás desse filtro é somar todas as intensidades dos pixels em uma vizinhança e dividir pelo número de elementos na mesma.

$$w(x,y) = \frac{1}{M.N}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{2.6}$$

Como mencionado na Seção **2.1.2**, é comum usar máscaras de tamanho ímpar para garantir que haja um pixel central na máscara. Além disso, máscaras quadradas são frequentemente utilizadas para explorar a simetria das vizinhanças de cada pixel. A Equação 2.7 mostra um exemplo de máscara quadrada com um lado de comprimento W, também conhecido como tamanho da janela. Esse parâmetro será amplamente explorado neste trabalho.

$$w(x,y) = \frac{1}{W^2}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{2.7}$$

Outro filtro de suavização muito utilizado na literatura é o filtro gaussiano. A ideia por trás deste filtro é criar uma máscara onde as intensidades são definidas a partir da função bidimensional gaussiana, descrita na Equação 2.8. O fato desta função possuir comportamento similar e não oscilatório tanto no domínio espacial quanto no domínio de frequência a torna ideal para ser aplicada em filtros de imagens (10).

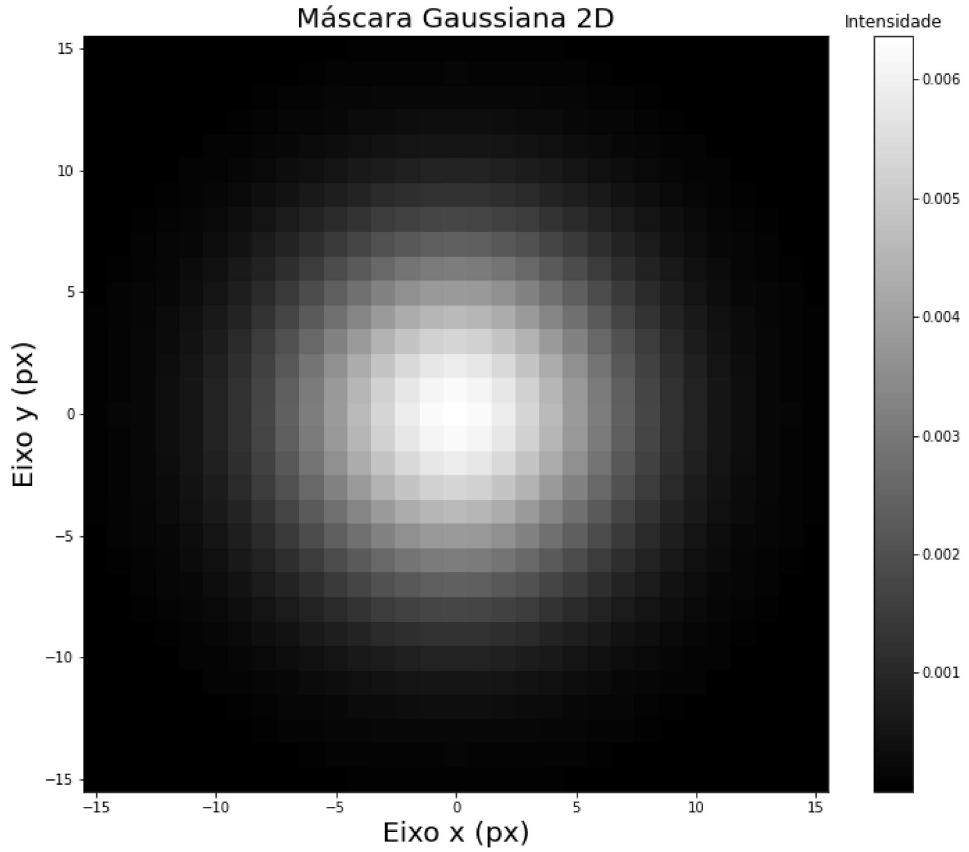
$$G(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}e^{-\frac{x^2}{2\sigma_x^2}-\frac{y^2}{2\sigma_y^2}} \tag{2.8}$$

Por simplicidade, é comum considerar o caso simétrico na fórmula da função gaussiana, onde as variâncias σ_x^2 e σ_y^2 são iguais. Assim, a máscara gaussiana w pode ser definida apenas em termos de σ^2, como mostrado na Equação 2.9. O tamanho da janela W é utilizado para truncar a função gaussiana, visto que ela é definida para todo o eixo real. A Figura 3 ilustra um exemplo visual dessa máscara com $\sigma = 5$ e $W = 31$. É importante observar que quanto mais distante um pixel está da região central da máscara, menor é sua contribuição ponderada na média realizada durante a filtragem, resultando em um peso menor em comparação com os pixels do centro.

$$G(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{2.9}$$

Uma característica única das duas máscaras apresentadas nesta seção é a simetria tanto horizontal quanto vertical. Essa propriedade resulta em um comportamento equivalente quando as máscaras são aplicadas tanto na operação de correlação quanto na de convolução, discutidas na Seção **2.1.2**. Por uma questão de conveniência, neste trabalho a

Figura 3 – Exemplo de máscara gaussiana de área normalizada com $\sigma = 5$ e $W = 31$.



Fonte: Elaborada pelo autor (2024).

filtragem espacial com os filtros lineares será realizada utilizando a fórmula da convolução, como representada na Equação 2.5.

2.1.2.2 FILTROS NÃO-LINEARES DE SUAVIZAÇÃO

Os filtros não-lineares de suavização mais utilizados são aqueles baseados no conceito de estatística de ordem, nos quais os pixels em uma vizinhança são ordenados e analisados. Esses filtros são comumente chamados de filtros de estatística de ordem (1). Um exemplo amplamente conhecido é o filtro de mediana, que, como o próprio nome sugere, realiza a operação de mediana sobre as vizinhanças da imagem de entrada.

A ideia por trás deste método é localizar na imagem os pixels que possuem intensidades extremas e, portanto, altamente improváveis e ignorar suas intensidades reais, substituindo-as por valores mais adequados (10). Por esta razão, este filtro é particularmente eficaz em lidar com o ruído impulsivo ou com o ruído *salt-and-pepper*.

Assim como os filtros apresentados na Seção *2.1.2.1*, o filtro de mediana é parametrizado pelo tamanho de janela W. Considerando o caso mais comum, onde W é ímpar, os valores de intensidade da vizinhança de um pixel p são ordenados em ordem crescente,

definidos pela função *f(i)*, onde $i = 0$ e $i = W^2 - 1$ representam os pixels de menor e maior intensidade respectivamente. Desta forma, cada pixel na imagem de saída terá intensidade I_m igual a Equação 2.10.

$$I_m = f(\frac{W^2 - 1}{2}) \tag{2.10}$$

O filtro de mediana é uma ferramenta poderosa para lidar com ruído em imagens. No entanto, ele apresenta algumas desvantagens que devem ser consideradas. Um dos principais pontos negativos é o seu alto custo computacional, especialmente devido à necessidade de ordenar as W^2 intensidades presentes em cada vizinhança, mesmo se algoritmos altamente otimizados forem usados (11). Esse custo computacional pode limitar sua aplicação em tempo real ou em imagens de grande escala.

Além disso, é importante ressaltar que os filtros não-lineares não podem ser implementados por meio da convolução, o que implica na impossibilidade de aproveitar uma das principais propriedades de filtragem no campo de processamento de imagens: a relação direta entre filtragem no domínio espacial e no domínio da frequência. Essa relação será discutida em detalhes na próxima seção.

2.1.3 FILTRAGEM NO DOMÍNIO DA FREQUÊNCIA

O conceito de filtro, amplamente abordado neste trabalho, na verdade tem origem no processamento no domínio da frequência. Um exemplo disso é o filtro passa-baixa, que permite a passagem das componentes de baixa frequência em um sinal, enquanto atenua as demais. Um entendimento aprofundado no campo do processamento digital de imagens é impossível sem antes compreender o uso da transformada de Fourier e o domínio da frequência. Desta forma, as Seções *2.1.3.1* e *2.1.3.2* discutirão alguns desses conceitos e propriedades e o uso de filtros de suavização no domínio de frequência, respectivamente.

2.1.3.1 TRANSFORMADA DE FOURIER

O matemático francês Jean Baptiste Joseph Fourier fez diversas contribuições que são aplicadas até hoje em diversas áreas. Seu trabalho mais famoso estabelece que qualquer função periódica pode ser expressa como uma soma de senos e cossenos de diferentes frequências, conhecida como Séries de Fourier (12). Essa propriedade foi posteriormente estendida para funções não periódicas que atendem a certos requisitos, como possuir uma área finita, resultando na Transformada de Fourier.

No campo do processamento digital de imagens, esta transformada é muito utilizada em sua versão bidimensional discreta, a chamada Transformada Discreta de Fourier (DFT) 2-D, expressa na Equação 2.11, onde $j = \sqrt{-1}$. Desta forma, a imagem digital *f(x,y)* de

dimensões MxN pode ser avaliada no domínio da frequência através das variáveis discretas u e v.

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{2.11}$$

A DFT é uma chamada transformada unitária, pois a operação descrita em 2.11 é invertível e satisfaz certas condições de ortogonalidade (8). Esta é uma propriedade muito útil, visto que diversos processamentos podem ser feitos no domínio da frequência e o resultado ser avaliado novamente no domínio espacial. A chamada Transformada Discreta de Fourier Inversa (IDFT) é a operação que faz essa transição do domínio da frequência para o domínio espacial, e pode ser verificada na equação 2.12.

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{2.12}$$

Essas duas formulações constituem o par de Transformadas Discretas de Fourier 2-D e possuem diversas propriedades importantes que relacionam a representação de imagens digitais no domínio espacial e no domínio da frequência. Uma das propriedades mais importantes no contexto deste trabalho é o teorema da convolução 2-D. Esse teorema estabelece que a operação de convolução entre uma imagem $f(x,y)$ e uma máscara $w(x,y)$ no domínio espacial pode ser descrita como a operação de multiplicação entre suas respectivas DFTs, $F(u,v)$ e $W(u,v)$, no domínio da frequência. Essa relação é expressa pela Equação 2.13.

$$f(x,y) * w(x,y) \quad \Leftrightarrow \quad F(u,v) \cdot W(u,v) \tag{2.13}$$

Nesse contexto, a operação de filtragem discutida na Seção *2.1.2.1* pode ser totalmente realizada no domínio da frequência, o que pode ser computacionalmente mais eficiente, dependendo da aplicação. Essa abordagem é viável graças à descoberta da Transformada Rápida de Fourier (FFT), que proporciona uma redução significativa na complexidade computacional necessária para a transição entre os domínios (13). Por essa razão, todas as implementações apresentadas neste trabalho que envolvem as equações discutidas nesta seção utilizarão a FFT.

2.1.3.2 SUAVIZAÇÃO DE IMAGENS NO DOMÍNIO DA FREQUÊNCIA

A correspondência entre os filtros lineares de suavização e a filtragem no domínio da frequência se torna intuitiva ao analisar o espectro gerado pela DFT. Como já dito anteriormente, o principal objetivo desses filtros é reduzir as transições abruptas causadas

pelo ruído em imagens, que geralmente contribuem para o conteúdo de alta frequência. Portanto, uma solução lógica é utilizar filtros passa-baixa que possam atenuar essas componentes, preservando as de baixa frequência (1).

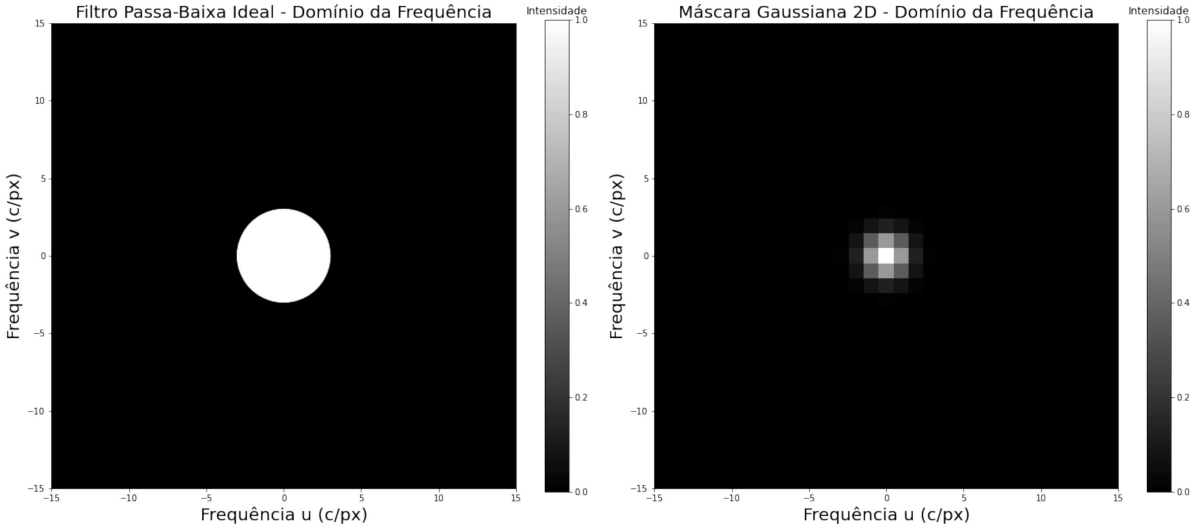
Existem diferentes representações para o espectro 2-D de frequências na literatura, sendo a mais comum aquela em que a origem e as componentes de alta frequência são localizadas no centro e nas bordas da imagem, respectivamente. Nessa representação, um filtro passa-baixa ideal pode ser visualizado como um círculo de raio D_0 centrado na origem. A região interna do círculo é conhecida como a banda de passagem, enquanto o restante é rejeitado. As Equações 2.14 e 2.15 apresentam a definição matemática desse filtro, considerando um tamanho MxN.

$$H(u,v) = \begin{cases} 1, & \text{se } D(u,v) \leq D_0 \\ 0, & \text{se } D(u,v) > D_0 \end{cases} \tag{2.14}$$

$$D(u,v) = \sqrt{(u - \frac{M}{2})^2 + (v - \frac{N}{2})^2} \tag{2.15}$$

Na prática, este filtro teórico não pode ser implementado devido a limitações computacionais. No entanto, seu comportamento se assemelha, por exemplo, ao filtro gaussiano apresentado na Seção *2.1.2.1*. A Figura 4 mostra uma comparação entre um filtro passa-baixa ideal com $D_0 = 3$ e o filtro gaussiano já apresentado, com $\sigma = 5$, no domínio da frequência, ambos quadrados com $W = 31$.

Figura 4 – Exemplo de filtro passa-baixa ideal com $D_0 = 3$ (esquerda) e filtro gaussiano com $\sigma = 5$ e $W = 31$ (direita) no domínio da frequência.



Fonte: Elaborada pelo autor (2024).

Além disso, os filtros projetados no domínio espacial geralmente têm dimensões significativamente menores do que a imagem a ser filtrada, especialmente quando a imagem contém informações de vários objetos. Portanto, antes de aplicar a DFT em um filtro espacial, sua matriz deve ser preenchida com zeros para que tenha o mesmo tamanho da imagem a ser filtrada. No entanto, é importante considerar que esse preenchimento pode introduzir distorções na DFT do filtro, especialmente se a matriz for preenchida com muitos zeros.

Por exemplo, o filtro de média possui uma DFT igual a um impulso centrado na origem, independentemente do parâmetro W utilizado. No entanto, ao aplicar o preenchimento com zeros, sua DFT apresenta algumas distorções que modifica um pouco este comportamento. Isso pode ser observado na Figura 5, onde dois filtros de média com $W = 5$ e $W = 31$ tiveram seus tamanhos modificados para filtrar uma imagem de tamanho 101x101. Consequentemente, é de suma importância projetar um filtro com o valor adequado de W, considerando também a imagem que será filtrada.

Filtros podem ser projetados diretamente no domínio da frequência, como é o caso, por exemplo, do filtro Butterworth (1, 14). Entretanto, o projeto de filtros em frequência pode ser um desafio e requer um profundo conhecimento dos dados neste domínio, não sendo esta a proposta deste trabalho. Este trabalho se concentrará nos filtros mencionados na Seção **2.1.2** e utilizará as propriedades da DFT e FFT apenas para a implementação da filtragem. Essas propriedades também servem como base teórica para o conceito de filtros casados, que será discutido na próxima seção.

2.1.4 DETECÇÃO DE SINAL ATRAVÉS DE FILTROS CASADOS

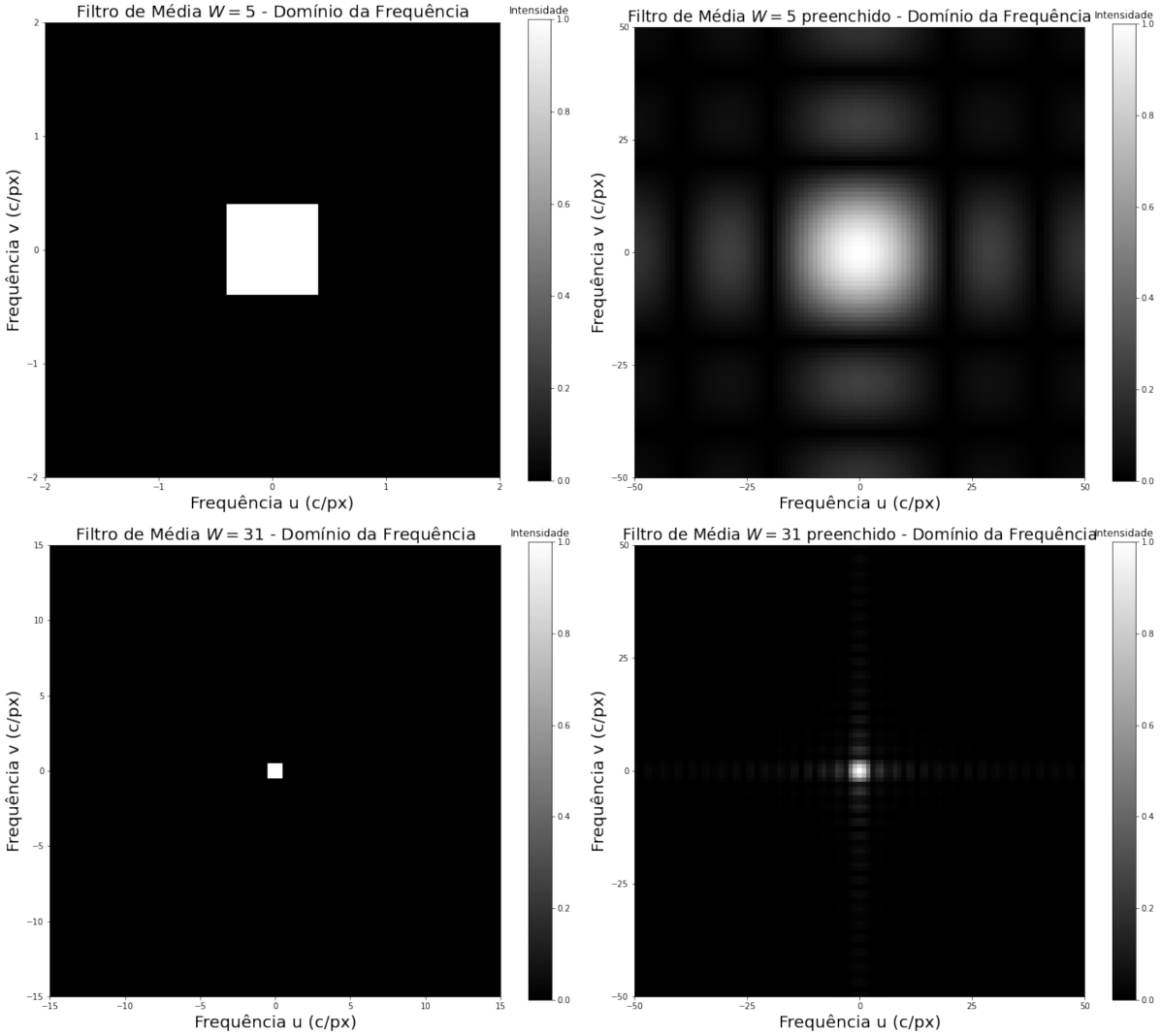
A detecção de sinais é uma área de grande relevância no campo do processamento de imagens e reconhecimento de padrões. Essa problemática pode ser dividida basicamente em duas situações distintas: aquelas em que não há informações prévias sobre o sinal a ser detectado e casos em que existe um conhecimento geral sobre a forma do sinal. No último caso, técnicas como os filtros casados se mostram altamente eficientes para a detecção, desde que algumas condições sejam atendidas (15, 16, 17).

Essa abordagem é fundamentada na análise espectral tanto do sinal presente na imagem quanto do ruído, utilizando as técnicas discutidas na Seção *2.1.3.2*. Considerando uma imagem $f(x, y)$ esteja corrompida por um ruído aditivo independente $n(x, y)$, define-se a imagem de entrada $f_i(x, y)$ (vide Equação 2.16), que passará pelo filtro casado $h(x, y)$ e gerará a imagem de saída $f_o(x, y)$, conforme Equação 2.17.

$$f_i(x, y) = f(x, y) + n(x, y) \tag{2.16}$$

$$f_o(x, y) = f_i(x, y) * h(x, y) \tag{2.17}$$

Figura 5 – Exemplo de dois filtros de média com $W = 5$ (cima) e $W = 31$ (baixo) em seus tamanhos originais, à esquerda, e preenchidos com zeros para filtrar uma imagem de tamanho 101x101, à direita.



Fonte: Elaborada pelo autor (2024).

A ideia principal do filtro casado é projetá-lo de maneira a maximizar a relação entre o espectro de energia do sinal e do ruído em um ponto específico (x_o, y_o) após a filtragem. As Equações 2.18 e 2.19 descrevem a energia instantânea do sinal na ausência de ruído, e a Equação 2.20 define a energia do ruído com densidade espectral $\mathcal{W}_n(u, v)$, assumindo-se que o ruído seja estacionário, ambas após a aplicação do filtro $H(u, v)$ (8).

$$|S(x_o, y_o)|^2 = |f(x, y) * h(x, y)|^2 \tag{2.18}$$

$$|S(x_o, y_o)|^2 = |\frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)H(u,v)e^{j2\pi\left(\frac{ux_o}{M}+\frac{vy_o}{N}\right)}|^2 \tag{2.19}$$

$$N_e = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} \mathcal{W}_n(u,v)|H(u,v)|^2 \tag{2.20}$$

Dessa forma, a relação a ser maximizada, conhecida como relação sinal-ruído (SNR), pode ser representada pela Equação 2.21. O filtro $H(u, v)$ que maximiza a SNR está relacionado com o complexo conjugado do sinal e o espectro de energia do ruído, conforme mostrado na Equação 2.22 (8). No caso ideal, em que o ruído é branco gaussiano, essa última parcela é essencialmente uma constante em todo o espectro ($\frac{n_w}{2}$), e a resposta ao impulso do filtro casado ideal $h(x, y)$ se torna uma versão do sinal $f(x, y)$ rotacionada em 180° e multiplicado por uma constante, como mostrado na Equação 2.23.

$$SNR = \frac{|\frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)H(u,v)e^{j2\pi\left(\frac{ux_o}{M}+\frac{vy_o}{N}\right)}|^2}{\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\mathcal{W}_n(u,v)|H(u,v)|^2} \tag{2.21}$$

$$H(u,v) = \frac{F^*(u,v)e^{-j2\pi\left(\frac{ux_o}{M}+\frac{vy_o}{N}\right)}}{\mathcal{W}_n(u,v)} \tag{2.22}$$

$$h(x,y) = \frac{2}{n_w} f(x_o - x, y_o - y) \tag{2.23}$$

Conforme apresentado na Seção 2.3, as operações de convolução e correlação se distinguem pela rotação de 180° das máscaras. Dessa maneira, o filtro casado pode ser empregado por meio da correlação espacial entre o objeto a ser detectado e uma imagem. Cada pixel na imagem de saída representa a medida de similaridade entre a máscara e a região sobreposta por ela na imagem original quando centrada nesse pixel. Portanto, o filtro casado é aquele que pode alcançar a maior correlação possível caso uma cópia exata dessa máscara esteja presente na imagem.

No entanto, é fundamental destacar que na prática nem todas as condições impostas ao ruído são plenamente satisfeitas, mesmo que em algumas aplicações ele se aproxime do caso ideal. Além disso, certos problemas podem envolver sinais com comportamento conhecido, mas não determinístico. Apesar desses desafios, o filtro casado tem a capacidade de fornecer resultados altamente satisfatórios, como será comprovado ao longo deste trabalho.

2.2 REDES NEURAIS CONVOLUCIONAIS

O termo rede neural tem suas origens em tentativas de encontrar representações matemáticas do processamento de informações em sistemas biológicos (18, 19, 20, 21). De fato, o termo tem sido usado de forma ampla para cobrir uma vasta gama de modelos diferentes, muitos dos quais foram objeto de reivindicações exageradas quanto à sua plausibilidade biológica. No entanto, do ponto de vista das aplicações práticas de reconhecimento de padrões, o realismo biológico imporia restrições completamente desnecessárias (22).

No contexto do processamento de imagens, uma das arquiteturas mais eficazes é a rede neural convolucional (CNN), que utiliza a operação de convolução para identificar padrões em imagens (23). Esta operação capitaliza várias ideias importantes que podem melhorar significativamente o desempenho de uma rede neural: primeiro, a alta correlação entre pixels próximos de um sinal em uma imagem facilita sua identificação quando avaliados em conjunto; segundo, o uso de máscaras pequenas em relação à imagem permite a detecção de características detalhadas como bordas; e terceiro, a invariância a deslocamento, onde a operação de convolução percorre a imagem para detectar características independentemente de sua posição (24).

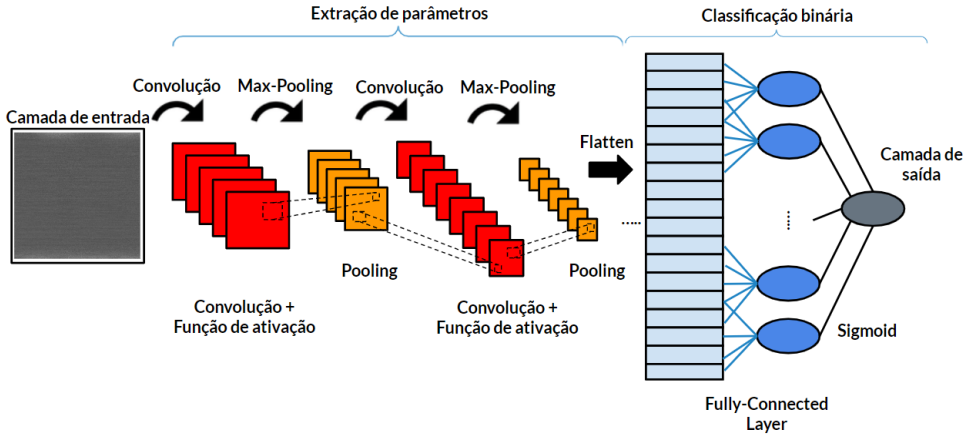
Além disso, diferentemente do filtro casado apresentado anteriormente, as CNNs não requerem que o usuário forneça conhecimento específico sobre os dados. Elas são capazes de aprender diretamente a partir dos exemplos fornecidos durante o treinamento, adaptando-se automaticamente às características relevantes para reconhecer possíveis padrões nas imagens. Para isso, a rede é exposta a dados de cada classe que ela deve identificar, e utiliza métodos baseados em gradientes para ajustar seus parâmetros de modo a minimizar um erro definido, alcançando assim uma configuração ótima para classificar novas imagens de forma adequada (25).

2.2.1 COMPONENTES DE REDES NEURAIS CONVOLUCIONAIS

Apesar da convolução ser de suma importância para as CNNs, ela não é a única operação utilizada na arquitetura de uma. A Figura 6 ilustra uma configuração típica de uma CNN, mostrando o fluxo desde a imagem de entrada até a saída da rede, que indica a probabilidade da imagem pertencer a uma classe específica.

Ao observar a Figura 6, pode-se notar que existem basicamente duas etapas nessa CNN: a extração de parâmetros e a classificação binária. A primeira etapa é composta principalmente por camadas de convolução, que podem ou não possuir uma função de ativação, seguidas sempre de camadas de *Max-Pooling*. Essa combinação é poderosa para identificar padrões enquanto diminui a dimensão das imagens (24). Na segunda etapa, todos os parâmetros extraídos até a última camada de *Max-Pooling* são conectados a neurônios na chamada *Fully-Connected Layer*, que então alimenta a camada de saída responsável pela classificação final da imagem. Esses componentes serão melhor explicados

Figura 6 – Exemplo de uma arquitetura de CNN para classificar se uma imagem possui sinal.



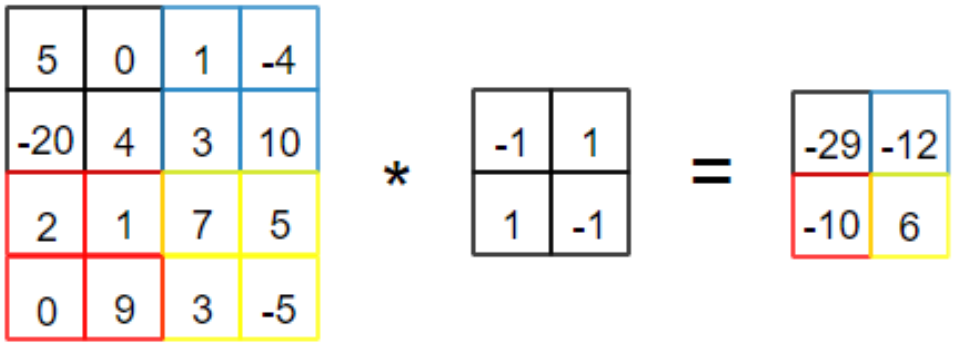
Fonte: Elaborada pelo autor (2024).

nas próximas seções.

2.2.1.1 CAMADA DE CONVOLUÇÃO

Como mencionado na Seção **2.2.1**, a camada de convolução é a principal responsável por identificar padrões na imagem a partir da operação de convolução. A Figura 7 ilustra um exemplo dessa operação entre uma imagem e uma máscara.

Figura 7 – Exemplo de aplicação de camada de convolução.



Fonte: Elaborada pelo autor (2024).

Na CNN, alguns parâmetros definem como a operação de convolução será realizada, sendo os mais importantes o número de máscaras, o tamanho das máscaras e o *stride*. Os dois primeiros parâmetros são intuitivos, determinando a quantidade de máscaras utilizadas em cada camada de convolução e o tamanho de cada uma. Naturalmente, um número maior de máscaras aumenta a complexidade da CNN, pois cada máscara buscará uma característica específica na imagem. Um tamanho de máscara maior permite que

pixels mais distantes influenciem o resultado da convolução. O *stride* determina o passo que a máscara dará ao percorrer a imagem. Na Figura 7, por exemplo, uma máscara de tamanho 2x2 e *stride* 2 foi utilizada. Cada quadrado de cor diferente foi utilizado para gerar o pixel correspondente na imagem de saída.

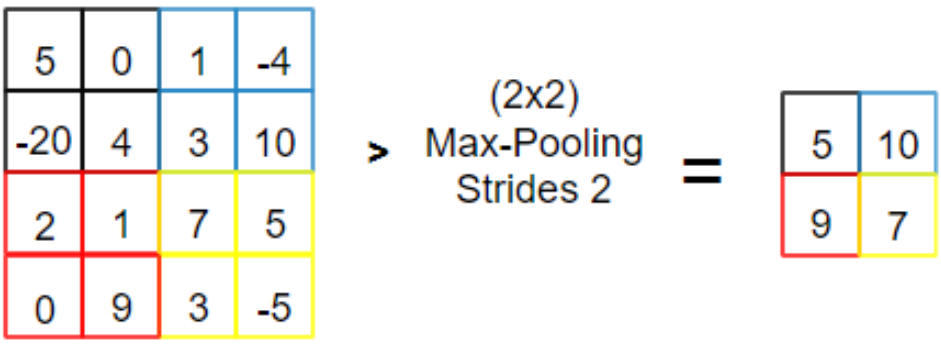
Além desses parâmetros, outros podem ser utilizados, como o *padding*. O *padding* define se a máscara sempre começará e terminará nos pixels extremos da imagem de entrada (primeiras e últimas linhas e colunas) ou se a imagem original será estendida para o cálculo da convolução. No último caso, quando utilizado com *stride* 1, resultará em uma imagem de saída com as mesmas dimensões da imagem de entrada.

2.2.1.2 CAMADA DE POOLING

As camadas de *pooling* têm como objetivo reduzir a dimensão de uma imagem enquanto mantêm e enfatizam as características encontradas na camada de convolução. Além disso, o *pooling* ajuda a tornar a representação aproximadamente invariável a pequenas translações da entrada. Essa invariância significa que, se a entrada for deslocada minimamente, os valores da maioria dos pixels de saída não mudam. Essa propriedade pode ser muito útil se a presença de uma característica for mais relevante do que sua localização exata (24).

Uma das camadas mais utilizadas com esse objetivo é a camada de *Max-Pooling* (25). Como o nome sugere, esta operação substitui um quadrado na imagem de entrada pelo pixel máximo dentro dessa região. Assim como na convolução, os parâmetros tamanho de janela e *stride* controlam como essa operação será realizada. Naturalmente, quanto maiores esses parâmetros, maior será a redução das dimensões da imagem. A Figura 8 ilustra um exemplo dessa camada sendo aplicada com tamanho 2x2 e *stride* 2.

Figura 8 – Exemplo de aplicação de camada de *Max-Pooling*.



Fonte: Elaborada pelo autor (2024).

2.2.1.3 CAMADA FULLY-CONNECTED

Considerando a configuração da Figura 6, a chamada *Fully-Connected Layer* é a última camada antes da saída. Sua principal função é utilizar todas as características extraídas pelas camadas anteriores para realizar a classificação da imagem de entrada.

O nome dessa camada se deve ao fato de que cada neurônio da camada anterior, neste caso a última camada de *Max-Pooling*, está conectado a todos os neurônios da próxima camada, que são utilizados para classificar a imagem. Para isso, normalmente é aplicada uma função de ativação, seguindo uma abordagem similar à utilizada pelo *Multilayer Perceptron* (24, 26). Os pesos desta camada são ajustados durante o processo de treinamento para torná-los eficazes na tarefa de classificação desejada.

2.2.1.4 FUNÇÃO DE ATIVAÇÃO

As funções de ativação desempenham um papel crucial nas CNNs, determinando se um neurônio será ativado com base em uma lógica específica e introduzindo a não-linearidade essencial para aprender padrões complexos e realizar a classificação de imagens. Uma das funções mais utilizadas é a *Rectified Linear Unit* (ReLU) (27, 28), que mantém os valores de entrada positivos e zera os negativos. A ReLU é comumente empregada em conjunto com as camadas de convolução, aproveitando-se de sua simplicidade computacional e eficácia na aprendizagem de representações discriminativas.

Outro exemplo importante é a função sigmoide (29), que mapeia os valores de entrada para um intervalo entre 0 e 1, adequando-se bem à camada de saída, especialmente em tarefas de classificação binária. As Equações 2.24 e 2.25 ilustram matematicamente o funcionamento da ReLU e da sigmoide, respectivamente, enquanto a Figura 9 oferece uma representação visual de como essas funções operam.

$$\mathrm{ReLU}(x) = \max(0, x) \tag{2.24}$$

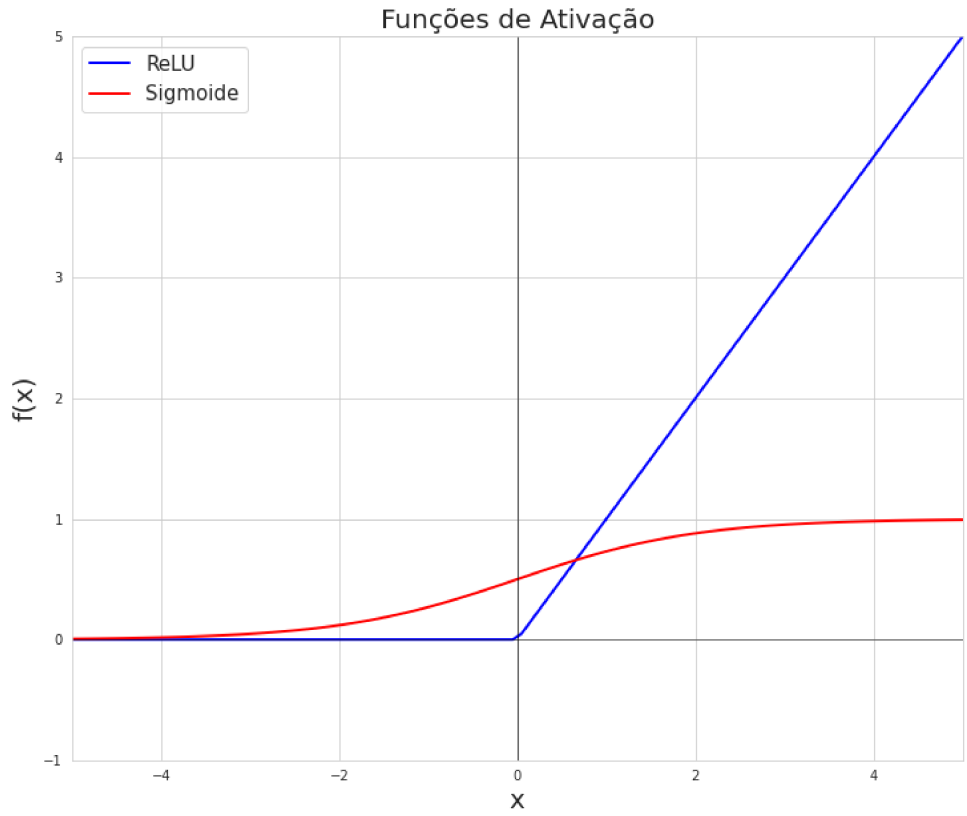
$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.25}$$

2.2.1.5 OUTRAS OPERAÇÕES UTILIZADAS

Além dos componentes apresentados anteriormente, existem outras operações que podem ser utilizadas dentro de CNNs para auxiliar no processo de treinamento, como as apresentadas a seguir.

- **Batch Normalization:** Proposto inicialmente por (30), este método visa acelerar o processo de convergência de uma CNN e melhorar a estabilidade do treinamento. A técnica é aplicada individualmente em uma camada, normalizando os dados de

Figura 9 – Exemplo de funções de ativação.



Fonte: Elaborada pelo autor (2024).

entrada para ter média 0 e desvio padrão 1. Isso é feito subtraindo a média e dividindo pelo desvio padrão dos dados em cada *batch* durante o treinamento. Após a normalização, são aplicados um coeficiente de escala e um deslocamento, permitindo à rede ajustar dinamicamente a normalização durante o aprendizado. Um aspecto importante dessa operação é o tamanho dos *batches*, que são subgrupos dos dados enviados para a CNN durante o treinamento.

- **Dropout:** Esta técnica tem como objetivo mitigar o *overfitting* durante o treinamento de uma CNN (31). Durante o processo de aprendizado, o *dropout* desativa aleatoriamente alguns neurônios com uma probabilidade p. Isso impede que a rede dependa excessivamente de neurônios específicos ou de combinações particulares de neurônios, promovendo uma melhoria na capacidade geral de generalização da rede. O *dropout* é especialmente eficaz em redes profundas, onde a tendência ao *overfitting* é maior (24).

- **Regularização L1 e L2:** As técnicas de regularização L1 e L2 são amplamente utilizadas para evitar *overfitting* em CNNs. A regularização L1, também conhecida como regularização de Lasso, adiciona a soma dos valores absolutos dos pesos ($L1$) à função de perda, incentivando a esparsidade dos pesos da rede. Por outro lado,

a regularização L2, conhecida como regularização de Ridge, adiciona a soma dos quadrados dos pesos ($L2$) à função de perda, incentivando a distribuição uniforme dos pesos. Essas técnicas penalizam pesos elevados, promovendo uma configuração de parâmetros mais simples e, assim, melhorando a generalização do modelo (32, 33).

Para finalizar, é essencial destacar que um conhecimento adequado sobre cada componente apresentado, aliado ao entendimento profundo dos dados, facilita significativamente a busca por uma configuração ótima de uma CNN. Ao longo deste trabalho, cada um desses componentes foi rigorosamente testado e ajustado para encontrar a configuração ideal, que será detalhadamente descrita nos resultados. Este processo meticuloso garante que a rede neural esteja bem equipada para lidar com os desafios específicos dos dados em questão e alcançar um desempenho superior.

3 CYGNO

Este capítulo será dedicado a apresentar uma visão geral sobre detecção de matéria escura e sobre o experimento CYGNO, mostrando seu objeto de estudo, onde e como este estudo é realizado.

3.1 DETECÇÃO DIRETA DE MATÉRIA ESCURA

Ao longo do último século, foi sendo gradualmente desenvolvido o que agora é aceito como o modelo padrão da física de partículas. Esse modelo procura explicar as partículas elementares e suas interações, visando compreender todos os fenômenos observáveis no universo. O modelo padrão tem se mostrado excepcionalmente bem-sucedido, descrevendo com precisão quase todos os dados empíricos disponíveis, embora algumas exceções e lacunas existam (34).

Uma das exceções significativas reside no fato de que existem fortes e convincentes evidências de que a maior parte da massa presente no universo observável não é composta por partículas conhecidas e presentes neste modelo padrão. De fato, uma série de indícios relacionados ao potencial gravitacional revela que a dinâmica das galáxias e aglomerados não pode ser explicada unicamente pelo potencial Newtoniano gerado pela matéria visível (35). Essas evidências apontam para a existência de uma substância "invisível"denominada matéria escura (DM), que se estima ser cerca de cinco vezes mais abundante do que a chamada matéria bariônica, e que desempenhou um papel fundamental na formação e evolução do universo tal como pode ser observado hoje (36).

Com o propósito de confirmar a existência da matéria escura e explicar sua natureza, têm sido desenvolvidos equipamentos capazes de detectar partículas com energias extremamente baixas. Esses aparatos têm como objetivo recriar ambientes propícios para a observação e estudo das interações da matéria escura. Além disso, foram elaborados diversos modelos teóricos para explicar a matéria escura (35, 37, 38, 39). Dentre eles, o que emprega as chamadas *Weakly Interacting Massive Particles* (WIMP) é amplamente aceito pela comunidade científica atualmente.

A possibilidade de detecção direta de matéria escura através da observação das interações das WIMPs foi inicialmente discutida em 1985 (40). Como essas partículas não possuem carga elétrica, sua interação com elétrons atômicos é pouco provável na maioria dos casos. No entanto, a chamada dispersão elástica pode ocorrer com os núcleos atômicos, resultando em um recuo atômico detectável (39). A Equação 3.1 descreve a perda total de energia durante esse recuo.

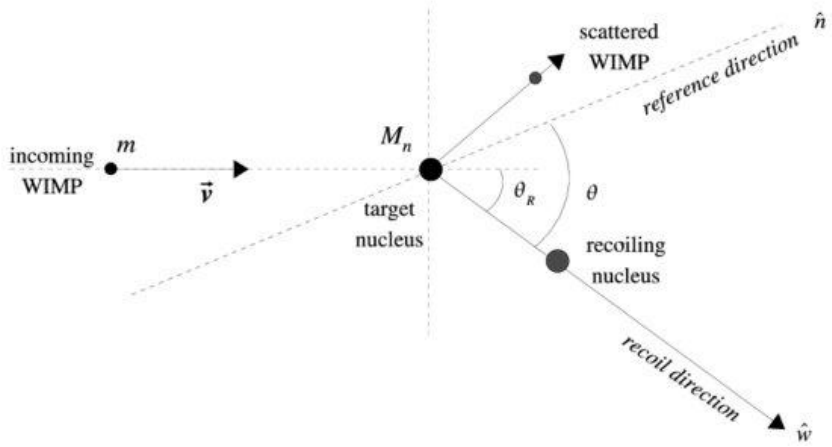
$$\left(\frac{dE}{dx}\right)_{tot} = \left(\frac{dE}{dx}\right)_{elet} + \left(\frac{dE}{dx}\right)_{nucl} \tag{3.1}$$

A interação entre WIMPs e núcleos atômicos resulta principalmente na dissipação de energia na forma de calor, levando ao movimento atômico. Uma parcela menor da energia é perdida por processos eletrônicos, podendo ocasionar ionização ou excitação de átomos. Isso pode levar à produção de cintilação luminosa, que pode ser detectada por fotossensores sensíveis. Entretanto, como uma pequena porção de sinal é gerada, a quantidade de fótons disponíveis para detecção é limitada, o que torna o desafio de identificar essas partículas ainda mais complexo para os detectores.

3.1.1 DETECTORES PARA WIMPS

Os experimentos atuais de detecção direta de matéria escura têm como objetivo obter informações por meio de colisões entre partículas do modelo padrão e a matéria escura. Essas colisões resultam na produção de duas partículas, uma do modelo padrão e outra da matéria escura, conforme ilustrado na Figura 10. A energia liberada durante a ionização da partícula conhecida é medida por meio da detecção dos fótons emitidos, seja quando o elétron preenche a lacuna na eletrosfera atômica ou quando interage com outros materiais após ser acelerado por um campo elétrico externo, como é o caso do experimento Xenon (41).

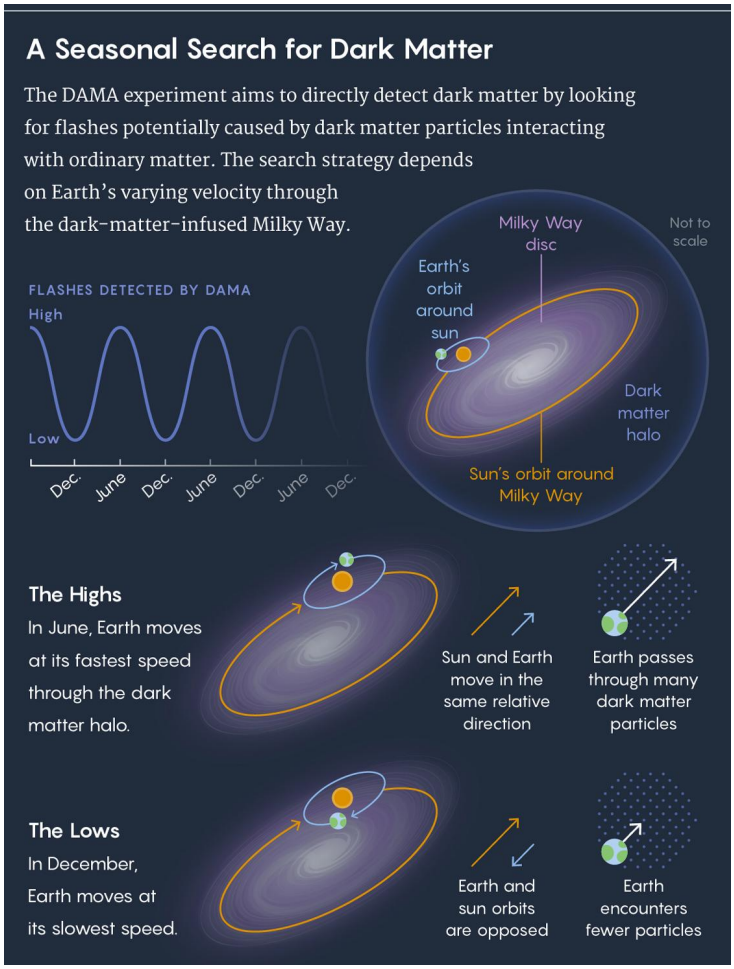
Figura 10 – Colisão entre uma partícula do modelo padrão e WIMP.



Fonte: Extraída de (42)

Alguns experimentos, como DAMA (43) e GoGeNT (44), lidam com a medição de um sinal que varia ao longo do ano, conhecido como modulação anual. Essa variação ocorre devido aos movimentos de translação da Terra e o do Sol ao redor da Via Láctea, que mudam a velocidade relativa com a qual a Terra atravessa o halo de matéria escura, conforme ilustrado na Figura 11. Em razão dessas variações, alguns desses experimentos fazem suposições sobre características específicas da DM para filtrar o sinal detectado.

Figura 11 – Diagrama simplificado da busca pela matéria escura durante os movimentos de translação da Terra e do Sol

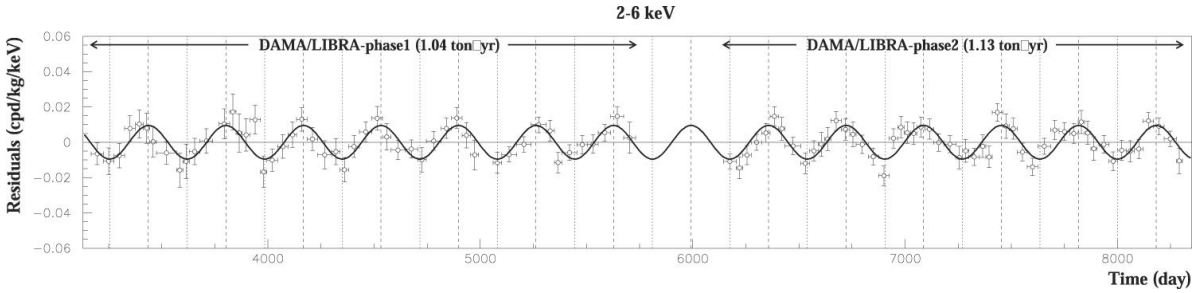


Fonte: Ilustração feita por Lucy Reading-Ikkanda para a Quanta Magazine

No caso específico do experimento DAMA, foi observada uma variação anual no número de eventos detectados, com uma taxa de eventos mais elevada ocorrendo uma vez por ano. Essa ocorrência está associada ao alinhamento das velocidades tangenciais da Terra e do Sol, resultando em um fluxo maior de DM atravessando a Terra (45). A Figura 12 ilustra esse comportamento, bem como a situação em que as velocidades tangenciais possuem sentidos diferentes.

No entanto, devido à ausência de detecção de um sinal semelhante em outros experimentos de detecção direta de matéria escura, surgem objeções e questionamentos em relação à descoberta relatada pelos pesquisadores do experimento DAMA. Como resultado, uma série de outros experimentos está em andamento em diversas partes do mundo, com o objetivo de investigar e confirmar esses resultados por meio da utilização de abordagens e tecnologias distintas. Dentre elas, uma das tecnologias mais promissoras é a utilização de detectores direcionais. Esses detectores são projetados para identificar a direcionalidade das WIMPs, permitindo discriminar os sinais de WIMPs de outros

Figura 12 – Modulação anual exibida pelas variações na taxa de eventos das duas fases DAMA/LIBRA



Fonte: Extraída de (45).

eventos. No experimento CYGNO, por exemplo, o detector foi construído utilizando as chamadas *Time Projection Chambers* (TPCs) e *Micro-pattern gas detectors* (MPGD), que serão abordadas nas próximas seções, juntamente com um sensor de imagem de alta granularidade para leitura dos sinais ópticos.

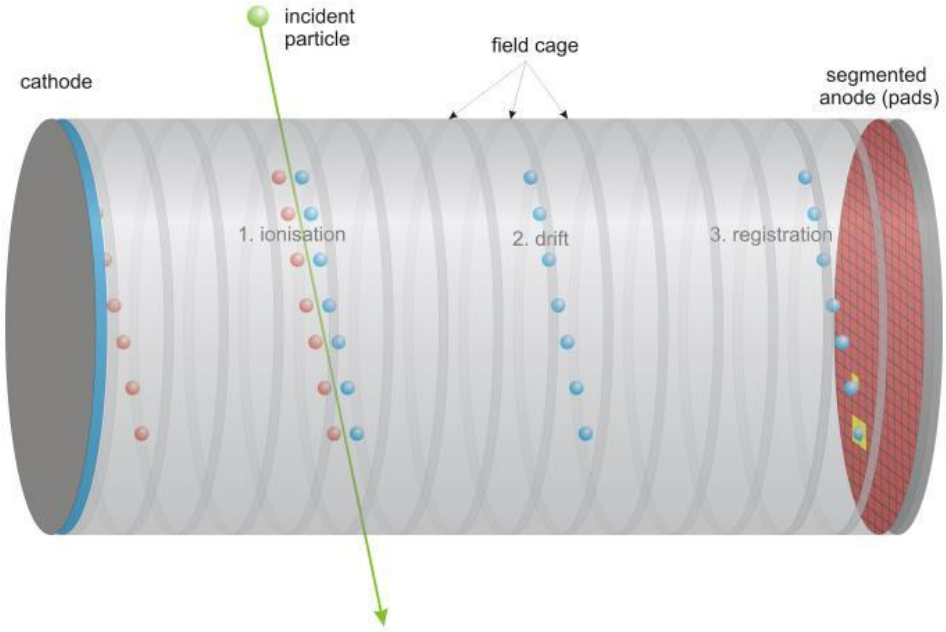
3.1.2 Time Projection Chambers (TPCs)

A *Time Projection Chamber* (TPC), introduzida por D.R. Nygren em 1976 (46), é um tipo de detector de partículas que utiliza uma combinação de campos elétricos e magnéticos em uma gaiola contendo gás ou líquido, que é a região sensível do mesmo, para reconstruir tridimensionalmente a trajetória ou interação de uma partícula. Seu funcionamento e composição podem ser observados na Figura 13. Quando uma partícula atravessa o meio e possui energia suficiente para ionizá-lo, ela deixa um rastro de elétrons e íons, que são direcionados para o ânodo e o cátodo, respectivamente, por ação do campo elétrico presente. A carga medida no terminal anódico é considerada proporcional à energia das partículas (47), permitindo a sua estimativa. Atualmente, a etapa de amplificação e leitura em uma TPC é baseada em *Micro-pattern Gas Detectors* (MPGD), substituindo as clássicas *Multi-Wire Proportional Chambers* (MWPC) (48).

A reconstrução tridimensional na TPC ocorre da seguinte maneira: as coordenadas x e y são determinadas diretamente pela leitura da segmentação do plano, alcançando uma resolução da ordem de poucos µm; a coordenada z precisa ser estimada se a velocidade (v_d) do elétron no volume de gás for conhecida, conforme a Equação 3.2, onde t_1 representa o tempo de chegada do elétron ao ânodo e t_0 é o tempo no qual ocorreu a interação. Caso esse último tempo não seja conhecido, ainda é possível reconstruir essa coordenada se a resolução temporal for suficientemente precisa para discriminar os tempos de chegada dos elétrons produzidos pelo mesmo rastro.

$$z = v_d(t_1 - t_0) \tag{3.2}$$

Figura 13 – Princípio de funcionamento da TPC



Fonte: Extraída de (49).

Outra possibilidade na TPC é a aplicação de um campo magnético (B) paralelo ao campo elétrico (E), com o objetivo de curvar a trajetória das partículas carregadas, possibilitando a medição do *momentum* das mesmas. Além disso, esse campo magnético (B) reduz a difusão dos elétrons ao longo do percurso até o ânodo, resultando em uma melhor resolução das coordenadas x e y. Todavia, para garantir condições de operação otimizadas, como velocidade e ganhos constantes, é essencial que haja uma boa homogeneidade entre esses campos, tornando necessário o monitoramento contínuo deles.

3.1.3 Micro-pattern gas detectors (MPGD)

Na física de partículas, os MPGDs são uma classe de detectores de ionização gasosa que se destacam por sua alta granularidade. Esses detectores são compostos por estruturas microeletrônicas, com distâncias extremamente reduzidas (inferiores a 1 mm) entre o cátodo e o ânodo. Atualmente, o experimento CYGNO utiliza essa tecnologia para amplificar o sinal gerado em uma TPC.

A primeira estrutura desse tipo a ganhar popularidade foi o chamado *Micro-Strip Gas Chamber* (MSGC) (50), que apresentava um espaçamento estreito entre o ânodo e as tiras, aumentando a taxa de capacidade dos detectores em duas ordens de magnitude. Entretanto, a estabilidade do detector ainda era um desafio, uma vez que descargas locais podiam modificar o campo elétrico. Essas descargas eram principalmente induzidas por partículas ionizantes intensas ou altas taxas de partículas, o que poderia danificar as tiras anódicas. Para superar esse problema, foi introduzida a tecnologia *Gas Electron Multiplier*

(GEM) (51) como uma etapa de pré-amplificação para esses detectores.

Essa tecnologia é composta por uma fina folha isolante revestida de cobre com uma matriz retangular de orifícios densamente espaçados. Ao aplicar uma tensão entre os eletrodos, forma-se um campo de dipolo concentrado dentro dos orifícios, o qual é suficientemente forte para permitir a amplificação do sinal. A GEM, sendo uma estrutura de amplificação, pode ser empilhada em cascata com outras GEMs, aumentando ainda mais a amplificação e reduzindo a probabilidade de descargas. Em particular, a configuração conhecida como *Triple-GEM*, que utiliza três GEMs, tem se tornado um padrão amplamente adotado em várias aplicações (52, 53), incluindo o experimento CYGNO.

3.2 EXPERIMENTO CYGNO

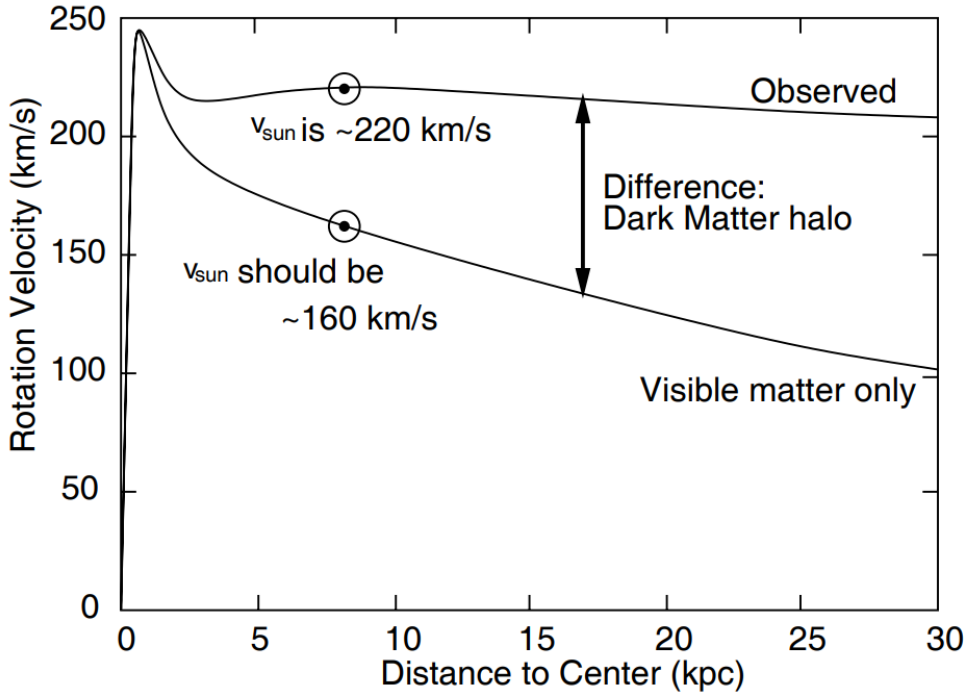
Um dos maiores desafios para os experimentos que visam desvendar os mistérios da matéria escura é lidar com a radiação de fundo, tanto interna aos detectores utilizados quanto proveniente do ambiente. Essa radiação pode produzir sinais com energias próximas da região de interesse (abaixo de 30 keV) (54), e assim serem confundidos com sinais gerados por WIMPs. Por essa razão, têm sido desenvolvidas técnicas com o objetivo de discriminar os sinais gerados por recuo nuclear, possivelmente originados por WIMPs, dos sinais gerados por recuo eletrônico, normalmente causados pela radiação de fundo. A fim de minimizar esse efeito, esses experimentos são frequentemente localizados em ambientes subterrâneos e utilizam materiais de altíssima pureza em sua fabricação, alguns dos quais possuem a capacidade de discriminar um recuo nuclear de outras interações que possam ocorrer mesmo em ambientes subterrâneos.

3.3 BUSCAS PELA MATÉRIA ESCURA ATRAVÉS DO EXPERIMENTO CYGNO

A Via Láctea é a única galáxia na qual é possível se examinar em detalhes, pois é a galáxia onde o sistema Solar está localizado. Ela apresenta um movimento de rotação no sentido horário ao redor do seu centro, o qual exibe irregularidades em relação ao que é previsto com base na massa total visível, composta por estrelas, gás e outros componentes. Uma dessas irregularidades é observada na velocidade de rotação dos objetos localizados nas regiões mais externas da galáxia. De acordo com a previsão, essa velocidade deveria ser inversamente proporcional à distância desses objetos ao centro da galáxia e proporcional à massa interna contida em uma esfera de raio igual a essa distância. No entanto, a velocidade observada é maior do que o previsto, como pode ser observado na Figura 14, o que sugere a existência de uma quantidade adicional de massa nas regiões externas da galáxia que não pode ser observada, possivelmente relacionada à matéria escura (55).

O Sol descreve uma órbita em torno do centro galáctico com uma velocidade aproximada de $220\,\mathrm{km\,s^{-1}}$, conforme ilustrado na Figura 14, e sua direção de velocidade aponta para a constelação de *Cygnus*. O sinal esperado no detector do experimento

Figura 14 – Curva da velocidade de rotação na Via Láctea com base na distância para o centro galático.



Fonte: Extraída de (55)

proveniente da interação com WIMPs surge do movimento relativo da Terra em relação ao halo galáctico, quando ela entra em contato com o chamado *DM wind* que aparenta vir da constelação de *Cygnus*. Determinar a direção das partículas de matéria escura que chegam do espaço pode estabelecer uma correlação com uma fonte astrofísica que não se assemelha a qualquer ruído de fundo, fornecendo assim as informações necessárias para identificar o sinal de DM. Além disso, a medição da direcionalidade dessas partículas pode auxiliar na discriminação entre os diferentes modelos utilizados para explicá-la (56) e proporcionar informações adicionais sobre as propriedades das WIMPs. Essas informações seriam inacessíveis sem detectores dessa natureza.

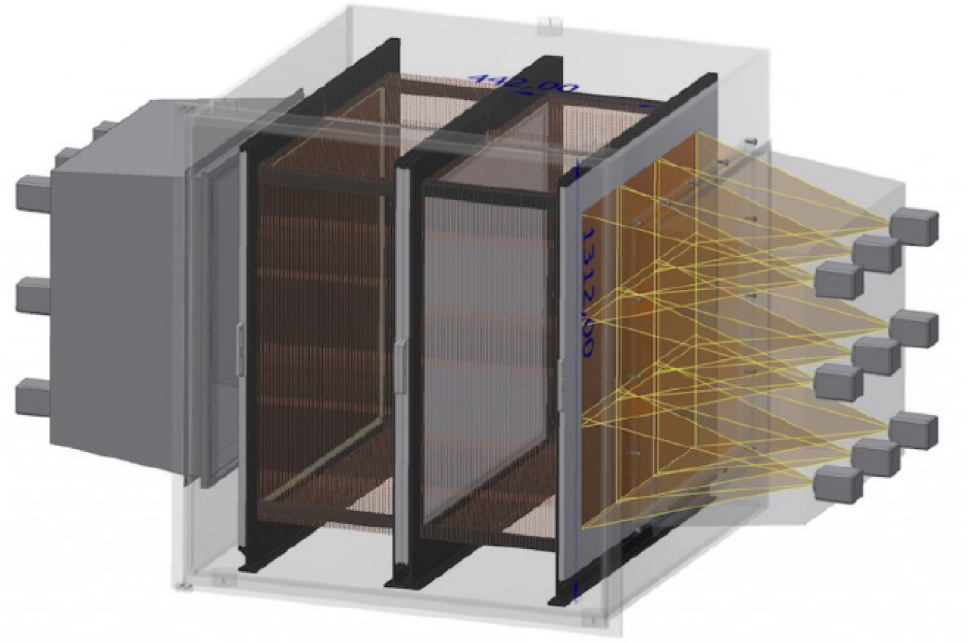
Desse modo, a colaboração CYGNO propõe uma abordagem diferenciada, utilizando um detector com TPC de alta resolução com baixa densidade de núcleos alvo leves para matéria escura (como Hélio e Flúor). O objetivo é aumentar a sensibilidade a WIMPs, mantendo a capacidade de identificar a direção das partículas e rejeitar o ruído de fundo, mesmo em baixas energias. Uma particularidade do uso de Hélio é a possibilidade de trabalhar à pressão atmosférica, o que reduz os custos de produção de equipamentos capazes de suportar diferentes pressões. Além disso, essa escolha garante uma razão adequada entre volume e massa alvo. Essas características permitem que o experimento explore novos aspectos da física de partículas que exigem uma alta capacidade de discriminação entre recuos nucleares e outras partículas, bem como o conhecimento da direção de chegada

dessas partículas.

3.4 DETECTOR DO EXPERIMENTO CYGNO

O experimento CYGNO busca desenvolver um detector da classe MPGD baseado em TPC com um *Triple-GEM* e um sensor de leitura óptica da tecnologia sCMOS que oferece recursos de rastreamento 3D de alta precisão. Ele é sensível à direção dos recuos nucleares e eletrônicos, ótimo para pesquisas relacionadas a WIMPs com baixos valores de massa (1-10 GeV) até o chamado *Neutrino Floor* (57). A Figura 15 mostra uma imagem 3D do detector.

Figura 15 – Esquemático 3D do detector do experimento CYGNO.



Fonte: Extraída de (6)

No entanto, como em vários outros experimentos, o desenvolvimento deste detector foi dividido em algumas etapas, como é descrito a seguir (4):

- Fase-0: É a fase em que atualmente o experimento se encontra e tem como foco a instalação de um grande protótipo (50 L de volume sensível) no subsolo no *INFN-Laboratori Nazionali del Gran Sasso* (LNGS), com o objetivo de estudar seu desempenho em um ambiente de baixo ruído de fundo e validar a simulação por computador.
- Fase-1: Teste da escalabilidade da abordagem experimental em um detector de 1 m^3 enquanto estuda e minimiza o ruído de fundo devido ao material do aparato.

- Fase-2: Deseja-se desenvolver um detector de 30-100 m^3.

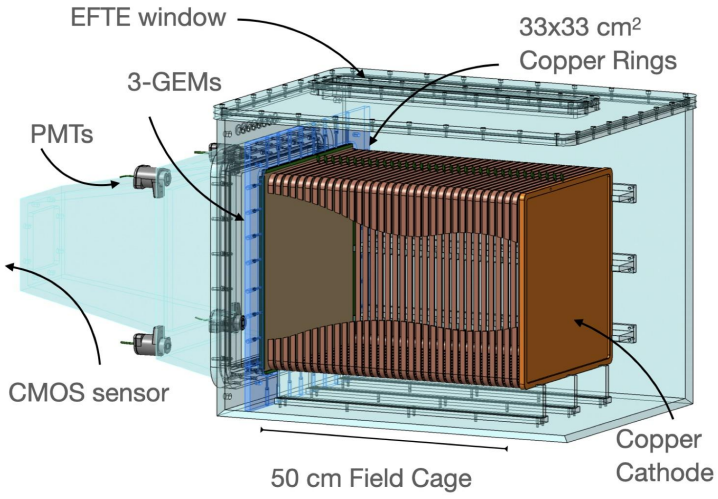
Recentemente, a colaboração CYGNO tem testado diferentes protótipos, como o NITEC (58), ORANGE (59), LEMOn (60, 61) e LIME (7, 62), variando a fonte radioativa (instalação de teste com feixe de elétrons, neutrôns) e algumas condições gerais de operação com o objetivo de entender as nuances do projeto e desenvolver o detector final da última fase do experimento. Este trabalho dará um foco especial ao LIME, que será descrito em mais detalhes na seção **3.4.1**.

A ideia principal por trás dos protótipos é essencialmente a mesma: uma caixa de acrílico preenchida com gás e, pelo menos, um lado transparente para permitir que a câmera tenha contato visual com a área sensível. Um campo elétrico é aplicado para direcionar os elétrons para o lado da caixa onde a câmera está localizada, enquanto um *Triple-GEM* é usado para amplificar o sinal gerado no processo de ionização. A câmera é posicionada do lado de fora da caixa para registrar os sinais de luz gerados durante todo esse processo.

3.4.1 DETECTOR LIME

Um dos protótipos mais recentes do experimento CYGNO é o *Long Imaging ModulE* (LIME) (7, 62), que foi usado para coletar todos os dados reais que compõem os bancos de dados utilizados neste trabalho. Um esboço deste detector é mostrado na Figura 16. Ele possui um volume sensível de 50 litros, com uma gaiola de campo em forma de paralelepípedo composta por 34 anéis de cobre com 10 mm de largura e espaçamento de 14 mm. O volume sensível é delimitado em um lado por um cátodo de cobre e, no outro lado, por uma estrutura *Triple-GEM* de 330×330 mm^2.

Figura 16 – Esboço do detector LIME.



Fonte: Extraída de (62).

O detector geralmente opera com uma mistura de gás He/CF4 em uma proporção de 60/40. Um recipiente de acrílico contém a região sensível e a estrutura *Triple-GEM*, garantindo a impermeabilidade de gás do protótipo. Na parte superior, ele é provido de uma janela de 50 cm de comprimento, fechada com uma fina camada de plástico à base de flúor (EFTE) para ser usada em testes com fontes de raios-X de baixa energia. As camadas *Triple-GEM* são separadas por 2 mm. Além disso, a espessura de 1 mm do acrílico localizado na frente da camada distante desta estrutura proporciona uma transmissão muito eficiente da luz produzida nos seus canais de multiplicação. Essa luz é capturada por:

- 4 fotomultiplicadores Hamamatsu R7378, com 22 mm de diâmetro (63);
- Uma câmera Orca Fusion da classe sCMOS (64) com 2304 ×2304 pixels, com uma área ativa de 6,5×6,5μm^2.

Além disso, o detector utiliza dois campos elétricos em seu funcionamento, sendo um no volume sensível (E_{Drift}) e outro entre as camadas das GEMs (E_{Transf}). Essa estrutura também necessita de uma fonte de alimentação para aplicar uma diferença de potencial V_{GEM} entre seus terminais para que esta possa realizar a operação de amplificação. A configuração típica de operação do detector está descrita a seguir (62):

- Fluxo de gás igual a 12 L/h;
- E_{Drift} = 0.9 kV/cm;
- E_{Transf} = 2.5 kV/cm;
- V_{GEM} = 440 V.

3.5 BANCOS DE DADOS

Os bancos de dados utilizados neste trabalho podem ser divididos em duas categorias principais: (1) imagens simuladas geradas pela colaboração, com o objetivo de reproduzir um ambiente controlado, e (2) imagens reais adquiridas com o protótipo LIME. Cada uma dessas categorias será detalhada nas seções seguintes.

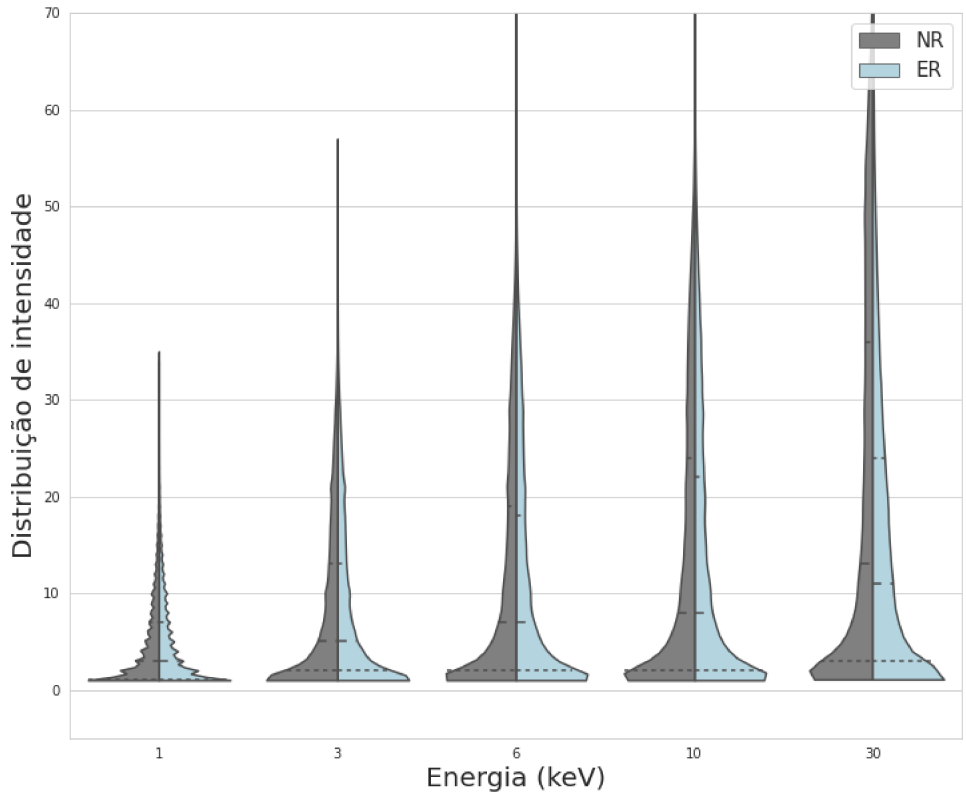
3.5.1 SIMULAÇÃO

As imagens de simulação foram geradas com base no protótipo LIME utilizando o software GEANT4 (65, 66), amplamente reconhecido na literatura, o qual simula as interações de partículas dentro do detector. Como mencionado na Seção 3.2, o experimento tem como objetivo detectar sinais com energias abaixo de 30 keV. Dessa forma, foram simuladas as energias de 1, 3, 6, 10 e 30 keV, cada uma com dois tipos principais de

eventos: recuos nucleares (NR) e eletrônicos (ER). Cada combinação de energia e tipo de evento resultou em 1000 imagens, cada uma com 2304x2304 pixels contendo um único sinal simulado, representando o processo de detecção pelo fotosensor, incluindo a digitalização.

A Figura 17 apresenta a função de distribuição de probabilidade das intensidades de cada pixel nas simulações de NR e ER para as energias mencionadas. Utilizou-se o gráfico violino para ilustrar essas distribuições, mostrando a mediana, o primeiro e o terceiro quartil dos dados (67). Observa-se que os dois tipos de sinais exibem comportamentos similares em baixas energias, enquanto a distribuição de NR apresenta uma cauda mais pronunciada em altas energias. Por outro lado, a distribuição de ER mostra mais pixels com intensidades menores. Isso sugere que os sinais tendem a se diferenciar mais conforme a energia aumenta.

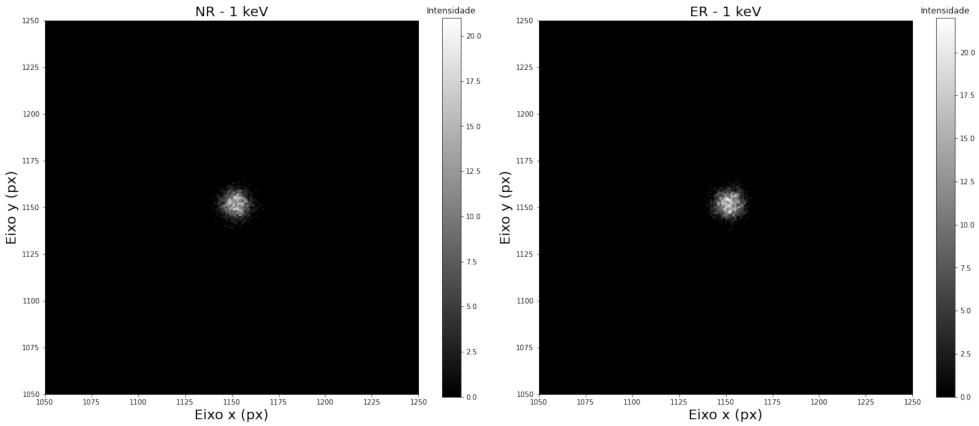
Figura 17 – Gráfico violino das intensidades dos pixels de simulação em cada energia para NR (cinza) e ER (azul).



Fonte: Elaborada pelo autor (2024).

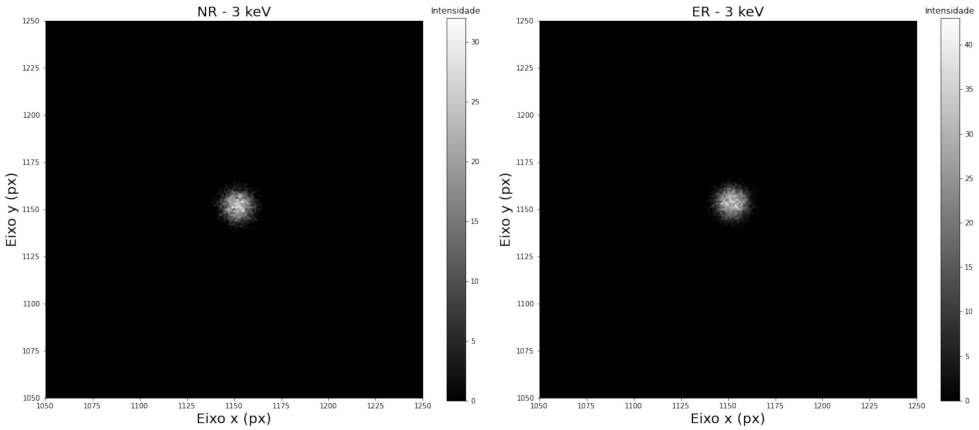
Essas semelhanças e diferenças podem ser verificadas nas Figuras 18, 19, 20, 21, 22, que mostram exemplos de sinal das energias citadas para NR (esquerda) e ER (direita). Os sinais possuem comportamento circular e acumulam boa parte de sua energia nos pixels centrais nas baixas energias, o que se repete apenas para o NR nas altas energias. Os sinais de ER começam a apresentar um rastro curvo com início e fim bem definidos de acordo com a intensidade dos pixels, a partir de 30 keV.

Figura 18 – Sinais simulados de NR (esquerda) e ER (direita) de 1 keV.



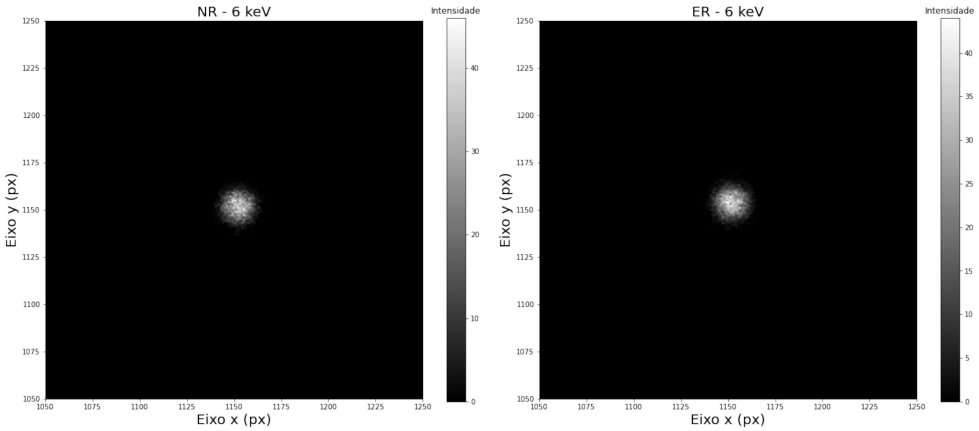
Fonte: Elaborada pelo autor (2024).

Figura 19 – Sinais simulados de NR (esquerda) e ER (direita) de 3 keV.



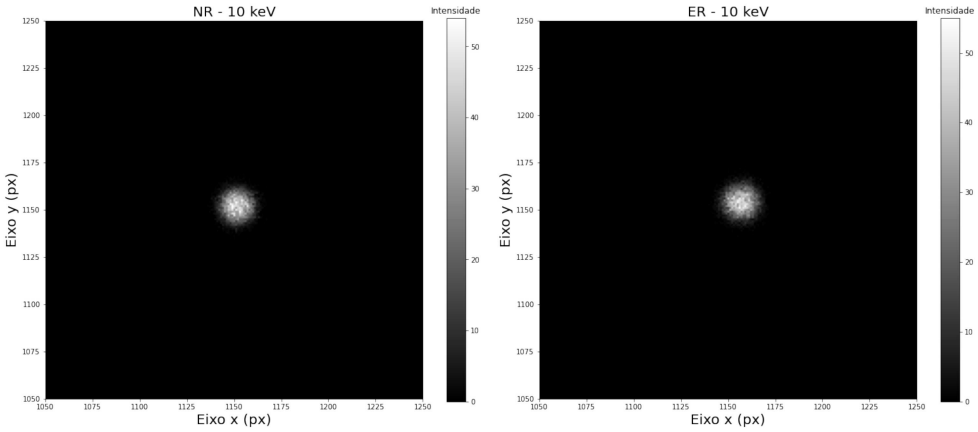
Fonte: Elaborada pelo autor (2024).

Figura 20 – Sinais simulados de NR (esquerda) e ER (direita) de 6 keV.



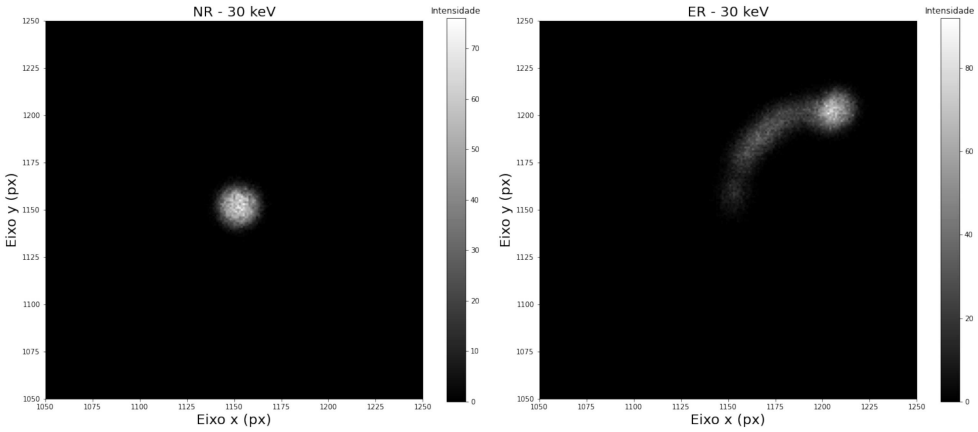
Fonte: Elaborada pelo autor (2024).

Figura 21 – Sinais simulados de NR (esquerda) e ER (direita) de 10 keV.



Fonte: Elaborada pelo autor (2024).

Figura 22 – Sinais simulados de NR (esquerda) e ER (direita) de 30 keV.



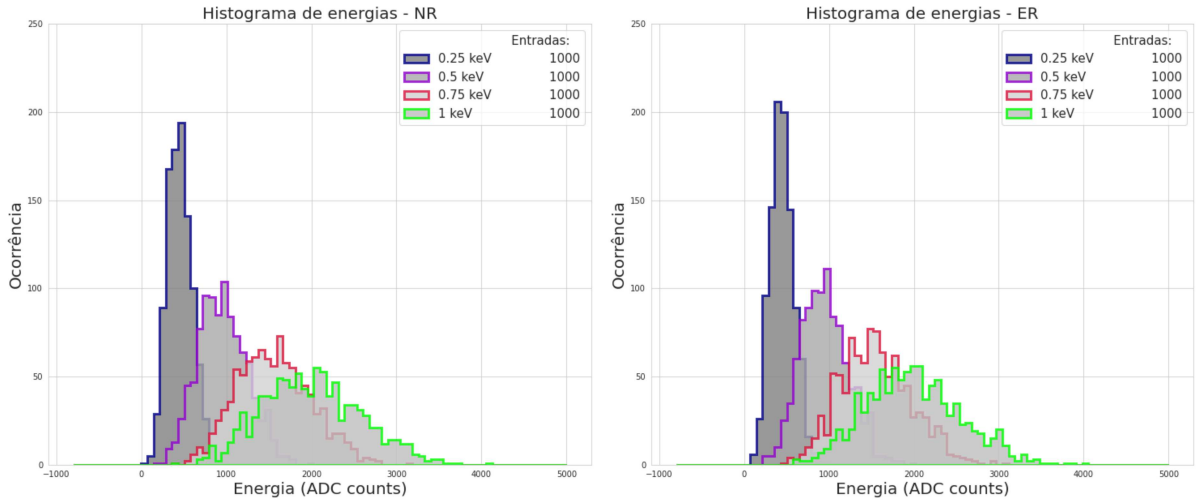
Fonte: Elaborada pelo autor (2024).

Vale ressaltar que, para a análise proposta nesse trabalho, espera-se que apenas sinais de baixíssimas energias apresentem alguma dificuldade para serem detectados. Com isso em mente, foram criadas outras três simulações, com 0.25, 0.5 e 0.75 keV, a partir da de 1 keV, para serem utilizadas nos algoritmos que serão descritos nos próximos capítulos. Cada pixel teve sua intensidade alterada de acordo com a energia desejada, também sendo aplicado o processo de digitalização. A Figura 23 mostra a soma das intensidades dos pixels de cada energia para NR (esquerda) e ER (direita), que basicamente é a forma de se calcular a energia de um sinal em uma imagem digital.

3.5.2 DADOS REAIS

As simulações são de suma importância para treinar modelos como os propostos neste trabalho, pois representam, em um ambiente controlado, as condições reais do

Figura 23 – Histograma de energias dos eventos simulados de NR (esquerda) e ER (direita).



Fonte: Elaborada pelo autor (2024).

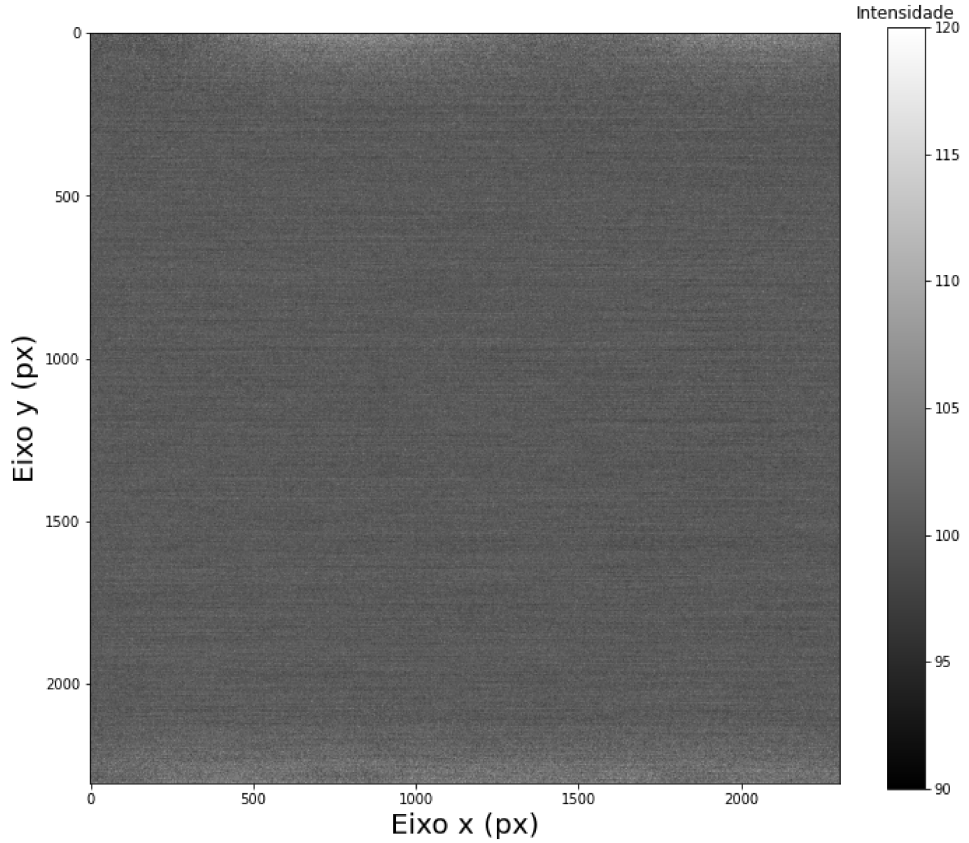
detector. No entanto, como em qualquer experimento, as imagens reais estão sujeitas a componentes de ruído intrínsecos à eletrônica do detector e ao sensor óptico. Por esta razão, imagens que contêm apenas ruído eletrônico devem ser utilizadas tanto como exemplos de imagens a serem descartadas por um algoritmo de *trigger*, quanto para serem adicionadas às simulações (que possuem intensidades diferentes de 0 apenas em pixels de sinal), emulando o que seria uma imagem real que possui algum sinal.

A Figura 24 mostra um exemplo de uma imagem de ruído eletrônico adquirida pelo detector, utilizando uma configuração com tensões V_{GEM} tão baixas que nenhum tipo de interação entre partícula e gás pode gerar um sinal na imagem. Pode-se observar que as bordas superior e inferior são mais ruidosas que o restante da imagem, o que justifica o corte que o algoritmo de análise das imagens do experimento aplica nessas regiões, que também será repetido para o algoritmo de *trigger*.

A maior parte das intensidades nas imagens de ruído está em um intervalo entre 90 e 120 ADCs, como mostrado na Figura 25, que exibe a distribuição das intensidades de 10 imagens de ruído eletrônico. No entanto, ainda existem pixels com intensidades de quase 200 ADCs, devido ao chamado ruído impulsivo, que ocorre em posições aleatórias nas imagens, dificultando a tarefa para algoritmos mais simples de identificar a presença de um sinal de baixo contraste com o fundo.

Além de imagens de ruído eletrônico, imagens reais contendo algum tipo de sinal podem ser adquiridas utilizando o detector com as configurações descritas na Seção **3.4.1**. O experimento normalmente realiza essa aquisição de duas formas: configuração normal, gerando imagens que contêm sinais provenientes de radiação natural, como cósmicos, com

Figura 24 – Exemplo de imagem de ruído eletrônico adquirida do protótipo LIME.



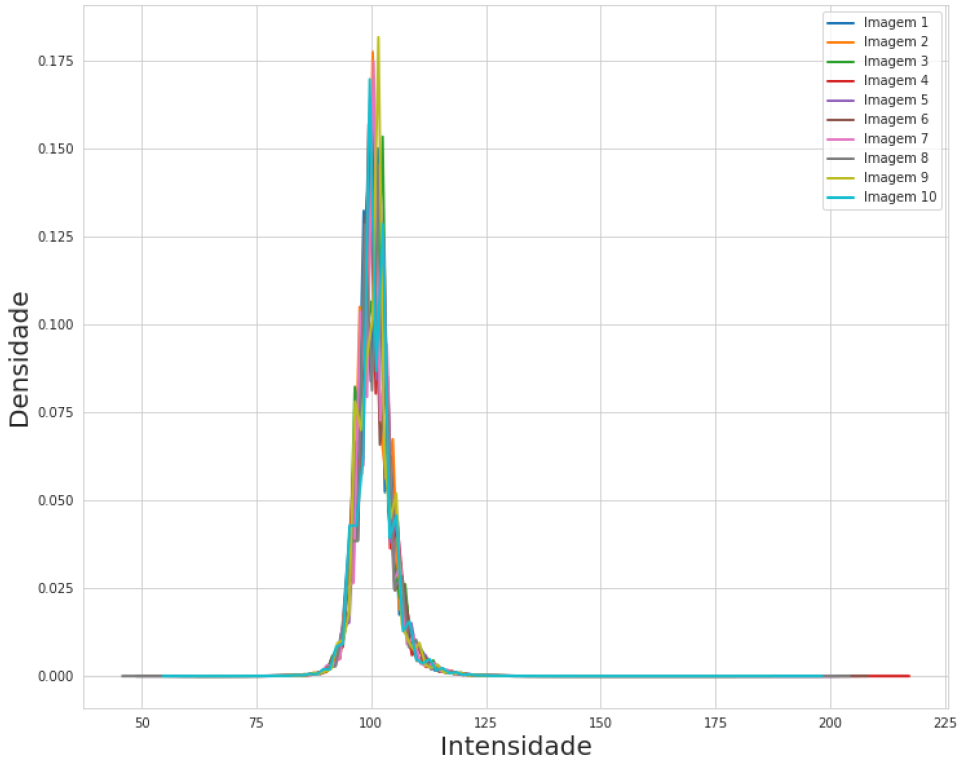
Fonte: Elaborada pelo autor (2024).

energia altamente variada; ou colocando uma fonte radioativa próxima ao detector, com o intuito de induzir sinais conhecidos, além dos citados anteriormente. Uma fonte muito utilizada pelo experimento é a de ^{55}Fe, que induz sinais de ER de 5.9 keV, usados para calibração do detector (62). A Figura 26 mostra exemplos desses tipos de evento.

No entanto, utilizar dados reais para o tipo de análise proposto neste trabalho pode ser desafiador, visto que não existe uma informação concreta se há ou não sinais nas imagens e onde eles estão, como no caso das simulações. Por este motivo, esses dados serão apenas utilizados para confirmar se os algoritmos descritos nas seções seguintes funcionarão bem no ambiente real, sem uma análise precisa de performance. Desta forma, pode-se definir os dois bancos de dados de imagens reais utilizados neste trabalho como:

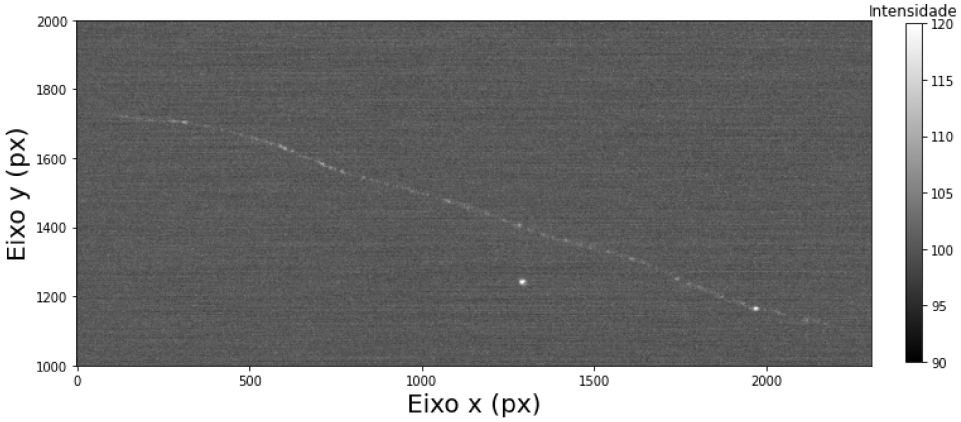
- **Ruído Eletrônico:** Possui 2000 imagens de ruído eletrônico, sendo que metade será utilizada em conjunto com as simulações descritas anteriormente.
- **Sinais Reais:** Possui 300 imagens de dados reais adquiridos com uma fonte de ^{55}Fe colocada ao lado do detector e, portanto, contém tanto sinais de radiação natural quanto ER de 5.9 keV.

Figura 25 – Distribuição de intensidades em imagens de ruído eletrônico.



Fonte: Elaborada pelo autor (2024).

Figura 26 – Exemplos de ER produzido por fonte radioativa de ^{55}Fe (circular) e sinal produzido por cósmico (longo).



Fonte: Elaborada pelo autor (2024).

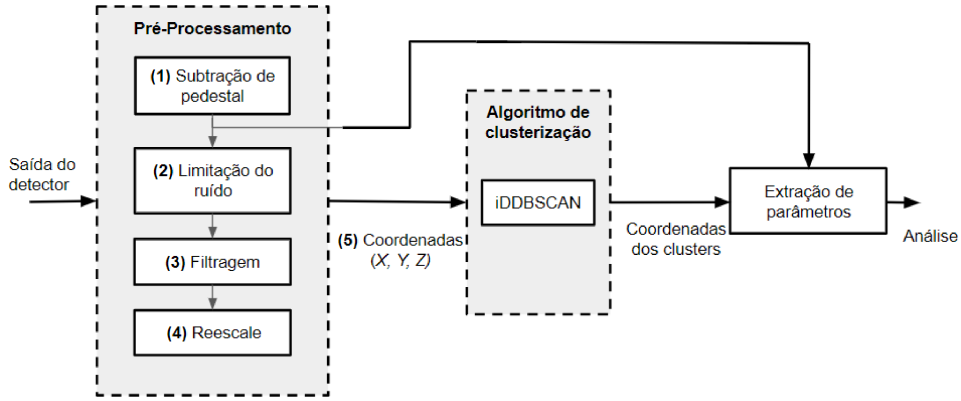
4 METODOLOGIA

Este capítulo abordará a metodologia empregada para determinar os pontos de operação de cada método proposto e comparar os respectivos resultados. Primeiramente, será apresentado o algoritmo de reconstrução do experimento CYGNO, utilizado para análises *offline*, que servirá como referência para a análise final dos resultados. Em seguida, serão discutidos os métodos de filtragem e a CNN, detalhando sua arquitetura e o uso dos dados de simulação neles.

4.1 ALGORITMO DE RECONSTRUÇÃO DO EXPERIMENTO CYGNO

A saída do detector do experimento CYGNO são imagens de 2304x2304 pixels, que podem conter sinais originados pela interação de partículas com o gás presente no detector. Um dos principais objetivos ao analisar essas imagens é identificar quais pixels, em conjunto, representam cada interação. O algoritmo responsável por essa tarefa é conhecido como algoritmo de reconstrução, cujo fluxograma é apresentado na Figura 27 e será explicado detalhadamente a seguir (62).

Figura 27 – Fluxograma do algoritmo de reconstrução do experimento CYGNO.



Fonte: Elaborada pelo autor (2024).

- **Subtração de pedestal:** Como descrito anteriormente, as imagens do experimento são compostas principalmente pela soma do ruído eletrônico proveniente da câmera e do detector, além de uma possível componente resultante da interação de partículas. Portanto, a primeira etapa de pré-processamento consiste em remover a parcela de ruído eletrônico da imagem, conhecida como pedestal. Este pedestal é calculado como a média pixel a pixel de imagens de ruído adquiridas, preferencialmente, pouco antes das imagens a serem analisadas.

- **Limitação de ruído:** Além da remoção da componente média de ruído das imagens, a variabilidade do ruído pixel a pixel também pode ser utilizada. Nesta etapa, o desvio padrão σ_n do ruído é usado para zerar cada pixel cuja intensidade, após a subtração do pedestal, seja menor que o desvio padrão multiplicado por um fator (atualmente 0,6). Além disso, os pixels presentes na região de $y < 304$ e $y > 2050$ são zerados para as análises, por ser uma região muito ruidosa, mesmo após essas duas etapas.

- **Filtragem:** Esta é uma das últimas etapas de pré-processamento e tem como objetivo realizar um ajuste fino na imagem, tentando reduzir possíveis componentes de ruído ainda presentes após as etapas anteriores. Para isso, um filtro de mediana com janela 2x2 é aplicado na imagem antes de ser enviada para a próxima etapa.

- **Reescale:** Mesmo após a filtragem para reduzir a quantidade de pixels que contêm apenas informação de ruído, uma imagem com 2304x2304 pixels ainda é muito grande para qualquer algoritmo de clusterização analisar. Por esta razão, uma etapa de *reescale* é utilizada para reduzir o tamanho da imagem para 576x576 pixels.

- **Clusterização:** Essa etapa utiliza um algoritmo para classificar quais pixels pertencem a um mesmo grupo, denominado cluster, originado da interação de uma única partícula. Para isso, foi desenvolvido um algoritmo chamado *intensity-based directional DBSCAN* (68), que, além de utilizar o conceito do algoritmo original DBSCAN (69) para identificar regiões densas nas imagens e encontrar clusters, inclui uma análise de direcionalidade para auxiliar na detecção de rastros com uma direção bem definida.

- **Extração de parâmetros:** Após a detecção dos clusters, diversos parâmetros são extraídos a partir da imagem em sua resolução original (após a subtração de pedestal). Entre esses parâmetros estão a energia (soma das intensidades dos pixels), posição central, tamanho, e posição de todos os pixels, entre outros. Esses parâmetros podem ser utilizados em análises posteriores para identificar a partícula responsável por gerar cada rastro presente na imagem.

Vale ressaltar que o algoritmo de reconstrução foi otimizado em trabalhos anteriores (62, 68) não apenas para identificar os rastros oriundos de partículas nas imagens, mas também para reconstruí-los de maneira eficaz, permitindo uma análise mais aprofundada. Como o propósito deste trabalho é desenvolver um algoritmo de *trigger*, que se preocupa apenas em determinar se há ou não algum sinal na imagem, a comparação com a reconstrução não é completamente adequada. No entanto, é crucial que os algoritmos de *trigger* sejam capazes de pelo menos identificar as imagens que realmente possuam um sinal e que foram reconstruídos pelo algoritmo de reconstrução.

4.2 ALGORITMOS DE TRIGGER

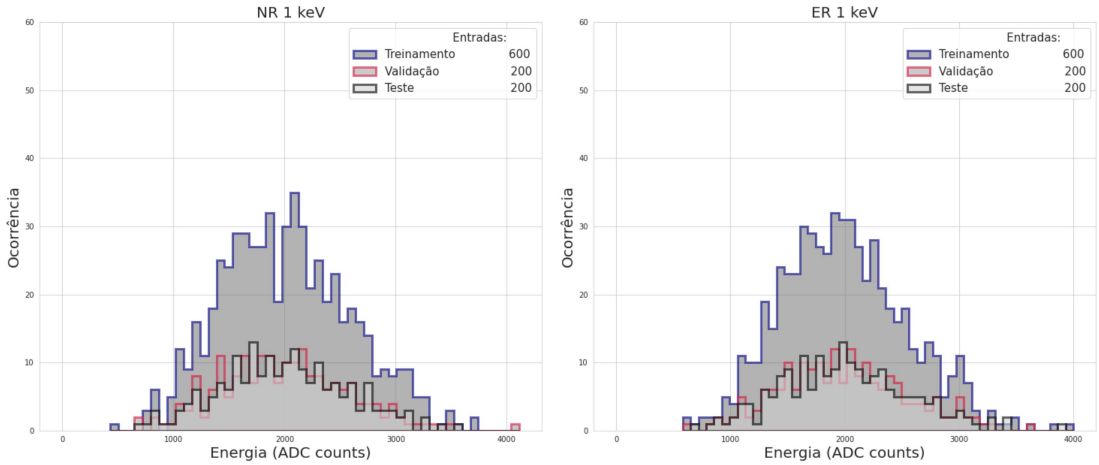
4.2.1 ORGANIZAÇÃO E DIVISÃO DOS DADOS

O desenvolvimento de modelos eficazes para a detecção de sinais, especialmente no contexto de algoritmos de aprendizado de máquina, depende fundamentalmente de uma preparação cuidadosa dos dados. No caso dos métodos chamados supervisionados, como os propostos neste trabalho, os bancos de dados devem ser robustos e suficientemente grandes para representar bem os casos que poderão ser encontrados na prática pelo algoritmo. Esta preparação inclui não apenas a coleta e o pré-processamento dos dados, mas também sua divisão em conjuntos de treino, validação e teste, para garantir que o modelo aprenda de forma eficiente e se generalize bem para novos dados (70, 71).

É muito comum na literatura utilizar uma separação de 60%-20%-20% para treino, validação e teste, sendo estes responsáveis por treinar o modelo, servir como base para dizer qual a melhor configuração durante o treinamento e ser utilizado no resultado final para comparar diferentes métodos, respectivamente (72). Em métodos como a CNN, utilizar o banco de dados de validação é vital para se evitar *overfitting*, enquanto métodos baseados em filtros poderiam ser treinados apenas o conjunto de treino.

Dessa forma, a Figura 28 mostra a divisão feita nos dados de simulação de 1 keV para NR (esquerda) e ER (direita), também considerando a energia em ADC *counts* de cada evento, de forma que cada banco seja balanceado seguindo a proporção desejada tanto em número quanto em energia. Como mencionado na Seção **3.5.1**, os eventos de 1 keV também foram utilizados como base para criar simulações com energias menores, de 0.25, 0.5 e 0.75 keV, de forma a propiciar uma condição onde a detecção de tais sinais seja uma tarefa mais difícil.

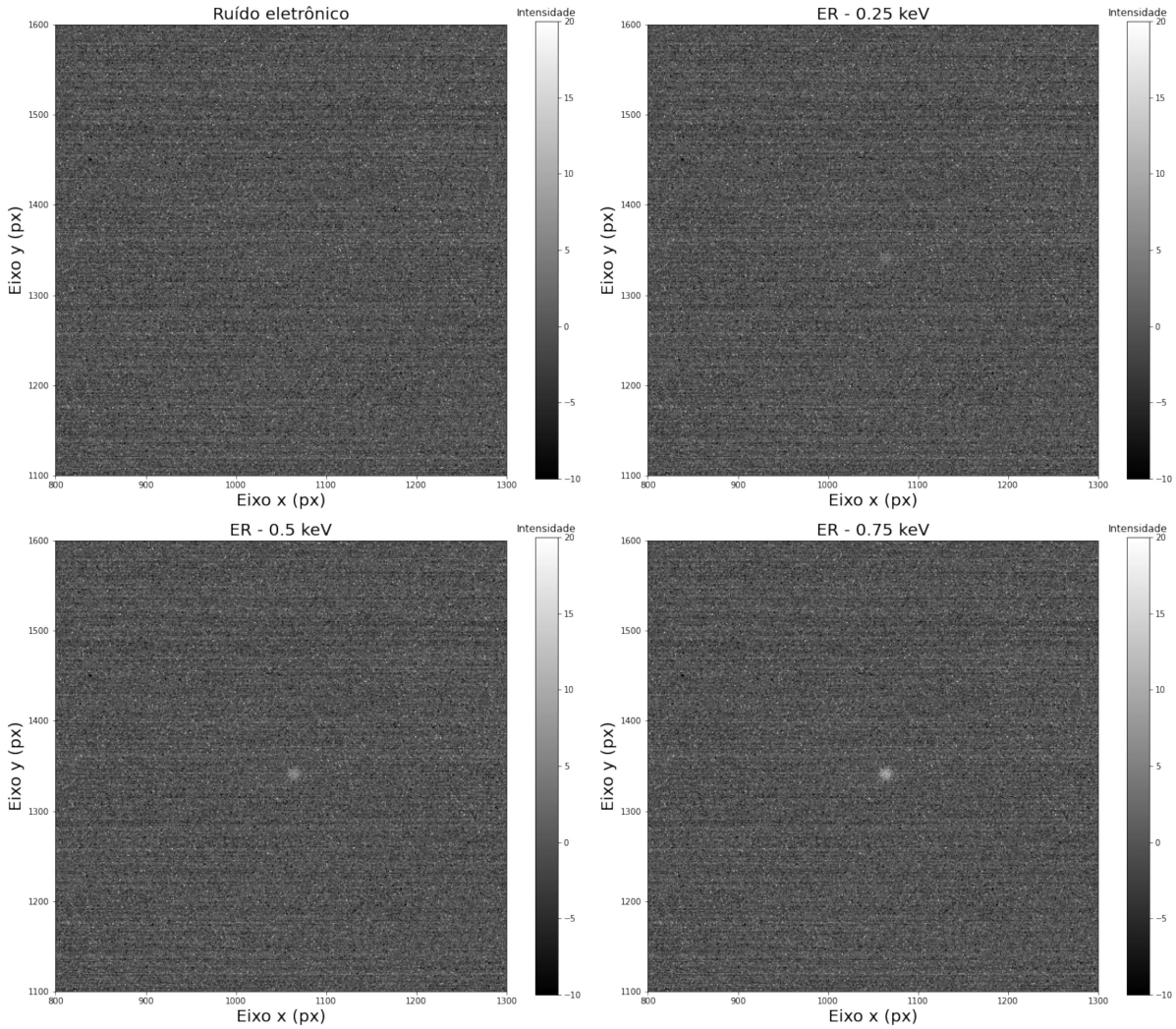
Figura 28 – Histograma de energias dos eventos simulados de NR (esquerda) e ER (direita) após separação em treino, validação e teste.



Fonte: Elaborada pelo autor (2024).

O banco de dados de ruído foi dividido de forma semelhante: metade foi reservada para criar combinações de sinal com ruído, resultando em imagens rotuladas como sinal, enquanto a outra metade foi utilizada como rótulo de ruído, garantindo que não houvesse repetição de imagens de ruído. A Figura 29 ilustra esse processo com um exemplo que contém uma imagem de apenas ruído eletrônico e imagens que também contêm sinais de 0.25, 0.5 e 0.75 keV, todas após a remoção do pedestal, procedimento também usado para os algoritmos de *trigger*. As imagens foram ampliadas, e o sinal foi adicionado ao ruído na mesma posição para demonstrar como o contraste entre sinal e ruído varia com o aumento da energia.

Figura 29 – Exemplo de imagens de ruído eletrônico (canto superior esquerdo) e também com sinais de ER com 0.25 keV (canto superior direito), 0.5 keV (canto inferior esquerdo) e 0.75 keV (canto inferior direito).



Fonte: Elaborada pelo autor (2024).

Como cada método possui suas peculiaridades, os bancos de dados foram ajustados

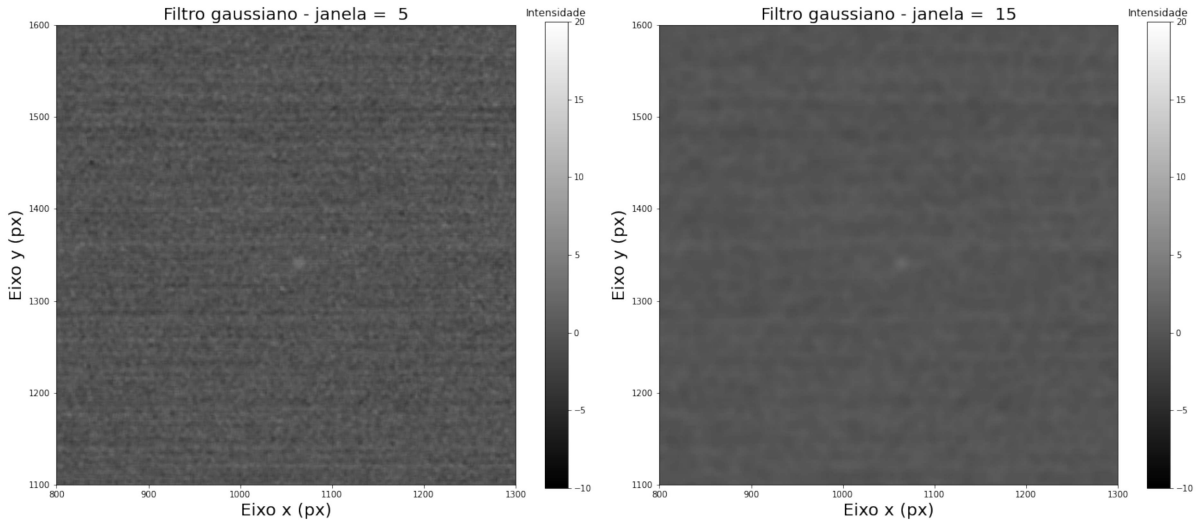
para encontrar o ponto de operação ideal para cada abordagem, enquanto o conjunto de teste foi utilizado para garantir a isonomia dos resultados. Esses ajustes e suas implicações serão detalhados nas próximas seções.

4.2.2 TRIGGER BASEADO EM FILTRAGEM

A detecção de sinais imersos em ruído é uma das principais aplicações dos filtros, especialmente do filtro casado. Conforme discutido na Seção **2.1.4**, este filtro é particularmente eficaz na identificação de sinais, pois maximiza a correlação entre uma máscara baseada no sinal e uma imagem que o contenha, indicando sua posição exata. Portanto, é evidente que, na presença de um sinal, a correlação máxima após a filtragem será mais significativa do que em imagens compostas apenas por ruído.

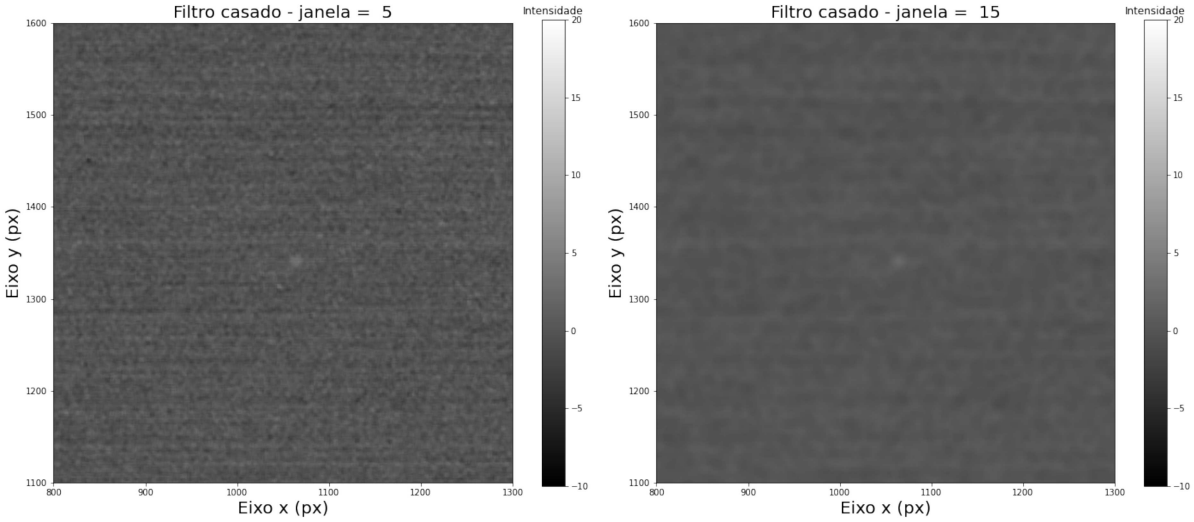
A ideia por trás de um algoritmo simples de *trigger* baseado em filtragem é aplicar o filtro a uma imagem, registrar o valor máximo após a filtragem e compará-lo a um limiar para determinar a presença de um sinal. Esse procedimento é motivado pelo filtro casado, podendo ser utilizado com outros filtros discutidos neste trabalho, na busca de uma otimização da relação sinal-ruído das imagens. Espera-se que o filtro casado obtenha o melhor desempenho de detecção, no entanto é interessante observar o comportamento dos outros filtros propostos em busca de uma melhor interpretação e entendimento dos resultados. As Figuras 30, 31, 32 e 33 mostram exemplos de alguns filtros sendo aplicados na imagem mostrada anteriormente, que contém um sinal de ER com 0.25 keV. Todas essas filtragens foram realizadas em um ambiente com Python (73) através da biblioteca SciPy (74).

Figura 30 – Exemplo imagem filtrada por filtro gaussiano com sigma igual a 5 e tamanho de janela 5 (esquerda) e 15 (direita).



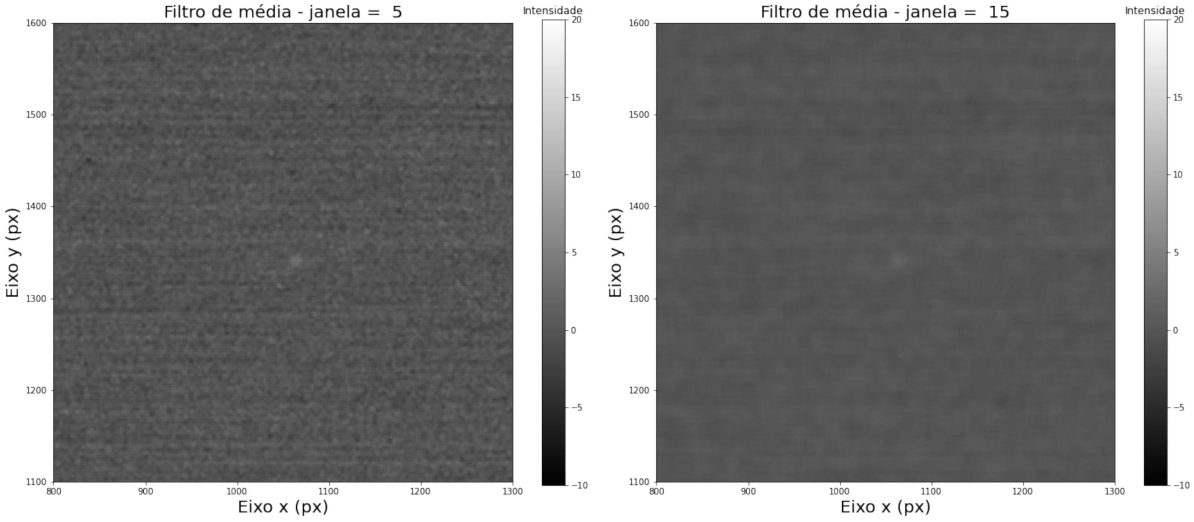
Fonte: Elaborada pelo autor (2024).

Figura 31 – Exemplo imagem filtrada por filtro casado com tamanho de janela 5 (esquerda) e 15 (direita).



Fonte: Elaborada pelo autor (2024).

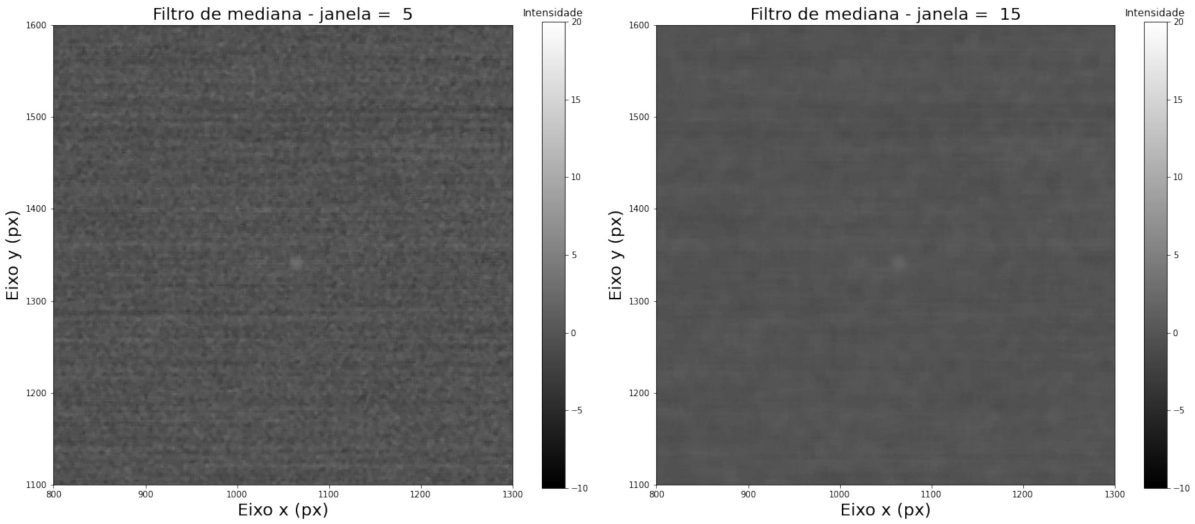
Figura 32 – Exemplo imagem filtrada por filtro média com tamanho de janela 5 (esquerda) e 15 (direita).



Fonte: Elaborada pelo autor (2024).

É possível notar que, em todos os casos, os filtros causaram o efeito de borramento nas imagens, uma característica comum para esses tipos de filtro, enquanto reduziram a quantidade de *hot-pixels* visíveis na Figura 29 e tornaram mais evidente o sinal, que antes possuía baixíssimo contraste com o fundo. Além disso, nas imagens filtradas com tamanho de janela 15, o pixel de maior intensidade (ou correlação) ocorreu exatamente na posição onde o sinal estava localizado.

Figura 33 – Exemplo imagem filtrada por filtro mediana com tamanho de janela 5 (esquerda) e 15 (direita).



Fonte: Elaborada pelo autor (2024).

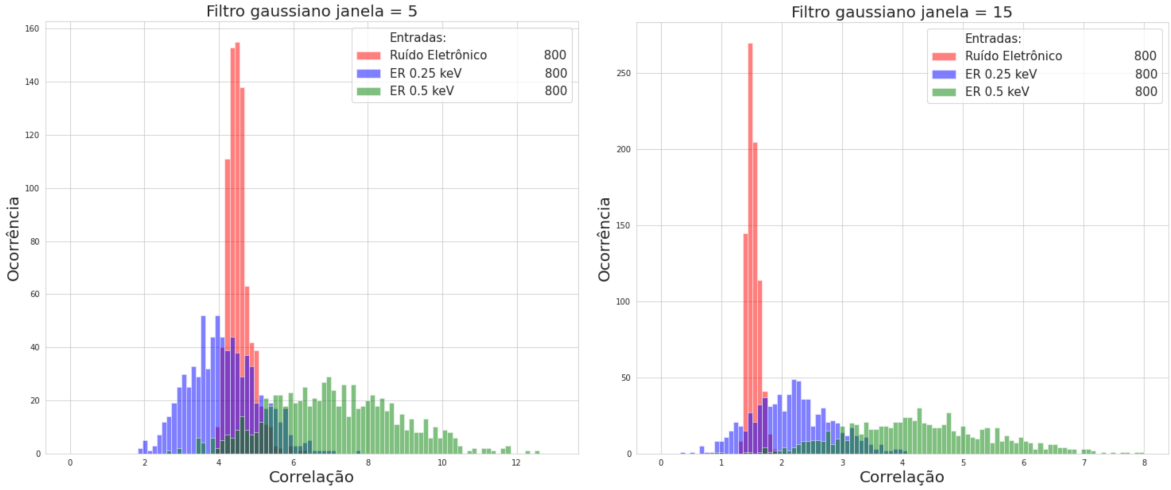
Entretanto, essa abordagem requer que o filtro e o limiar sejam configurados adequadamente para maximizar a performance do *trigger*. Essa configuração deve ser determinada na etapa de treinamento, utilizando os bancos de dados de treino e validação. Durante esse processo, os valores máximos de correlação são armazenados: para imagens apenas com ruído, busca-se a máxima correlação na imagem completa; para imagens com sinal, considera-se a máxima correlação apenas nos pixels onde o sinal está localizado. Isso resulta em duas distribuições distintas para cada configuração de filtro, como exemplificado na Figura 34 para o filtro gaussiano.

Para as configurações de filtro gaussiano mostradas na Figura 34, as distribuições dos sinais de ER estão mais distantes da distribuição de ruído eletrônico quando o tamanho da janela é 15. O sinal de 0.5 keV está quase totalmente à direita do ruído, enquanto o de 0.25 keV apresenta maior interseção. O objetivo é identificar a configuração que mantenha o ruído o mais distante possível do sinal e determinar um limiar eficaz para separá-los. Assim, o problema se transforma em uma discriminação binária, cuja metodologia será discutida na Seção **4.2.4**.

4.2.3 TRIGGER BASEADO EM CNN

Uma das principais aplicações das CNNs é a classificação de dados, e, no contexto de algoritmos de *trigger*, o objetivo é determinar se uma imagem contém um sinal. Conforme discutido na Seção 2.2, as CNNs têm a capacidade de aprender padrões diretamente dos dados sem a necessidade de conhecimento prévio específico sobre eles. No entanto, esse conhecimento pode ser aproveitado indiretamente na escolha da arquitetura da rede, como,

Figura 34 – Distribuições de correlação máxima após aplicação de filtro gaussiano com sigma 5, janela 5 (esquerda) e 15 (direita) em imagens que contém apenas ruído eletrônico (vermelho), sinais de ER com 0.25 keV (azul) e 0.5 keV (verde).



Fonte: Elaborada pelo autor (2024).

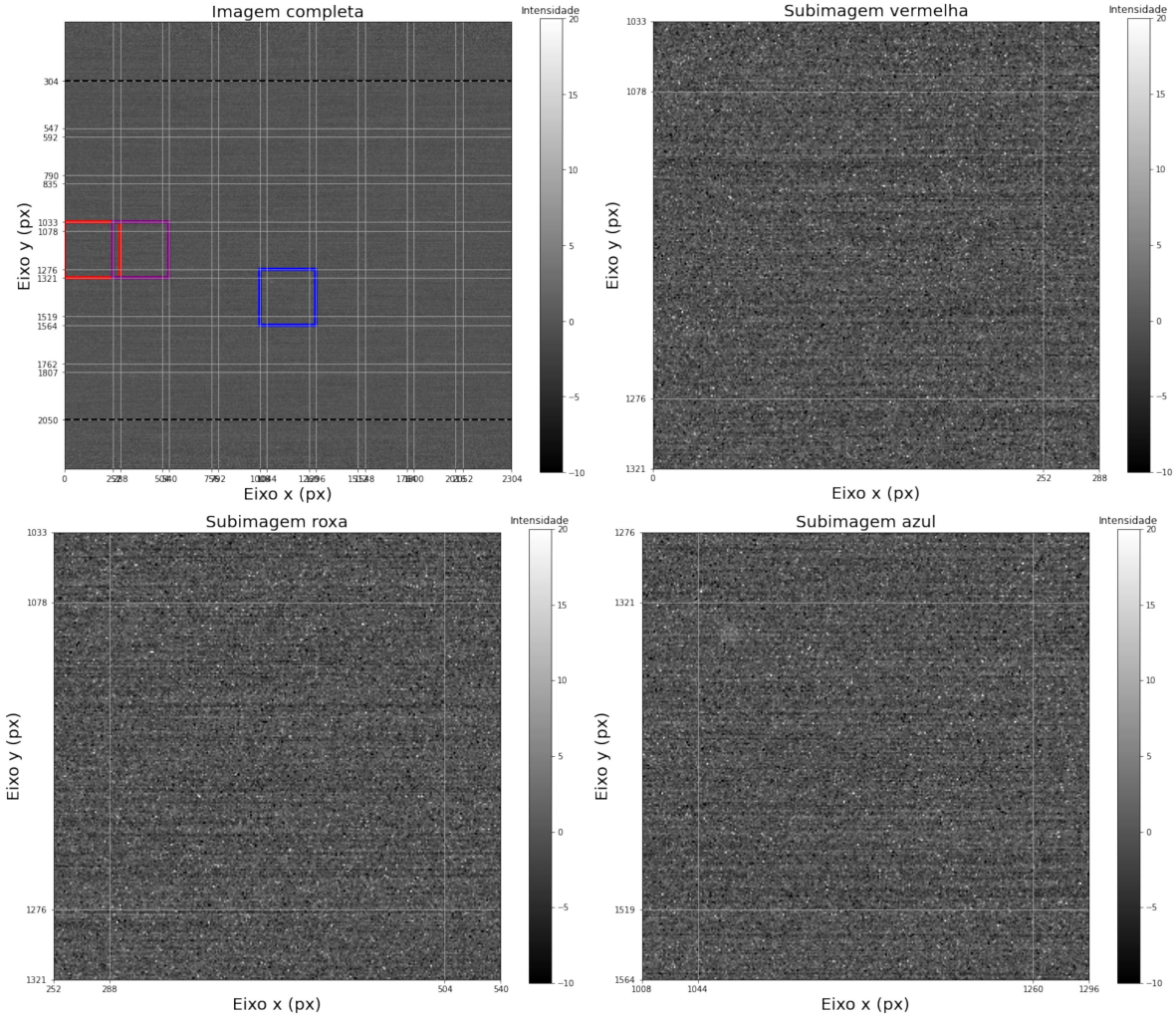
por exemplo, na definição da camada de entrada.

As imagens utilizadas neste trabalho têm dimensões de 2304x2304 pixels, o que torna o processamento direto por uma CNN inviável devido ao tamanho excessivo. Em contraste, muitas aplicações, como aquelas relacionadas ao desafio ILSVRC (*ImageNet Large Scale Visual Recognition Challenge*) (75), lidam com imagens muito menores, com 224x224 pixels, por exemplo, o que representa aproximadamente um décimo dos pixels em cada dimensão. Para viabilizar o processamento, a imagem precisa ser reduzida de alguma forma. No entanto, um simples *rescale*, como o aplicado no algoritmo de reconstrução, não é adequado neste caso, pois a imagem original é predominantemente dominada por ruído eletrônico, com apenas alguns pixels contendo informações de sinal. Portanto, uma abordagem mais eficaz é dividir a imagem em subimagens menores e analisá-las individualmente. Caso qualquer subimagem contenha um sinal, isso indica que a imagem original também contém um sinal. Se todas as subimagens forem classificadas como ruído, a imagem original será considerada como tal.

A Figura 35 ilustra um exemplo de uma imagem que contém um sinal de ER com 0.25 keV, dividida em subimagens de 288x288 pixels cada. Essa divisão foi realizada da seguinte forma: a região considerada é a mesma do algoritmo de reconstrução (entre as linhas tracejadas em preto). Subimagens adjacentes possuem uma área de sobreposição, garantindo que, caso um sinal ocorra próximo à fronteira entre duas subimagens, pelo menos uma delas terá uma quantidade suficiente de pixels de sinal para identificá-lo. O tamanho de 288 pixels em cada eixo foi escolhido por ser um múltiplo do tamanho original, compatível com outras aplicações e cerca de 10 vezes maior que o raio de um sinal de ER

e NR com 0.25 keV. Devido ao corte aplicado apenas no eixo y, a região de sobreposição neste eixo é menor do que no eixo x, já que as subimagens têm tamanho quadrado. Vale notar que a subimagem azul é onde o sinal está contido e, portanto, deve ser classificada como tal por uma CNN.

Figura 35 – Exemplo de imagem com ER de 0.25 keV em resolução original (esquerda superior) e suas subimagens vermelha (direita superior), roxa (esquerda inferior) e azul (direita inferior).



Fonte: Elaborada pelo autor (2024).

Após definir a camada de entrada, a próxima etapa é selecionar as camadas de convolução apropriadas para extrair características das imagens. Existem duas abordagens principais para aumentar a capacidade de aprendizado das CNNs nas camadas de convolução: (1) aumentar o número de filtros em cada camada e (2) aumentar a profundidade da rede adicionando mais camadas de convolução. Em geral, a segunda abordagem tende a oferecer melhores resultados, como demonstrado pelas arquiteturas desenvolvidas na ILSVRC, como AlexNet (76), VGGNet (77), GoogLeNet (78), ResNet (79) e DenseNet (80). No entanto, a profundidade excessiva da rede pode causar *overfitting*. Para mitigar

esse problema, essas arquiteturas empregam técnicas como *dropout*, regularização, *batch normalization* e outras estratégias específicas de cada modelo.

Por fim, a etapa de classificação deve ser definida, envolvendo a *Fully-Connected layer* e a camada de saída. Na primeira, é comum utilizar um número de neurônios que seja múltiplo de 32, em conjunto com técnicas como *dropout* e *batch normalization*. Após isso, o resultado deve ser transformado em uma probabilidade. No contexto de *trigger*, existem duas possibilidades para a classificação: a imagem pode conter um sinal ou ser composta apenas por ruído eletrônico. Portanto, uma camada de saída com um único neurônio é suficiente para realizar essa distinção.

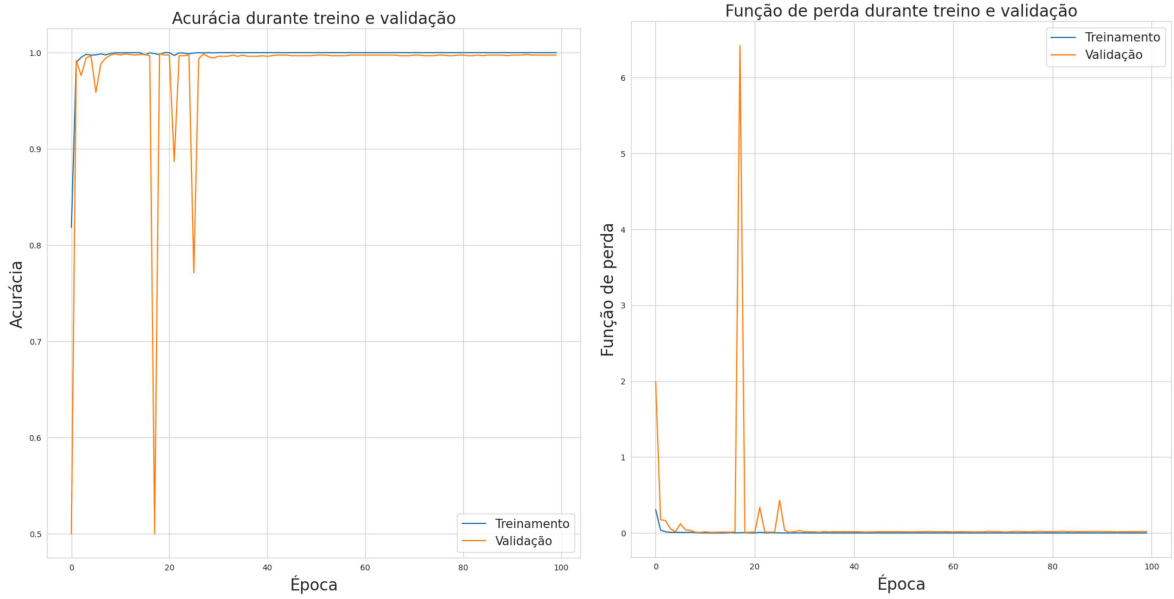
Após definir uma arquitetura, o próximo passo é treinar o modelo. Foram separadas 4800 imagens para treinamento e 1600 para validação, utilizando tanto ER quanto NR dos respectivos bancos de dados descritos em **4.2.1**. Metade das imagens são de ruído eletrônico, sendo uma subimagem aleatória escolhida, enquanto a outra metade contém sinais de ER e NR adicionados a uma subimagem de ruído aleatória. Cada evento de sinal foi replicado duas vezes para gerar este volume de dados. Técnicas de *data augmentation*, como rotação e deslocamento aleatório dos sinais dentro das imagens, foram aplicadas para aumentar a variabilidade dos dados e melhorar a robustez do modelo durante o treinamento (81).

Para otimizar a configuração da CNN, utilizou-se a otimização bayesiana, um método eficiente para encontrar o melhor conjunto de hiperparâmetros. Esta técnica constrói um modelo probabilístico da função de custo e usa-o para selecionar de forma inteligente os hiperparâmetros a serem testados em seguida, maximizando a performance do modelo com menos avaliações do que métodos de busca tradicionais (82, 83). Foram exploradas arquiteturas com 6, 7, 8 e 9 camadas de convolução, variando a inclusão de *batch normalization*, regularizadores, *dropout*, número de filtros em cada camada, taxa de aprendizado e número de neurônios na *Fully-Connected layer*.

O ambiente utilizado para o treinamento também foi o Python, mas dessa vez com a biblioteca TensorFlow (84). A Figura 36 ilustra o processo de treinamento de uma CNN com 7 camadas de convolução, em um dos passos da otimização bayesiana, empregando o otimizador Adam, conhecido por sua rápida convergência (85), e uma taxa de aprendizado de 0.0003. Observou-se que a CNN atingiu o pico de desempenho rapidamente, com estabilidade alcançada e a performance não decaindo mais após 30 épocas, tanto em termos de acurácia, que é a porcentagem de acertos da CNN, quanto de função de perda. A função de perda utilizada é a cross-entropia binária (BCE), cuja fórmula é apresentada na Equação 4.1. Esta função avalia a discrepância entre as probabilidades previstas pelo modelo (p_i) e os rótulos verdadeiros (y_i), sendo fundamental para determinar o melhor modelo no treinamento.

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)\right] \tag{4.1}$$

Figura 36 – Evolução da acurácia e perda de uma CNN com 7 camadas de convolução durante um treinamento por 100 épocas.



Fonte: Elaborada pelo autor (2024).

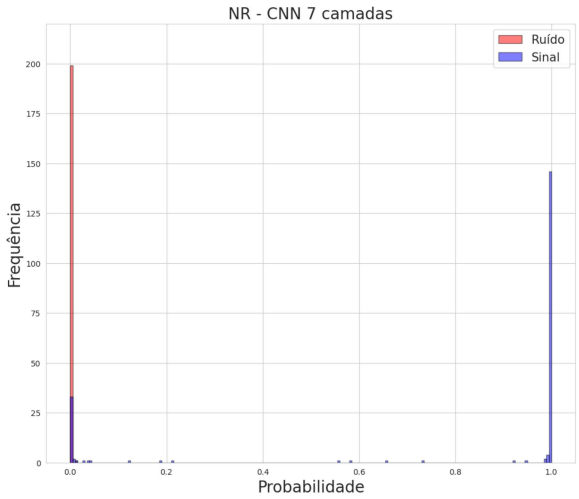
Uma vez definido o modelo durante o treinamento, este será aplicado ao banco de dados de teste para comparação com outros métodos. Devido a limitações computacionais, o treinamento foi realizado comparando subimagens individualmente para os rótulos de sinal e ruído. No entanto, durante a fase de teste, aplica-se a lógica de que uma imagem deve ser classificada como ruído somente se todas as suas subimagens forem de ruído. Assim, ao aplicar o modelo tanto em imagens com sinal quanto em imagens somente com ruído, obtêm-se duas distribuições distintas, conforme ilustrado na Figura 37. Para avaliar esses resultados, será aplicada uma metodologia de discriminação binária, a qual será detalhada na Seção **4.2.4**.

4.2.4 DISCRIMINAÇÃO BINÁRIA

As técnicas de *trigger* discutidas neste trabalho geram duas distribuições distintas: uma associada a imagens contendo sinal e outra a imagens que contêm apenas ruído eletrônico. Portanto, é essencial aplicar uma metodologia de discriminação binária para avaliar a eficácia do *trigger*, com foco na capacidade de separar essas distribuições de maneira precisa.

Para compreender as métricas extraídas dessas duas distribuições no contexto de classificação, é fundamental utilizar a matriz de confusão, conforme ilustrado na Tabela 1.

Figura 37 – Predições de uma CNN com 7 camadas de convolução em imagens com NR de 0.25 keV (vermelho) e apenas ruído eletrônico (azul)



Fonte: Elaborada pelo autor (2024).

Com base em um limiar definido, os verdadeiros positivos (TP) correspondem a eventos de sinal corretamente classificados como sinal (predição acima do limiar), enquanto os verdadeiros negativos (TN) são eventos de ruído corretamente classificados como ruído (predição abaixo do limiar). Os falsos negativos (FN) e falsos positivos (FP) são eventos classificados incorretamente em relação à sua classe real.

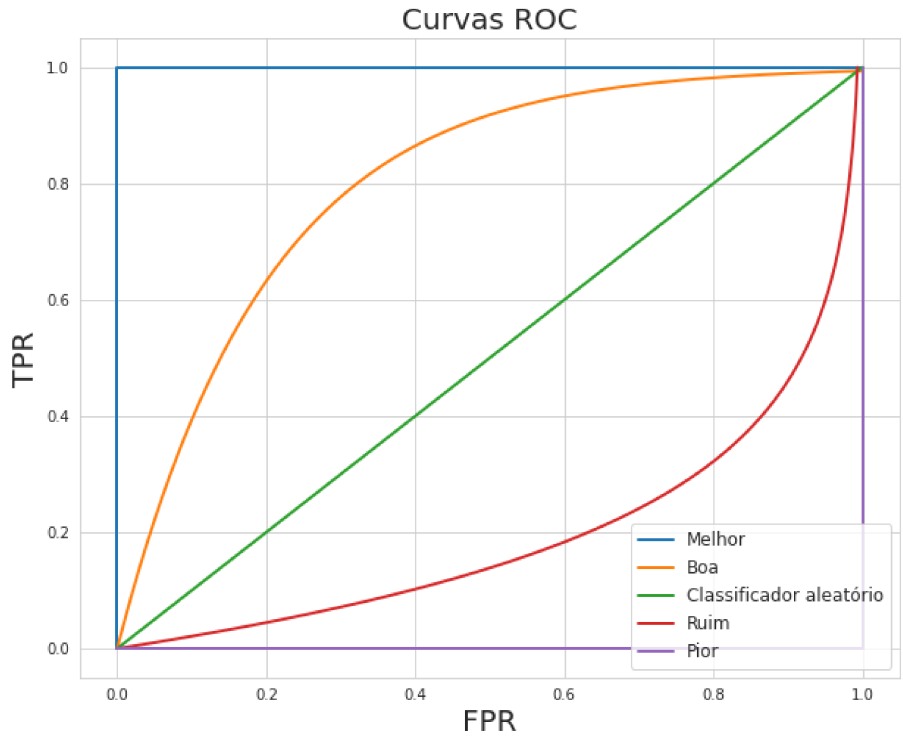
	Predito Positivo	**Predito Negativo**
Real Positivo	TP	FN
Real Negativo	FP	TN

Tabela 1 – Matriz de Confusão

Com esses valores definidos, pode-se utilizar uma das análises mais comuns na literatura: a *Receiver Operating Characteristic curve*, ou curva ROC. Para uma classificação binária, essa curva mostra como a taxa de verdadeiros positivos (TPR) e a taxa de falsos positivos (FPR) evoluem para todos os limiares possíveis entre as duas distribuições. A Figura 38 ilustra cinco exemplos de curvas ROC: (1) o melhor caso possível, quando as duas distribuições são totalmente separáveis; (2) um bom caso, quando a distribuição de sinal está ligeiramente à direita da de ruído; (3) um classificador aleatório, com uma probabilidade de acerto de 50%; (4) um caso ruim, quando a distribuição de sinal está ligeiramente à esquerda da de ruído; (5) o pior caso, quando a distribuição de sinal está totalmente à esquerda da de ruído. Os dois últimos casos geralmente indicam problemas com a definição dos bancos de dados ou com a formulação do problema, e, por isso, normalmente não são muito vistos na literatura (86).

Uma forma comum de avaliar a eficácia de uma curva ROC é através da métrica

Figura 38 – Exemplos de curva ROC: (1) melhor caso em azul; (2) um bom caso em laranja; (3) equivalente a um classificador aleatório em verde; (4) um caso ruim em vermelho e (5) o pior caso possível em roxo.



Fonte: Elaborada pelo autor (2024).

chamada *Area Under Curve* (AUC). Esta métrica calcula a área sob a curva ROC, como visto na Equação 4.2; portanto, quanto mais próxima a AUC estiver de 1, melhor será a separação entre os dados. O valor da AUC deve ser pelo menos igual ao que um classificador aleatório alcança, que é 0.5. Esta métrica é utilizada para definir a melhor configuração dos filtros no *trigger* baseado em filtragem e para comparar o desempenho entre as CNNs e os filtros.

$$\text{AUC} = \int_0^1 \text{ROC}(x)\, dx \tag{4.2}$$

No entanto, a aplicação do *trigger* ainda requer a definição de um ponto de operação específico, ou seja, uma configuração de CNN ou filtro e um limiar. Idealmente, o limiar deve ser escolhido de modo a maximizar a separação entre os dados, aproximando TPR e FPR de 1 e 0, respectivamente. Por outro lado, um limiar também pode ser escolhido fixando um valor específico de TPR ou FPR, como por exemplo, 0.8 de TPR, que significaria que 80% dos sinais estão sendo detectados, ou 0.1 de FPR, que significaria um falso alarme de 10%. Assim, pode-se observar como a outra métrica se comporta para esse valor fixo e se um limiar encontrado no treino mantém resultados parecidos no banco de teste.

5 Resultados

Este capítulo apresentará os resultados obtidos com a aplicação dos algoritmos de *trigger* baseados em filtragem e em CNN nos bancos de dados de treino, validação e teste. Os dois primeiros serão utilizados para a seleção dos pontos de operação ideais para cada método, enquanto o último será empregado para a comparação direta entre os diferentes métodos. Em seguida, uma análise do tempo de execução de cada algoritmo será apresentada, uma vez que a eficiência temporal é tão importante quanto a performance de detecção para um algoritmo de *trigger*. Finalmente, será feita uma comparação entre o melhor ponto de operação de cada método e o algoritmo de reconstrução.

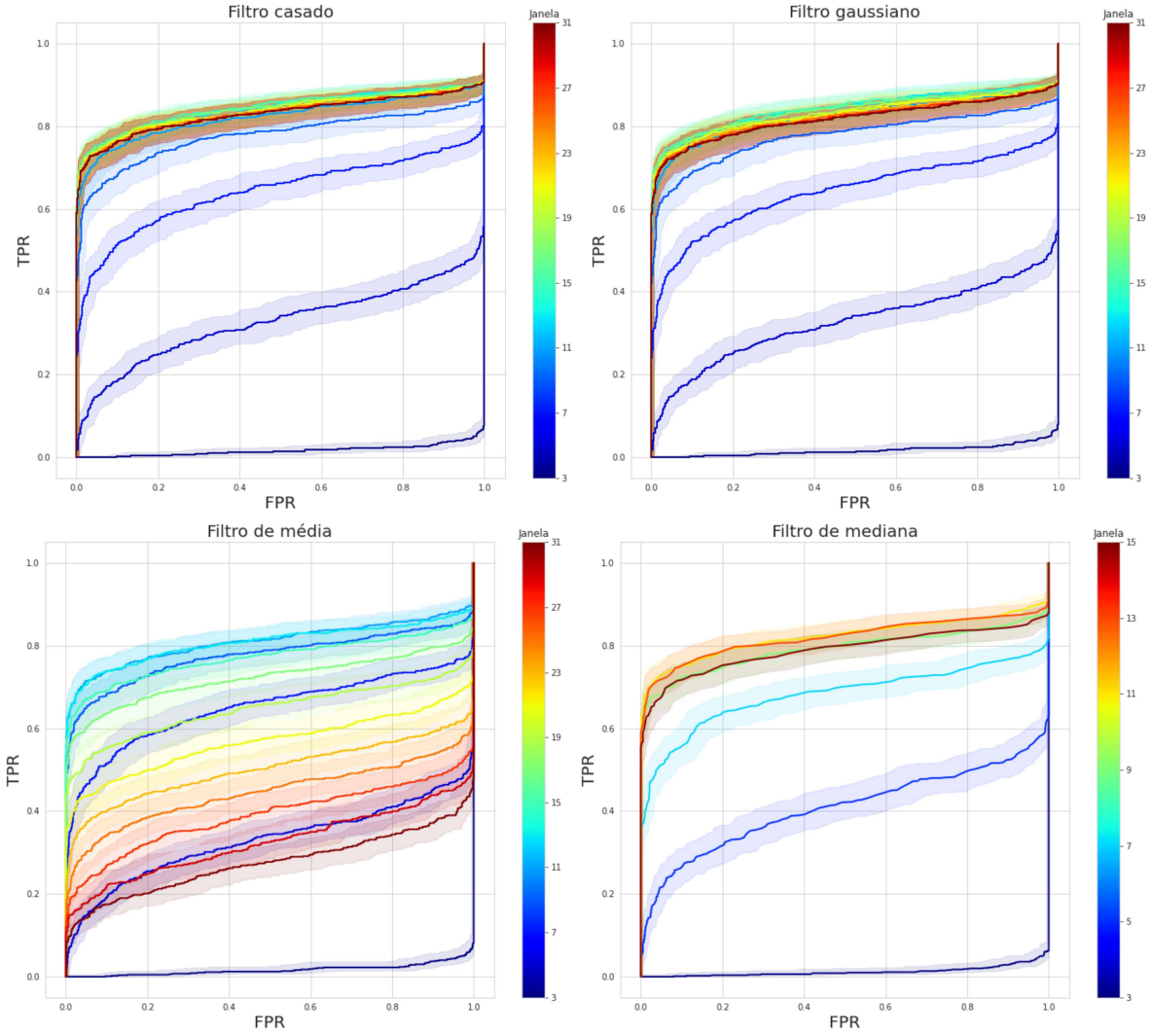
5.1 TRIGGER BASEADO EM FILTRAGEM

Conforme discutido nas Seções **4.2.1** e **4.2.2**, um *trigger* baseado em filtragem apresenta menor risco de *overfitting* em comparação às CNNs, pois seu modelo não se ajusta aos dados a cada iteração. Para determinar a melhor configuração de filtro, os conjuntos de dados de treino e validação foram combinados, focando em eventos de 0.25 keV, que representam a condição mais adversa possível. O conjunto de teste foi reservado para comparação de resultados, abrangendo tanto a energia de 0.25 keV quanto energias maiores.

A Figura 39 mostra a evolução das curvas ROC para os filtros casado, gaussiano (com sigma 5), média e mediana em diferentes tamanhos de janela, considerando os sinais de NR com 0.25 keV. Observa-se que, para os filtros gaussiano e casado, há uma maior concentração de curvas potencialmente boas, enquanto os filtros de mediana e média apresentam menos candidatos promissores. Os tamanhos de janela variaram de 3 a 31 para todos os filtros, exceto para o filtro de mediana, cujo intervalo foi limitado de 3 a 15 por razões computacionais. O filtro casado foi criado a partir da média de 50 sinais do banco de dados de treinamento, com filtros distintos para ER e NR, e, apesar de já possuir um tamanho definido, foi truncado seguindo esses intervalos de janela para manter a isonomia da metodologia.

Cada curva ROC possui um valor de AUC associado, utilizado para identificar a configuração ótima de cada tipo de filtro. A Figura 40 apresenta a variação da AUC para diferentes valores de sigma e tamanhos de janela no filtro gaussiano, considerando NR (esquerda) e ER (direita) com 0.25 keV. Observa-se que, para ambos os tipos de sinal, os maiores valores de AUC concentram-se na faixa de sigma entre 4 e 7 e tamanhos de janela entre 11 e 17. Além disso, a AUC mantém-se estável para sigma igual a 4 e 5 com tamanhos de janela superiores a 15, enquanto para valores de sigma maiores, a AUC começa a diminuir. Isso sugere que o sigma ideal seria 4 ou 5, sendo o último escolhido como valor final.

Figura 39 – Curvas ROC para diferentes tamanhos de janela com os filtros casado (esquerda superior), gaussiano (direita superior), média (esquerda inferior) e mediana (direita inferior) considerando sinais de NR com 0.25 keV e ruído eletrônico.

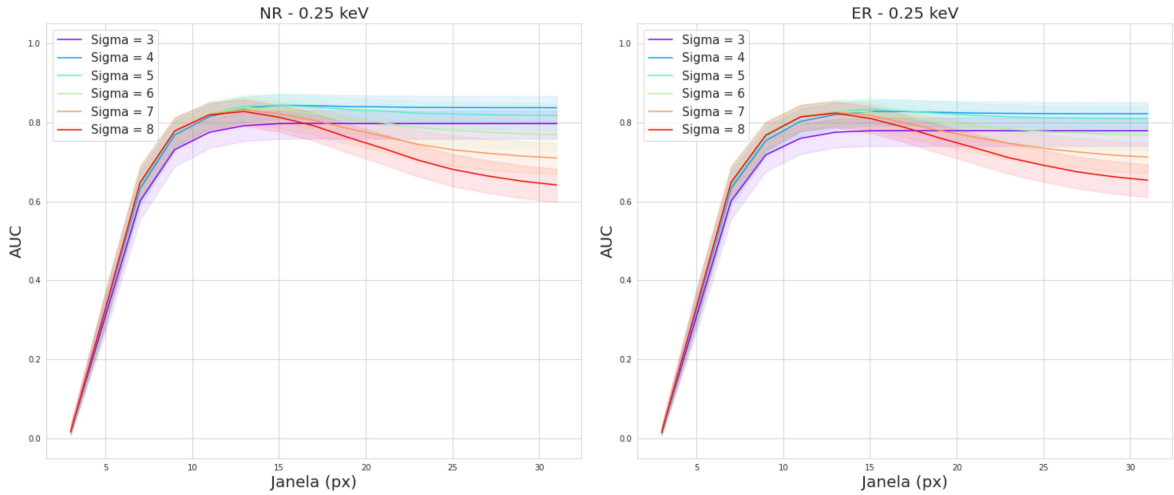


Fonte: Elaborada pelo autor (2024).

A Figura 41 ilustra a evolução da AUC para os quatro filtros selecionados, considerando NR (esquerda) e ER (direita). Os filtros de média e mediana mostram uma AUC ligeiramente menor em seus picos, enquanto os filtros gaussiano (com sigma 5) e casado se destacam com valores de AUC semelhantes. Este comportamento deve-se à proximidade dos sinais a uma distribuição gaussiana 2D com o valor de sigma utilizado, resultando em um desempenho similar entre o filtro gaussiano e o que seria um filtro casado parametrizado. Portanto, os pontos de operação selecionados foram: filtro casado com janela de 15; filtro gaussiano com janela de 15 e sigma 5; e filtros de média e mediana com janela de 11.

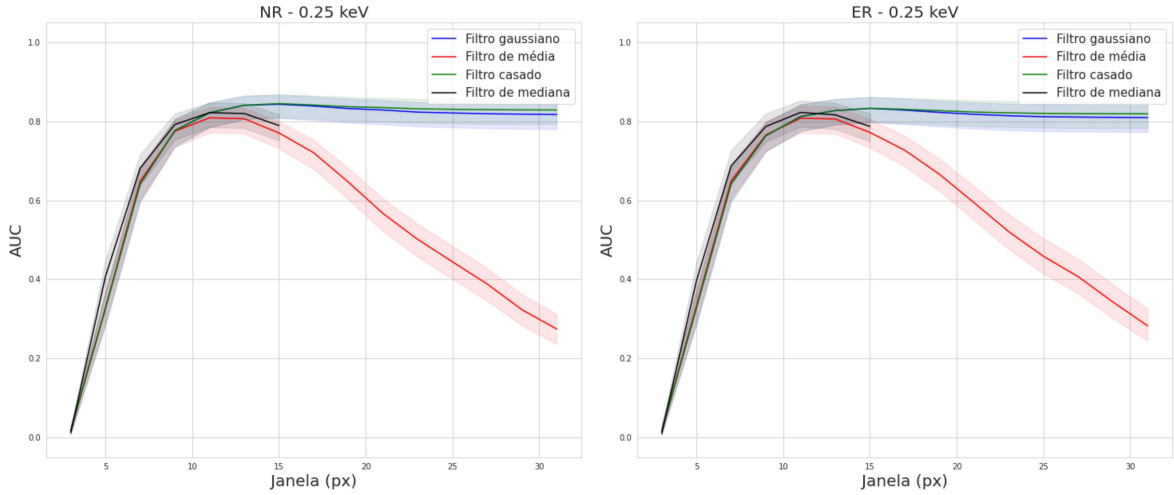
Após a seleção dos pontos de operação para cada filtro, estes foram aplicados ao banco de dados de teste com energias de 0.25 e 0.5 keV, conforme ilustrado nas Figuras

Figura 40 – Evolução da AUC para diferentes valores de sigma e janela no filtro gaussiano para NR (esquerda) e ER (direita) com 0.25 keV.



Fonte: Elaborada pelo autor (2024).

Figura 41 – Evolução da AUC dos filtros selecionados para NR (esquerda) e ER (direita) com 0.25 keV.

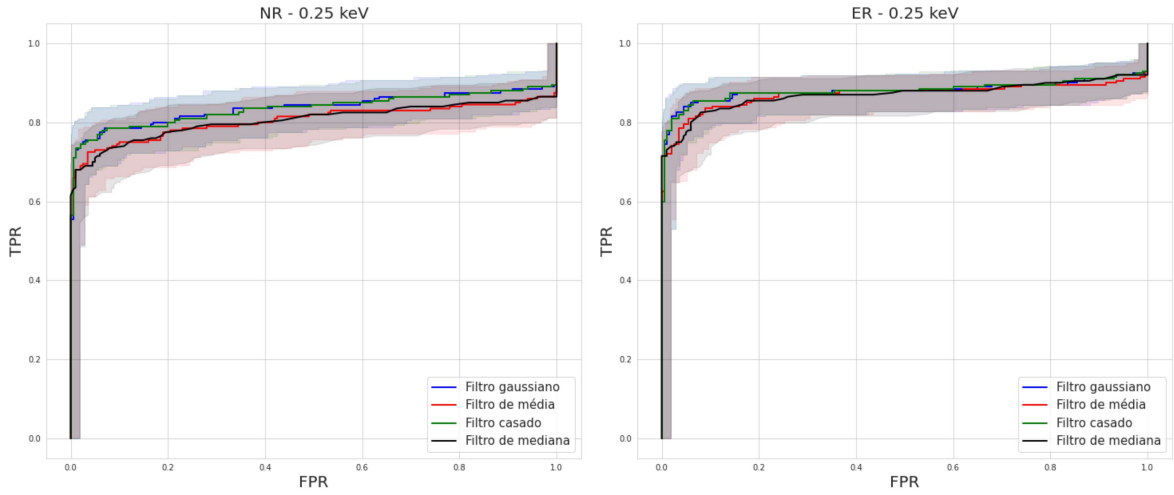


Fonte: Elaborada pelo autor (2024).

42 e 43. Os filtros gaussiano e casado continuam a demonstrar a melhor performance em 0.25 keV, enquanto todos os filtros apresentam curvas ROC quase perfeitas em 0.5 keV, indicando uma separabilidade quase total entre as distribuições de sinal e ruído. As Tabelas 2 e 3 mostram os valores de AUC para cada filtro em 0.25 e 0.5 keV. Para energias superiores, as distribuições de sinal e ruído foram totalmente separáveis, resultando em valores de AUC iguais a 1.

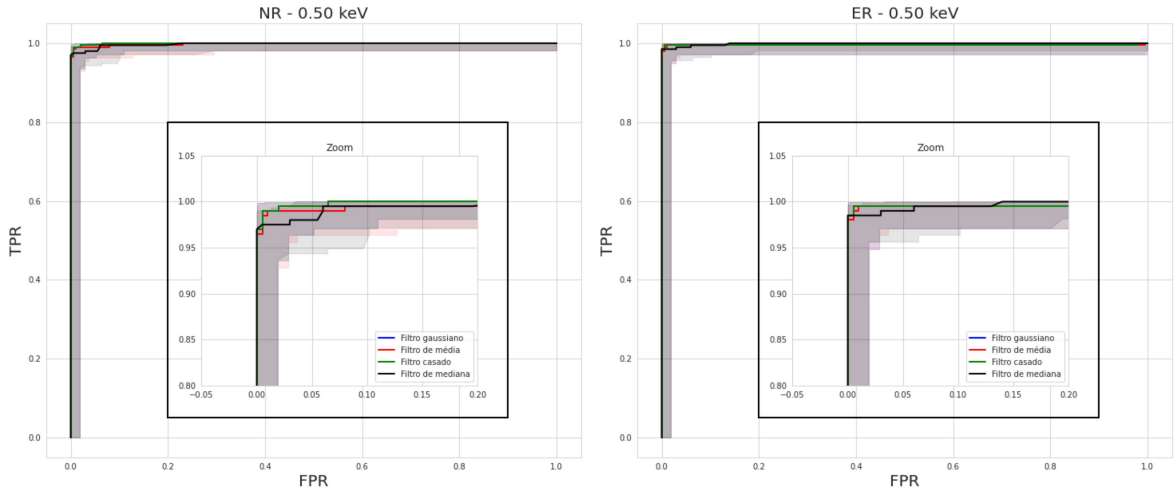
Os valores de AUC alcançados no treino e no teste foram consistentes para 0.25 keV, com exceção dos sinais de ER, que apresentaram uma performance ligeiramente melhor no banco de dados de teste em comparação ao treino. Os intervalos de confiança são

Figura 42 – Curvas ROC dos filtros aplicados no banco de dados de teste com os pontos de operação selecionados no treino para NR (esquerda) e ER (direita) com 0.25 keV.



Fonte: Elaborada pelo autor (2024).

Figura 43 – Curvas ROC dos filtros aplicados no banco de dados de teste com os pontos de operação selecionados no treino para NR (esquerda) e ER (direita) com 0.5 keV.



Fonte: Elaborada pelo autor (2024).

naturalmente menores no treino devido à maior quantidade de dados disponíveis. Esses intervalos foram calculados com 95% de nível de confiança, utilizando um método baseado na distribuição binomial (87, 88).

Além da seleção do ponto de operação do filtro, a escolha de um limiar adequado baseado em uma métrica desejada é crucial para otimizar o desempenho do *trigger*. Por exemplo, deseja-se que o algoritmo de *trigger* aceite um falso alarme de até 10%, correspondente a um FPR de 0.1 nas curvas ROC, pode-se selecionar um limiar que atenda a esse critério no banco de dados de treino. A Figura 44 demonstra como a escolha desse

Tabela 2 – AUC das curvas ROC dos filtros nos pontos de operação selecionados para NR e ER com 0.25 keV

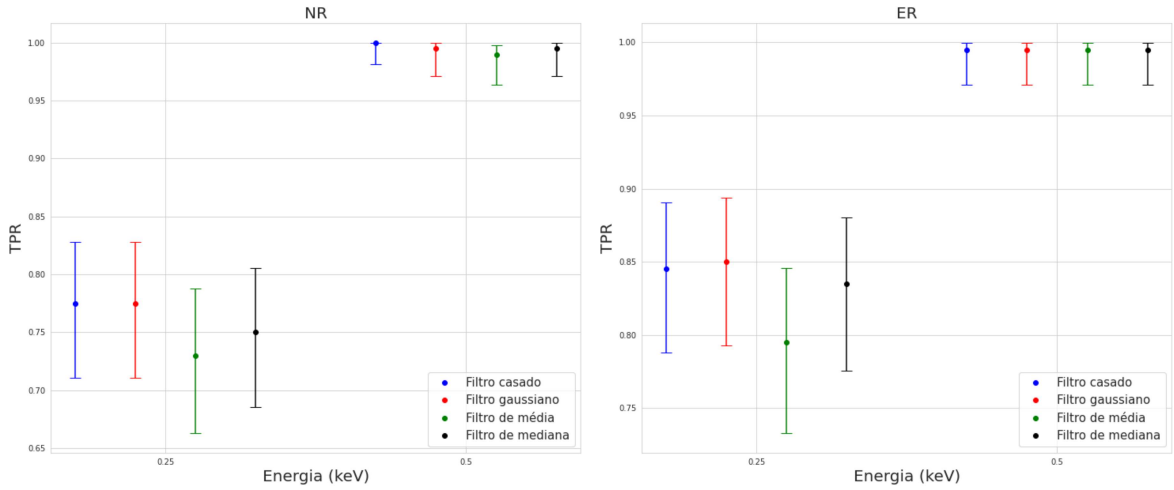
Filtro	**NR**		**ER**	
	Treino	**Teste**	**Treino**	**Teste**
Casado	$0.8448\pm^{0.0283}_{0.0351}$	$0.8334\pm^{0.0548}_{0.0801}$	$0.8332\pm^{0.0287}_{0.0352}$	$0.8810\pm^{0.0449}_{0.0743}$
Gaussiano	$0.8433\pm^{0.0284}_{0.0352}$	$0.8345\pm^{0.0544}_{0.0799}$	$0.8328\pm^{0.0287}_{0.0352}$	$0.8802\pm^{0.0450}_{0.0741}$
Média	$0.8089\pm^{0.0315}_{0.0376}$	$0.8019\pm^{0.0597}_{0.0829}$	$0.8079\pm^{0.0310}_{0.0371}$	$0.8706\pm^{0.0476}_{0.0760}$
Mediana	$0.8219\pm^{0.0304}_{0.0368}$	$0.8009\pm^{0.0600}_{0.0831}$	$0.8226\pm^{0.0301}_{0.0365}$	$0.8700\pm^{0.0476}_{0.0763}$

Tabela 3 – AUC das curvas ROC dos filtros nos pontos de operação selecionados para NR e ER com 0.5 keV

Filtro	**Teste NR**	**Teste ER**
Casado	$0.9994\pm^{0.0006}_{0.0383}$	$0.9950\pm^{0.0048}_{0.0422}$
Gaussiano	$0.9994\pm^{0.0006}_{0.0383}$	$0.9950\pm^{0.0048}_{0.0422}$
Média	$0.9969\pm^{0.0027}_{0.0407}$	$0.9949\pm^{0.0048}_{0.0422}$
Mediana	$0.9965\pm^{0.0029}_{0.0410}$	$0.9989\pm^{0.0010}_{0.0388}$

limiar impacta na detecção dos sinais de NR (esquerda) e ER (direita) de 0.25 e 0.5 keV no banco de dados de teste. Novamente, os filtros gaussiano e casado destacam-se dos demais em 0.25 keV, enquanto a performance em 0.5 keV é mais próxima. Outro ponto a destacar é que o falso alarme no banco de dados de teste com esse limiar se manteve em 10%, exceto para o filtro de mediana, que apresentou um falso alarme de 10.87%. Isso demonstra a robustez do método em relação às distribuições de ruído.

Figura 44 – Deteccção dos sinais de NR (esquerda) e ER (direita) para cada filtro com um limiar referente a um falso alarme de 10%.

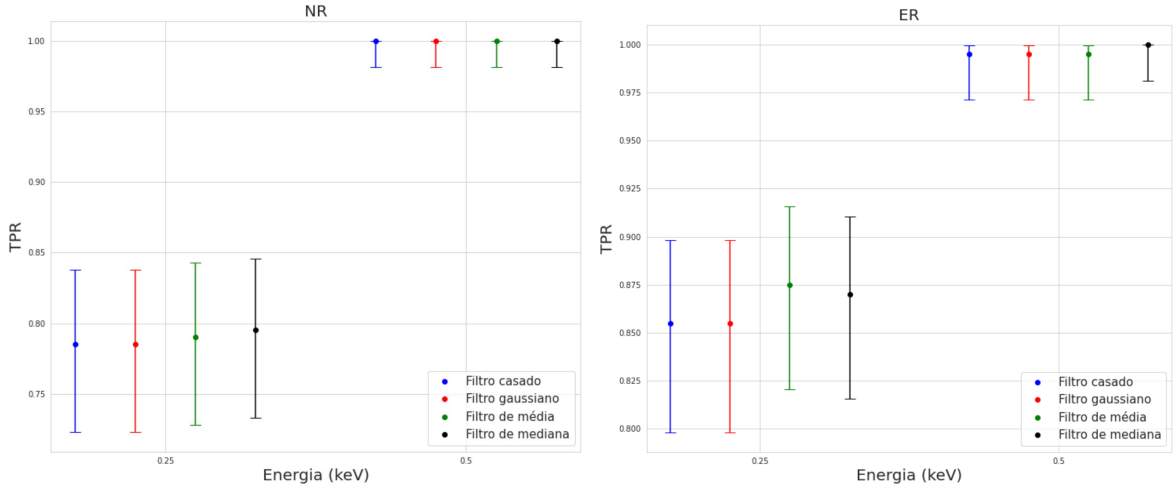


Fonte: Elaborada pelo autor (2024).

Analogamente, pode-se definir um limiar baseado na detecção de sinais desejada. A Figura 45 mostra o comportamento dos sinais de NR (esquerda) e ER (direita) no banco de dados de teste com um limiar que detecta 80% desses sinais no treino. Nota-se que todos

os filtros mantêm uma detecção de sinal consistente com esse valor, exceto os sinais de ER, que apresentaram um valor de detecção ligeiramente maior, comportamento também observado nas medidas de AUC. Em contrapartida, a Tabela 4 mostra o comportamento do falso alarme com esses limiares, tanto no treino quanto no teste. Verifica-se que, para um valor próximo de detecção de sinal, os filtros gaussiano e casado apresentam um falso alarme significativamente menor em comparação aos demais.

Figura 45 – Deteccção dos sinais de NR (esquerda) e ER (direita) para cada filtro com um limiar referente a uma detecção de sinal de 80%.



Fonte: Elaborada pelo autor (2024).

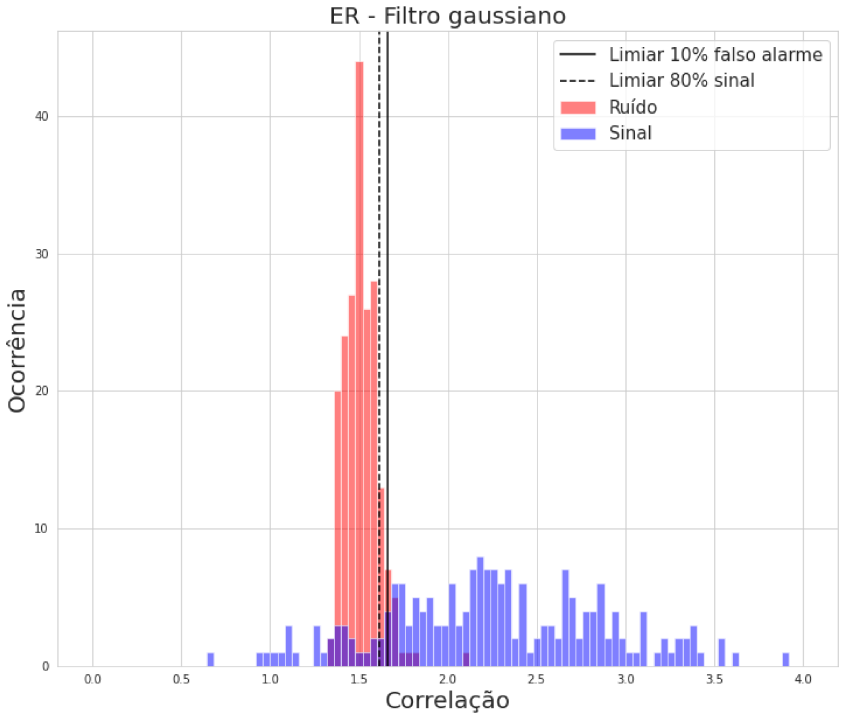
Tabela 4 – Falso alarme dos filtros no treino e teste na presença de um limiar que detecta 80% dos sinais de 0.25 keV.

Filtro	**NR**		**ER**	
	Treino	**Teste**	**Treino**	**Teste**
Casado	$0.17\pm^{0.0280}_{0.0250}$	$0.0900\pm^{0.0493}_{0.0332}$	$0.1613\pm^{0.0274}_{0.0245}$	$0.0900\pm^{0.0493}_{0.0332}$
Gaussiano	$0.1775\pm^{0.0280}_{0.0251}$	$0.1150\pm^{0.0520}_{0.0382}$	$0.1837\pm^{0.0286}_{0.0258}$	$0.1150\pm^{0.0520}_{0.0382}$
Média	$0.3387\pm^{0.0337}_{0.0326}$	$0.2900\pm^{0.0623}_{0.0655}$	$0.3575\pm^{0.0343}_{0.0326}$	$0.3200\pm^{0.0688}_{0.0629}$
Mediana	$0.2762\pm^{0.0324}_{0.0321}$	$0.2900\pm^{0.0623}_{0.0655}$	$0.2762\pm^{0.0324}_{0.0301}$	$0.2900\pm^{0.0623}_{0.0655}$

A Figura 46 apresenta as distribuições de ER com 0.25 keV e ruído para o filtro gaussiano com configuração ótima no banco de teste. Nota-se que, mesmo na configuração que maximiza a performance do filtro, ainda existe interseção entre as distribuições, explicando os valores de detecção de sinal e falso alarme para cada limiar escolhido anteriormente. Esses limiares estão próximos, como mostrado na figura, justificando os resultados similares em detecção de sinal e falso alarme.

A partir desses resultados, conclui-se que os filtros gaussiano (com sigma 5) e casado apresentaram os melhores resultados, estando tecnicamente empatados na detecção. A vantagem do filtro gaussiano é que, por não ter sua máscara criada a partir dos dados

Figura 46 – Distribuições de correlação após aplicação do filtro gaussiano com janela 15 e sigma 5 para imagens com ER de 0.25 keV e apenas ruído eletrônico



Fonte: Elaborada pelo autor (2024).

específicos, possui maior potencial para ser utilizado na detecção de outros tipos de sinal. Além disso, a performance temporal de cada um desses filtros deve ser verificada antes de escolher o melhor, tema que será abordado na Seção 5.3.

5.2 TRIGGER BASEADO EM CNN

Como descrito na Seção **4.2.3**, as CNNs têm uma tendência maior ao *overfitting* em comparação com métodos de filtragem. Para mitigar esse risco, os bancos de treino e validação foram utilizados de forma separada: o banco de treino foi empregado para o ajuste dos hiperparâmetros da rede, enquanto o banco de validação foi usado para selecionar a melhor configuração da rede.

Devido a limitações computacionais, a lógica de *trigger* com imagens completas não pôde ser implementada na busca pela melhor configuração de CNN. Portanto, durante a otimização bayesiana, subimagens rotuladas como sinal e ruído foram comparadas individualmente para CNNs com 6, 7, 8 e 9 camadas de convolução. Observou-se que o uso de sinais de 0.25 keV no treino frequentemente levava ao *overfitting*, enquanto o treino com sinais de 0.5 keV apresentava bons resultados que se mantinham para sinais de 0.25 keV.

As arquiteturas resultantes para cada configuração estão apresentadas na Figura

47. Durante a construção das redes, notou-se que a saída após a quinta camada de *max pooling* apresentava um tamanho de 9x9 pixels. Por se tratar de uma dimensão ímpar, o uso de *max pooling* 2x2 com *strides* de 2 tornou-se inviável. Para contornar essa limitação e permitir a utilização de 8 ou 9 camadas, foi decidido aplicar *max pooling* 2x2 com *strides* de 1, reduzindo o tamanho da imagem para 8x8, e, em seguida, retornar a *strides* de 2 nas camadas subsequentes. Para as arquiteturas de 6 e 7 camadas, constatou-se que *max pooling* 3x3 com *strides* de 3 após a quinta camada de convolução proporcionou melhores resultados.

Figura 47 – Arquiteturas de CNN encontradas através da otimização bayesiana com 6 (esquerda superior), 7 (direita superior), 8 (esquerda inferior) e 9 (direita inferior) camadas.

Layer (type)	Output Shape	Param #
batch_normalization (BatchNormalization)	(None, 288, 288, 1)	4
conv2d (Conv2D)	(None, 288, 288, 96)	960
max_pooling2d (MaxPooling2D)	(None, 144, 144, 96)	0
conv2d_1 (Conv2D)	(None, 144, 144, 64)	55,360
max_pooling2d_1 (MaxPooling2D)	(None, 72, 72, 64)	0
conv2d_2 (Conv2D)	(None, 72, 72, 64)	36,928
max_pooling2d_2 (MaxPooling2D)	(None, 36, 36, 64)	0
conv2d_3 (Conv2D)	(None, 36, 36, 224)	129,248
max_pooling2d_3 (MaxPooling2D)	(None, 18, 18, 224)	0
conv2d_4 (Conv2D)	(None, 18, 18, 192)	387,264
max_pooling2d_4 (MaxPooling2D)	(None, 6, 6, 192)	0
conv2d_5 (Conv2D)	(None, 6, 6, 160)	276,640
max_pooling2d_5 (MaxPooling2D)	(None, 2, 2, 160)	0
flatten (Flatten)	(None, 640)	0
dense (Dense)	(None, 416)	266,656
batch_normalization_1 (BatchNormalization)	(None, 416)	1,664
dropout (Dropout)	(None, 416)	0
dense_1 (Dense)	(None, 1)	417

Total params: 1,155,141 (4.41 MB)
Trainable params: 1,154,307 (4.40 MB)
Non-trainable params: 834 (3.26 KB)

Layer (type)	Output Shape	Param #
batch_normalization (BatchNormalization)	(None, 288, 288, 1)	4
conv2d (Conv2D)	(None, 288, 288, 96)	960
max_pooling2d (MaxPooling2D)	(None, 144, 144, 96)	0
conv2d_1 (Conv2D)	(None, 144, 144, 64)	55,360
max_pooling2d_1 (MaxPooling2D)	(None, 72, 72, 64)	0
conv2d_2 (Conv2D)	(None, 72, 72, 160)	92,320
max_pooling2d_2 (MaxPooling2D)	(None, 36, 36, 160)	0
conv2d_3 (Conv2D)	(None, 36, 36, 96)	138,336
max_pooling2d_3 (MaxPooling2D)	(None, 18, 18, 96)	0
conv2d_4 (Conv2D)	(None, 18, 18, 256)	221,440
max_pooling2d_4 (MaxPooling2D)	(None, 9, 9, 256)	0
conv2d_5 (Conv2D)	(None, 9, 9, 224)	516,320
max_pooling2d_5 (MaxPooling2D)	(None, 3, 3, 224)	0
conv2d_6 (Conv2D)	(None, 3, 3, 256)	516,352
max_pooling2d_6 (MaxPooling2D)	(None, 1, 1, 256)	0
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 320)	82,240
batch_normalization_1 (BatchNormalization)	(None, 320)	1,280
dropout (Dropout)	(None, 320)	0
dense_1 (Dense)	(None, 1)	321

Total params: 1,624,933 (6.20 MB)
Trainable params: 1,624,291 (6.20 MB)
Non-trainable params: 642 (2.51 KB)

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 288, 288, 128)	1,280
max_pooling2d (MaxPooling2D)	(None, 144, 144, 128)	0
conv2d_1 (Conv2D)	(None, 144, 144, 96)	110,688
max_pooling2d_1 (MaxPooling2D)	(None, 72, 72, 96)	0
conv2d_2 (Conv2D)	(None, 72, 72, 64)	55,360
max_pooling2d_2 (MaxPooling2D)	(None, 36, 36, 64)	0
conv2d_3 (Conv2D)	(None, 36, 36, 32)	18,464
max_pooling2d_3 (MaxPooling2D)	(None, 18, 18, 32)	0
conv2d_4 (Conv2D)	(None, 18, 18, 160)	46,240
max_pooling2d_4 (MaxPooling2D)	(None, 9, 9, 160)	0
conv2d_5 (Conv2D)	(None, 9, 9, 256)	368,896
max_pooling2d_5 (MaxPooling2D)	(None, 8, 8, 256)	0
conv2d_6 (Conv2D)	(None, 8, 8, 64)	147,520
max_pooling2d_6 (MaxPooling2D)	(None, 4, 4, 64)	0
conv2d_7 (Conv2D)	(None, 4, 4, 128)	73,856
max_pooling2d_7 (MaxPooling2D)	(None, 2, 2, 128)	0
flatten (Flatten)	(None, 512)	0
dense (Dense)	(None, 480)	246,240
batch_normalization (BatchNormalization)	(None, 480)	1,920
dropout (Dropout)	(None, 480)	0
dense_1 (Dense)	(None, 1)	481

Total params: 1,070,945 (4.09 MB)
Trainable params: 1,069,985 (4.08 MB)
Non-trainable params: 960 (3.75 KB)

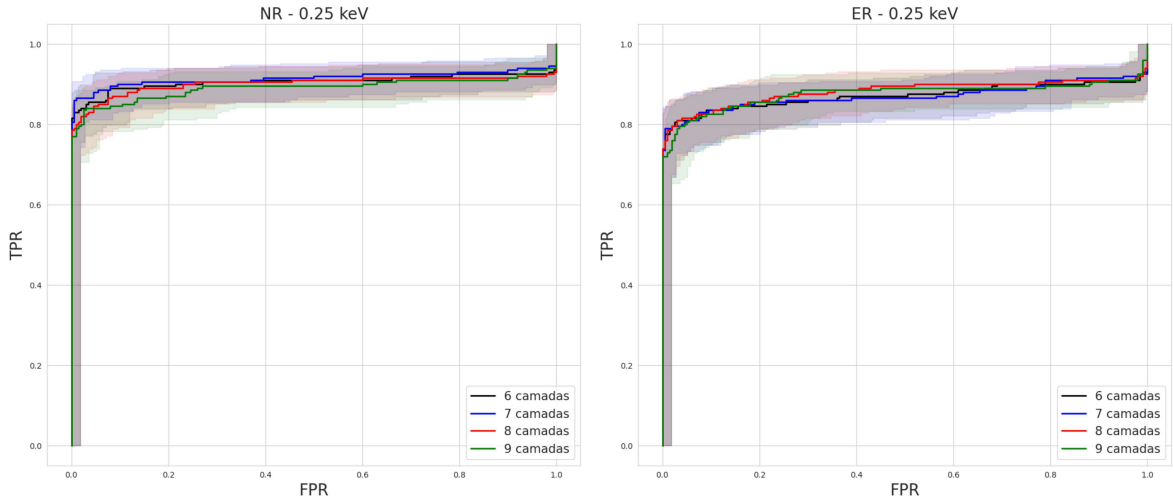
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 288, 288, 64)	640
max_pooling2d (MaxPooling2D)	(None, 144, 144, 64)	0
conv2d_1 (Conv2D)	(None, 144, 144, 32)	18,464
max_pooling2d_1 (MaxPooling2D)	(None, 73, 73, 32)	0
conv2d_2 (Conv2D)	(None, 72, 72, 64)	18,496
max_pooling2d_2 (MaxPooling2D)	(None, 36, 36, 64)	0
conv2d_3 (Conv2D)	(None, 36, 36, 32)	18,464
max_pooling2d_3 (MaxPooling2D)	(None, 18, 18, 32)	0
conv2d_4 (Conv2D)	(None, 18, 18, 224)	64,736
max_pooling2d_4 (MaxPooling2D)	(None, 9, 9, 224)	0
conv2d_5 (Conv2D)	(None, 8, 9, 224)	451,808
max_pooling2d_5 (MaxPooling2D)	(None, 8, 8, 224)	0
conv2d_6 (Conv2D)	(None, 8, 8, 256)	516,352
max_pooling2d_6 (MaxPooling2D)	(None, 4, 4, 256)	0
conv2d_7 (Conv2D)	(None, 4, 4, 128)	295,040
max_pooling2d_7 (MaxPooling2D)	(None, 2, 3, 128)	0
conv2d_8 (Conv2D)	(None, 2, 2, 256)	295,168
max_pooling2d_8 (MaxPooling2D)	(None, 1, 1, 256)	0
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 160)	41,120
batch_normalization (BatchNormalization)	(None, 160)	640
dropout (Dropout)	(None, 160)	0
dense_1 (Dense)	(None, 1)	161

Total params: 1,721,089 (6.57 MB)
Trainable params: 1,720,769 (6.56 MB)
Non-trainable params: 320 (1.25 KB)

Fonte: Elaborada pelo autor (2024).

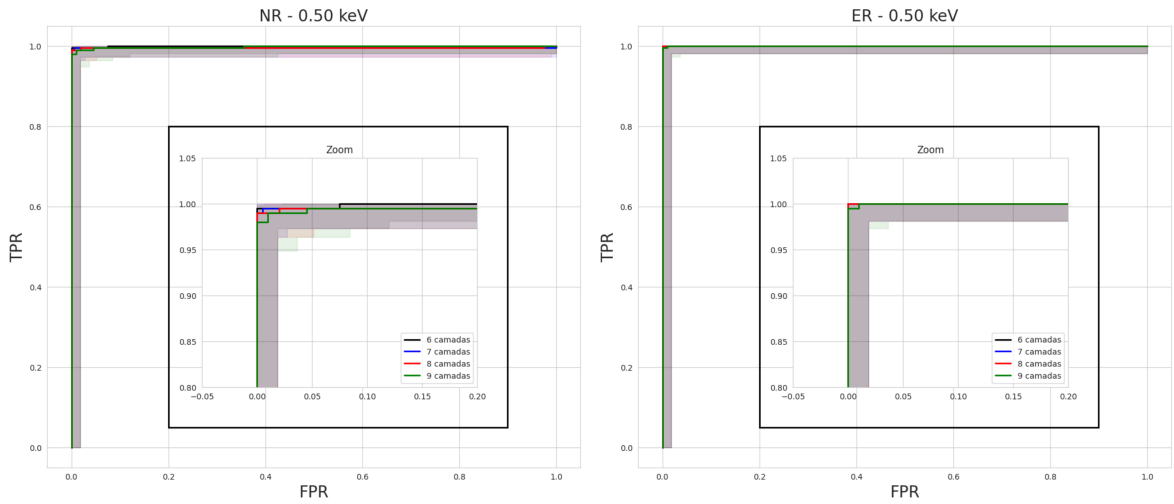
Após a seleção das melhores configurações de CNN, estas foram comparadas novamente no banco de dados de validação, agora utilizando as imagens completas. As Figuras 48 e 49 mostram os resultados para sinais de NR (esquerda) e ER (direita) com 0.25 e 0.5 keV, respectivamente. Observa-se que as quatro configurações apresentaram resultados muito semelhantes, com curvas ROC praticamente perfeitas para 0.5 keV e desempenho superior em comparação com os métodos de filtragem para 0.25 keV.

Figura 48 – Curvas ROC para CNNs com 6, 7, 8 ou 9 camadas de convolução no banco de validação para NR (esquerda) e ER (direita) com 0.25 keV.



Fonte: Elaborada pelo autor (2024).

Figura 49 – Curvas ROC para CNNs com 6, 7, 8 ou 9 camadas de convolução no banco de validação para NR (esquerda) e ER (direita) com 0.50 keV.



Fonte: Elaborada pelo autor (2024).

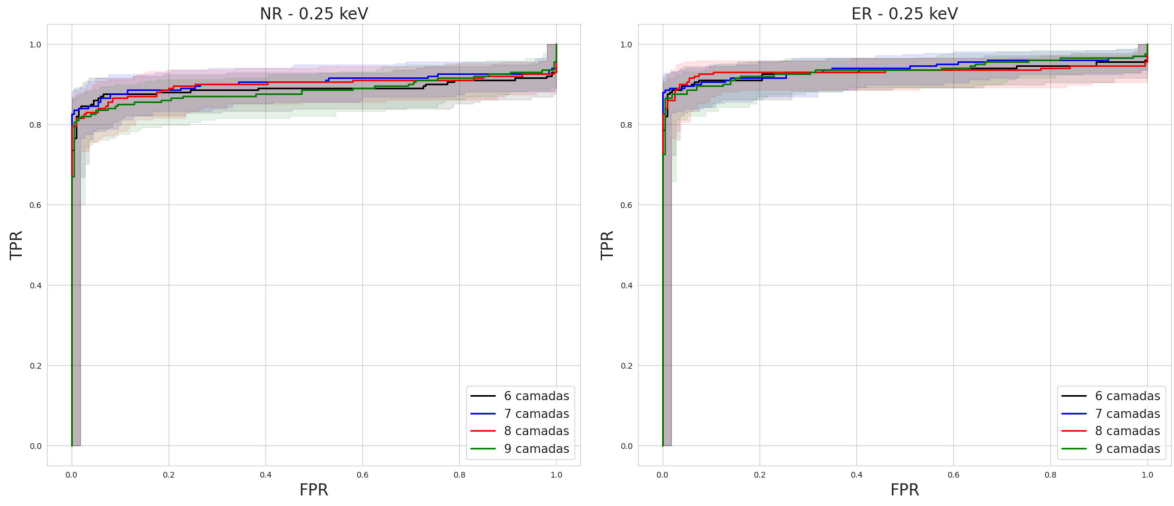
De forma análoga ao método da filtragem, essas configurações foram aplicadas no banco de teste, para verificar a consistência das CNNs em dados não vistos no treinamento.

As Figuras 50 e 51 ilustram as curvas ROC para NR (esquerda) e ER (direita) com 0.25 e 0.5 keV, respectivamente, no banco de testes. A Tabela 5 mostra os valores de AUC para cada uma dessas curvas para os sinais de 0.25 keV, que não apresentaram curvas ROC perfeitas. Nota-se novamente que os resultados foram consistentes na validação e teste, com exceção do ER, que no teste teve uma performance um pouco melhor. Isto indica que o problema de *overfitting* foi mitigado para essas configurações.

Tabela 5 – AUC das curvas ROC das CNNs com 6, 7, 8 e 9 camadas de convolução para NR e ER com 0.25 keV.

Configuração	**NR**		**ER**	
	Validação	**Teste**	**Validação**	**Teste**
6 camadas	$0.9060\pm^{0.0382}_{0.0689}$	$0.8888\pm^{0.0423}_{0.0714}$	$0.8706\pm^{0.0468}_{0.0751}$	$0.9308\pm^{0.0322}_{0.0650}$
7 camadas	$0.9144\pm^{0.0362}_{0.0670}$	$0.9036\pm^{0.0393}_{0.0691}$	$0.8699\pm^{0.0471}_{0.0753}$	$0.9389\pm^{0.0298}_{0.0631}$
8 camadas	$0.9008\pm^{0.0400}_{0.0700}$	$0.8974\pm^{0.0405}_{0.0708}$	$0.8815\pm^{0.0446}_{0.0740}$	$0.9312\pm^{0.0315}_{0.0637}$
9 camadas	$0.8910\pm^{0.0422}_{0.0728}$	$0.8836\pm^{0.0444}_{0.0742}$	$0.8754\pm^{0.0466}_{0.0756}$	$0.9348\pm^{0.0317}_{0.0654}$

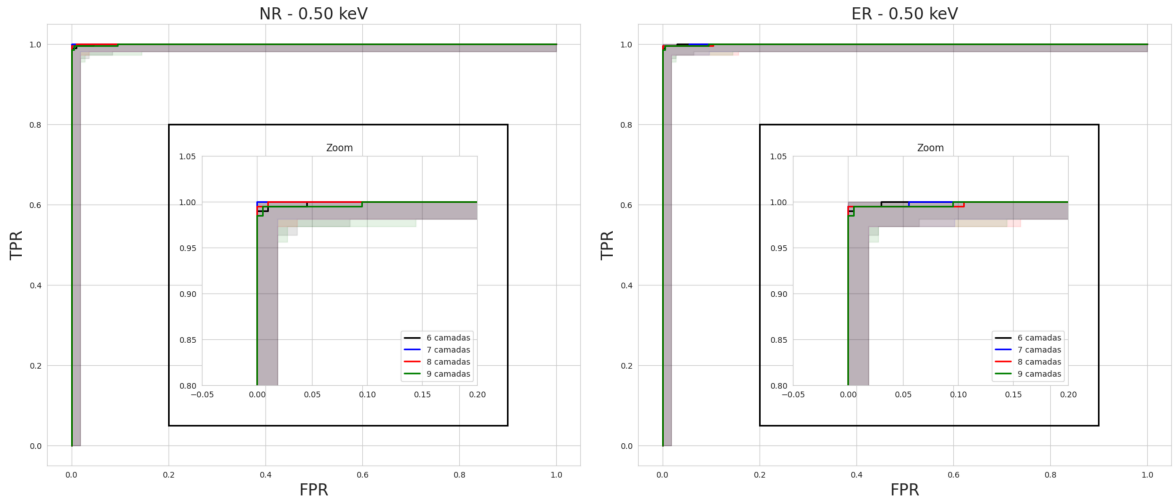
Figura 50 – Curvas ROC para CNNs com 6, 7, 8 ou 9 camadas de convolução no banco de teste para NR (esquerda) e ER (direita) com 0.25 keV.



Fonte: Elaborada pelo autor (2024).

Diferentemente do método de filtragem, cuja saída é ainda relacionada às intensidades dos pixels, a CNN gera uma probabilidade como saída. Dessa forma, pode-se escolher um limiar de forma mais direta, como por exemplo, 50%. Esse é um valor intuitivo e aplicável para o caso da CNN, visto que para as quatro arquiteturas, as distribuições de sinal e ruído ficaram bem separadas, com menos interseção em comparação com a filtragem. As Tabelas 6 e 7 mostram a detecção dos sinais de NR e ER com 0.25 keV e o falso alarme correspondente para cada arquitetura. Com esse limiar, a detecção atingiu quase 80%, um valor próximo ao obtido com filtragem, mas com um falso alarme significativamente menor, variando entre 0 e 3%.

Figura 51 – Curvas ROC para CNNs com 6, 7, 8 ou 9 camadas de convolução no banco de teste para NR (esquerda) e ER (direita) com 0.50 keV.



Fonte: Elaborada pelo autor (2024).

Tabela 6 – Detecção dos sinais de NR e ER com 0.25 keV na validação e teste para CNNs com 6, 7, 8 ou 9 camadas de convolução.

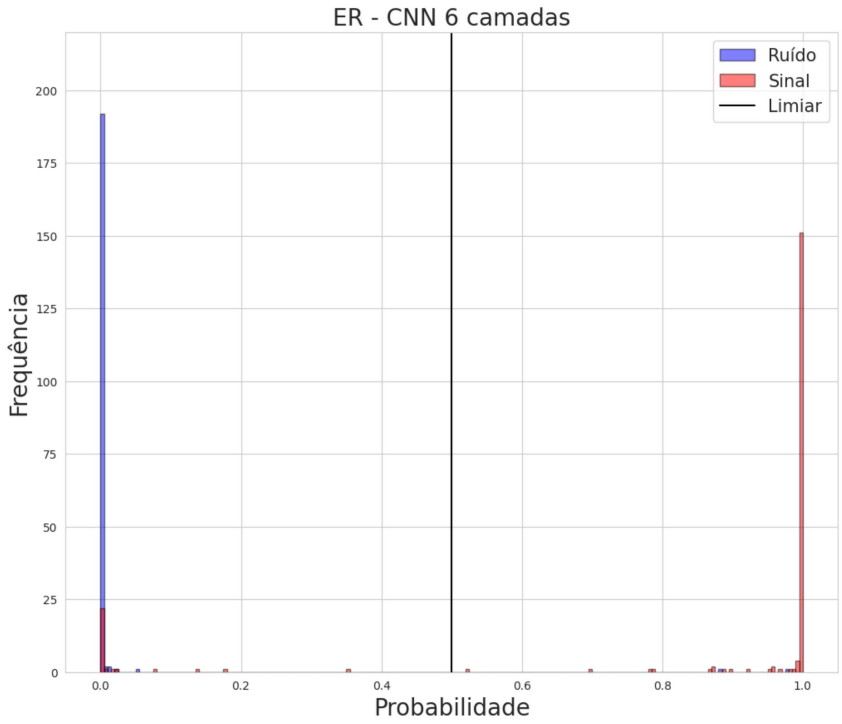
Configuração	**NR**		**ER**	
	Validação	**Teste**	**Validação**	**Teste**
6 camadas	$0.82\pm^{0.0481}_{0.0596}$	$0.7950\pm^{0.0510}_{0.0621}$	$0.76\pm^{0.0555}_{0.0647}$	$0.8550\pm^{0.0431}_{0.0570}$
7 camadas	$0.82\pm^{0.0481}_{0.0596}$	$0.7900\pm^{0.0532}_{0.0621}$	$0.7450\pm^{0.0577}_{0.0647}$	$0.8500\pm^{0.0441}_{0.05700}$
8 camadas	$0.79\pm^{0.0531}_{0.0621}$	$0.7850\pm^{0.0531}_{0.0622}$	$0.7550\pm^{0.0555}_{0.0647}$	$0.8250\pm^{0.0482}_{0.0596}$
9 camadas	$0.80\pm^{0.0505}_{0.0621}$	$0.80\pm^{0.0505}_{0.0621}$	$0.76\pm^{0.0555}_{0.0647}$	$0.8350\pm^{0.0456}_{0.0595}$

Tabela 7 – Falso alarme na validação e teste para CNNs com 6, 7, 8 ou 9 camadas de convolução.

Configuração	**NR**		**ER**	
	Validação	**Teste**	**Validação**	**Teste**
6 camadas	$0.0050\pm^{0.0237}_{0.0047}$	$0.0100\pm^{0.0264}_{0.0082}$	$0.0050\pm^{0.0237}_{0.0047}$	$0.0100\pm^{0.0264}_{0.0082}$
7 camadas	$0.0050\pm^{0.0237}_{0.0047}$	$0.0000\pm^{0.0188}_{0.0000}$	$0.0050\pm^{0.0237}_{0.0047}$	$0.0000\pm^{0.0188}_{0.0000}$
8 camadas	$0.0050\pm^{0.0237}_{0.0047}$	$0.0000\pm^{0.0188}_{0.0000}$	$0.0050\pm^{0.0237}_{0.0047}$	$0.0000\pm^{0.0188}_{0.0000}$
9 camadas	$0.0250\pm^{0.0312}_{0.0151}$	$0.0050\pm^{0.0237}_{0.0047}$	$0.0250\pm^{0.0312}_{0.0151}$	$0.0050\pm^{0.0237}_{0.0047}$

A Figura 52 ilustra as distribuições para sinais de NR com 0.25 keV e ruído, junto do limiar de 50%, para a CNN com 6 camadas de convolução. Os resultados foram similares para as outras três configurações de CNN. A clara separação das distribuições justifica o baixo índice de falso alarme observado, comparado com os métodos de filtragem, que apresentaram maior interseção entre sinal e ruído. Assim, conclui-se que todas as configurações de CNN analisadas têm potencial superior para distinguir sinal de ruído em comparação com os métodos de filtragem. A próxima etapa é avaliar o tempo de processamento de cada método, tópico que será abordado na Seção 5.3.

Figura 52 – Predições de uma CNN com 6 camadas de convolução em imagens com ER de 0.25 keV (vermelho) e apenas ruído eletrônico (azul)



Fonte: Elaborada pelo autor (2024).

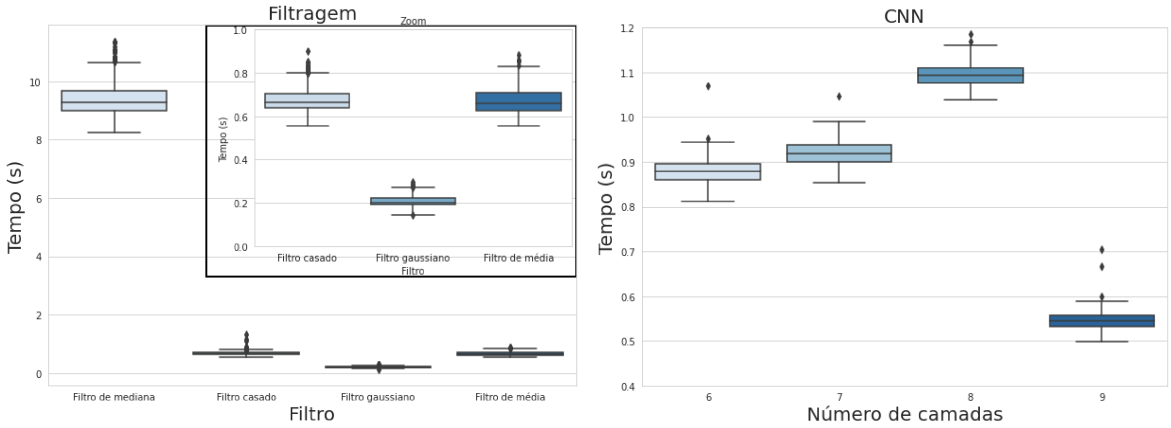
5.3 TEMPO DE PROCESSAMENTO

Conforme discutido nas seções anteriores, os filtros gaussiano e casado se destacaram entre os métodos de filtragem, enquanto as quatro arquiteturas de CNN apresentaram desempenhos próximos e superiores aos dos filtros. No entanto, é fundamental avaliar o tempo de processamento de cada um desses algoritmos, uma variável crucial para um algoritmo de *trigger*. A Figura 53 mostra o tempo de processamento de cada filtro (esquerda) e arquitetura de CNN (direita). Esses tempos foram aferidos com um processador Intel Xeon (Cascadelake) com frequência de 2.2 GHz (89).

O filtro de mediana é significativamente mais lento que os demais, devido à sua natureza não-linear e à impossibilidade de ser aplicado por convoluções. Em contraste, o filtro gaussiano é aproximadamente três vezes mais rápido que o filtro casado e o de média, realizando a filtragem em apenas 0.2 segundos. Isso se deve ao fato de que, sendo um filtro simétrico, há uma função dedicada na biblioteca SciPy (74) que realiza a convolução usando gaussianas 1D, aplicando a convolução primeiro nas linhas e depois nas colunas. Os outros dois filtros utilizam funções baseadas em FFT e IFFT.

No *trigger* baseado em CNN, o número de camadas de convolução normalmente influencia no tempo de processamento, pois um maior número de camadas geralmente aumenta a complexidade da rede. No entanto, uma arquitetura com mais camadas pode

Figura 53 – Tempos de processamento para os métodos baseados em filtragem (esquerda) e CNN (direita) utilizando o processador Intel Xeon.



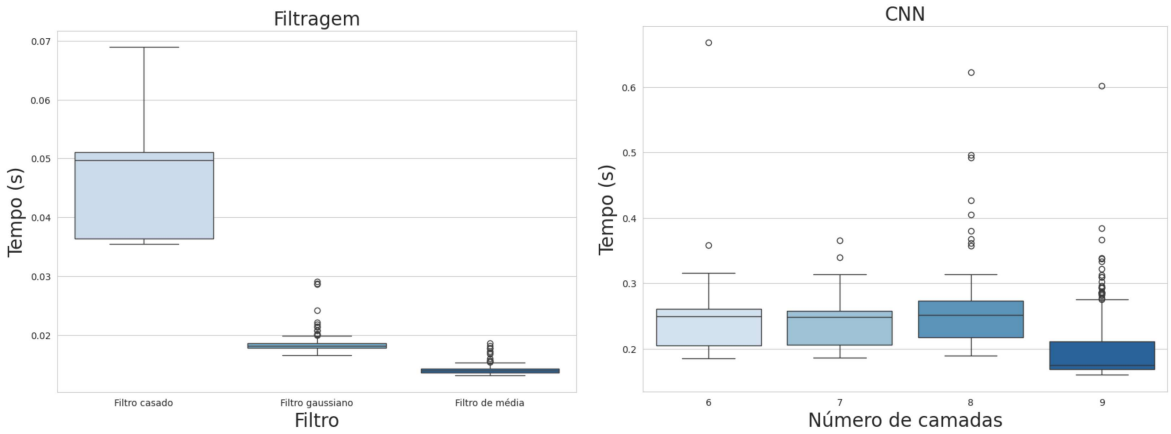
Fonte: Elaborada pelo autor (2024).

ser mais rápida em situações onde o número total de operações realizadas seja menor. Além disso, o uso de unidades de processamento gráfico (GPU) tem contribuído para popularizar esse tipo de método, dado seu poder de processamento superior em comparação com processadores padrão, otimizando ainda mais o desempenho em aplicações com a biblioteca TensorFlow.

A Figura 54 mostra o tempo de processamento desses mesmos algoritmos utilizando a GPU Tesla T4 da NVIDIA (90). Os métodos de filtragem foram ajustados para utilizar a biblioteca TensorFlow e aproveitar ao máximo o uso da GPU. O filtro de mediana continuou a ser significativamente mais lento e foi omitido do gráfico, enquanto os outros filtros mostraram tempos de processamento similares. Os métodos baseados em CNN também experimentaram uma melhoria significativa no tempo de processamento com a GPU, apresentando diferenças mínimas entre as quatro configurações. No entanto, mesmo com a GPU, a filtragem permanece cerca de 10 vezes mais rápida do que qualquer uma das CNNs.

Com uma frequência de aquisição de aproximadamente 1 Hz, qualquer um desses métodos é viável se uma GPU como a Tesla T4 estiver disponível. No entanto, para sistemas que utilizam apenas um processador, como o Intel Xeon, o filtro gaussiano ou a CNN com 9 camadas são as melhores opções, pois outros métodos têm tempos de processamento próximos de 1 segundo. É importante notar que outras etapas do *trigger*, como o carregamento da imagem para classificação, não foram consideradas nesta análise, o que pode afetar o tempo total de processamento.

Figura 54 – Tempos de processamento para os métodos baseados em filtragem (esquerda) e CNN (direita) utilizando a GPU T4.



Fonte: Elaborada pelo autor (2024).

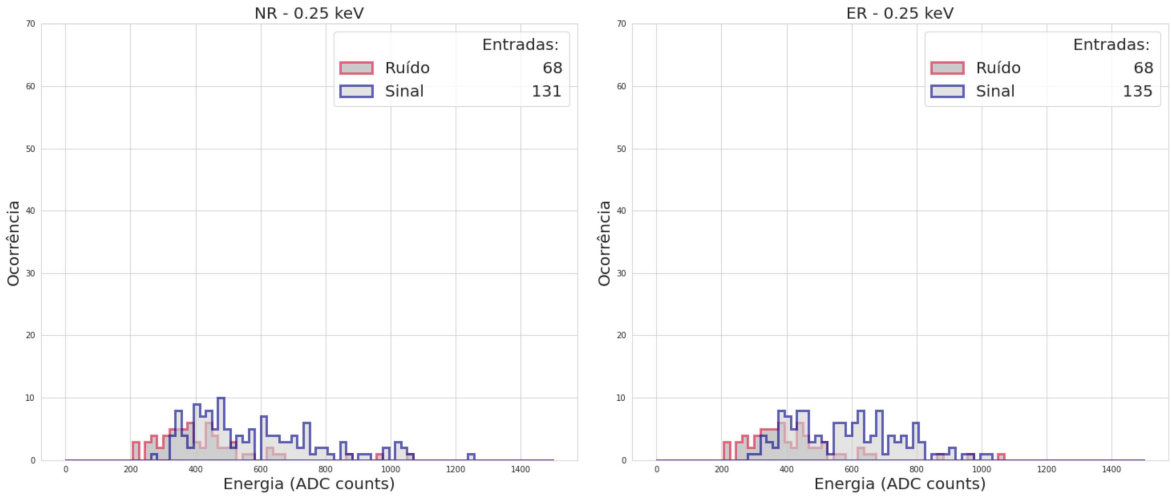
5.4 COMPARATIVO COM A RECONSTRUÇÃO

Os métodos de *trigger* apresentados até aqui demonstram um desempenho quantitativo excelente na detecção de sinais. Contudo, é fundamental comparar esses resultados com o algoritmo de reconstrução utilizado no experimento para análises *offline*. Embora não seja um método de *trigger*, o algoritmo de reconstrução é utilizado para identificar rastros de partículas nas imagens e agrupar os pixels pertencentes a eles em clusters. Assim, pode-se avaliar quantos dos sinais são identificados pelo algoritmo de reconstrução e a taxa de clusters formados por pixels de ruído eletrônico.

As Figuras 55 e 56 apresentam os histogramas de energia dos clusters identificados pela reconstrução no banco de dados de teste para NR (esquerda) e ER (direita) com 0.25 e 0.5 keV, respectivamente. Como se trata de uma simulação, é possível distinguir entre clusters de rastros de sinal e aqueles formados por pixels de ruído eletrônico. Para sinais de 0.25 keV, o algoritmo de reconstrução detectou aproximadamente 65.5% dos sinais de NR e 67.5% dos de ER, enquanto achou 68 clusters de ruído em 55 imagens, resultando em um falso alarme de 27.5%. Em comparação, todos os métodos de *trigger*, nos limiares apresentados anteriormente, conseguiram identificar os eventos detectados pela reconstrução e ainda mais alguns. Para sinais de 0.5 keV, o algoritmo de reconstrução detectou quase todos os sinais, com desempenho similar ao dos métodos de filtragem e CNN. No entanto, os métodos de *trigger* alcançaram essa performance com uma taxa de falso alarme muito menor.

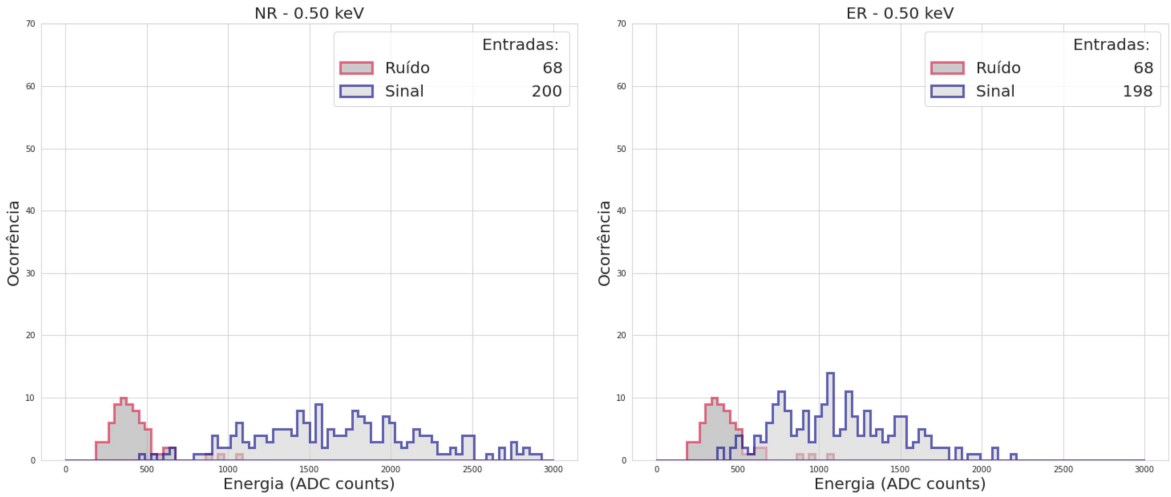
Além das simulações, o algoritmo de reconstrução também foi usado no banco de dados de sinais reais. A Figura 57 ilustra o histograma de energia dos clusters encontrados nessas imagens, com um pico bem definido ao redor 4500 ADCs, originado da fonte radioativa colocada ao lado do detector na aquisição dessas imagens. Por se tratar de

Figura 55 – Histograma de energia dos clusters encontrados no banco de dados de teste de NR (esquerda) e ER (direita) com 0.25 keV.



Fonte: Elaborada pelo autor (2024).

Figura 56 – Histograma de energia dos clusters encontrados no banco de dados de teste de NR (esquerda) e ER (direita) com 0.50 keV.

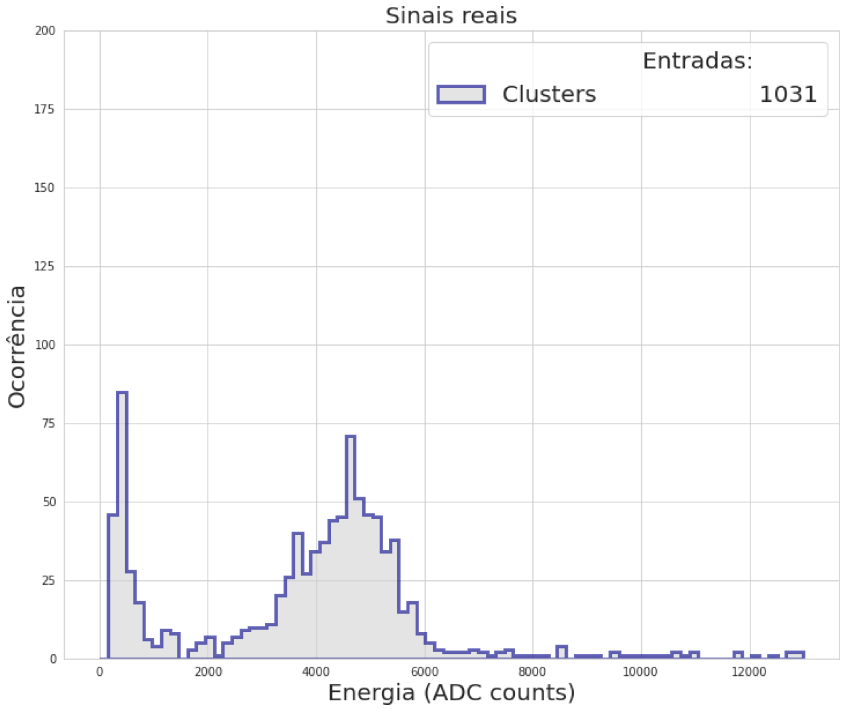


Fonte: Elaborada pelo autor (2024).

imagens reais, não é possível saber com precisão onde estão localizados os eventos de sinal nelas. Entretanto, é possível avaliar o *trigger* com base nos resultados encontrados pela reconstrução, como ilustrado na Figura 58.

A Figura 58 apresenta novamente os histogramas de energia dessas imagens, avaliando se algum pixel pertencente a cada cluster seria detectado pelo algoritmo de *trigger* baseado no filtro gaussiano (esquerda) e na CNN com 9 camadas (direita). O limiar utilizado pelo primeiro foi o que admite até 10% de falso alarme, enquanto o do segundo foi a probabilidade de 50%. Nota-se que apenas clusters com baixa energia não

Figura 57 – Histograma de energia dos clusters encontrados no banco de dados de sinais reais.



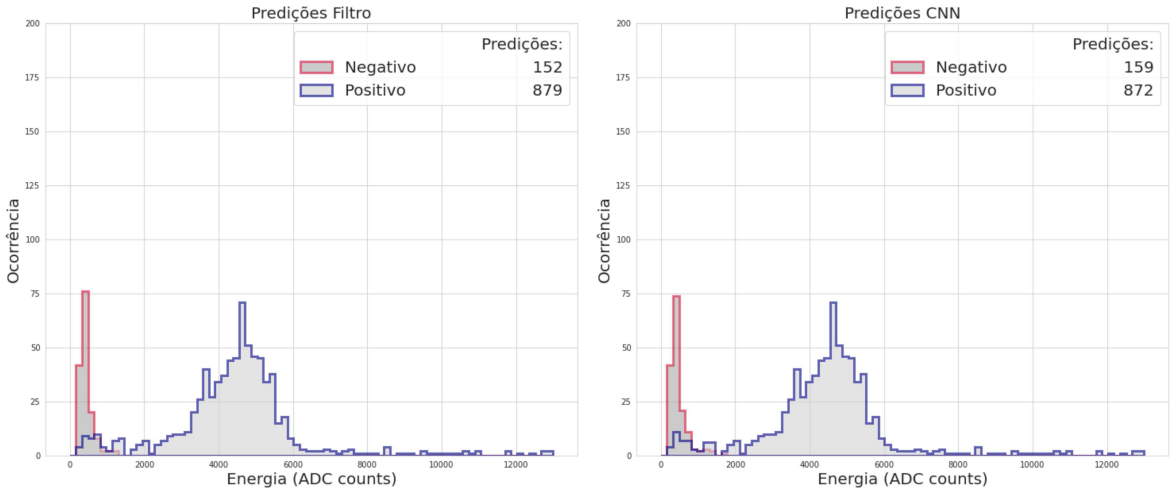
Fonte: Elaborada pelo autor (2024).

foram detectados por esses algoritmos, pois provavelmente são formados por pixels de ruído eletrônico. Em contrapartida, todos os clusters com energia superior a 2000 ADCs foram identificados, mostrando que esses algoritmos também são eficazes na detecção de sinais reais de alta energia. Outro ponto importante é que esse banco de dados de sinais reais foi coletado cerca de 3 meses após as imagens de ruído eletrônico usadas na simulação, indicando que os métodos de *trigger* mantêm sua eficácia sem a necessidade de retreinamento frequente.

Para ilustrar o funcionamento desses métodos de *trigger*, as Figuras 59 e 60 mostram uma imagem do banco de dados de sinais reais e as regiões detectadas pelos algoritmos, respectivamente. A CNN identifica subimagens, o que pode afetar os resultados caso um cluster de ruído esteja na mesma subimagem que um de sinal. Em contraste, o método de filtragem detecta sinais a nível de pixel, como visto nos pixels ativos na imagem da esquerda. Além disso, observa-se um rastro de sinal além dos limites usados pela reconstrução no canto inferior da imagem original, que não foi identificado pelos algoritmos, já que estes também não analisam essa região.

Um aspecto importante a ser considerado também é o tempo de processamento dos algoritmos. A reconstrução é altamente dependente do número de clusters encontrados nas imagens, pois além da formação dos clusters, diversas métricas também são extraídas deles. No banco de dados de sinais reais, cada imagem levou aproximadamente 6 segundos

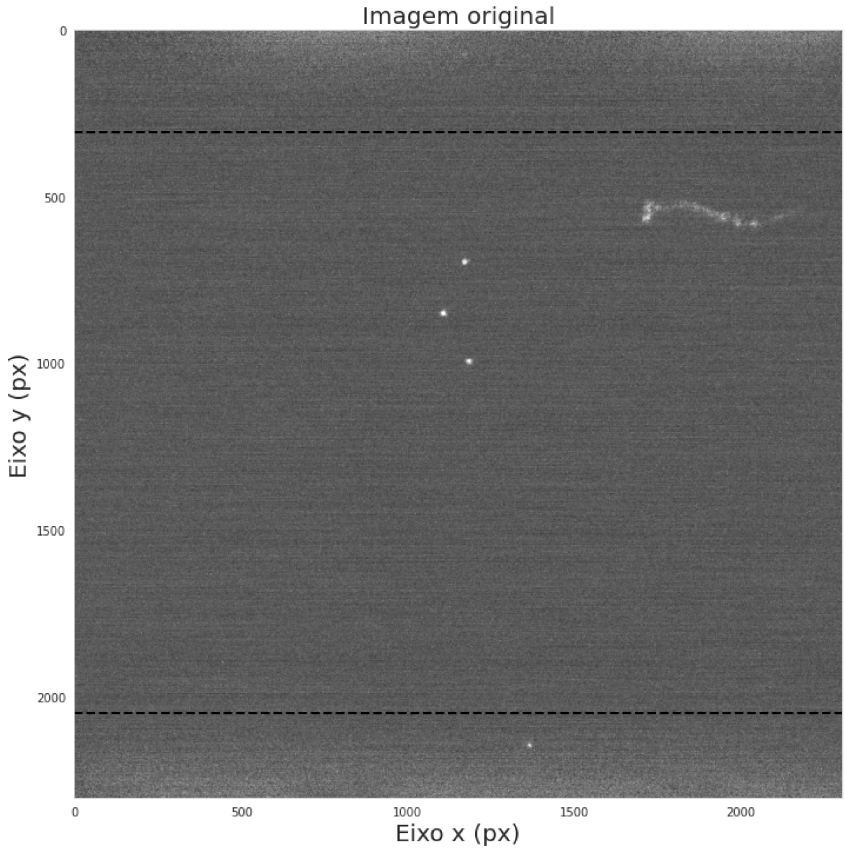
Figura 58 – Histograma de energia dos clusters que foram detectados pelo *trigger* baseado no filtro gaussiano (esquerda) e CNN com 9 camadas (direita).



Fonte: Elaborada pelo autor (2024).

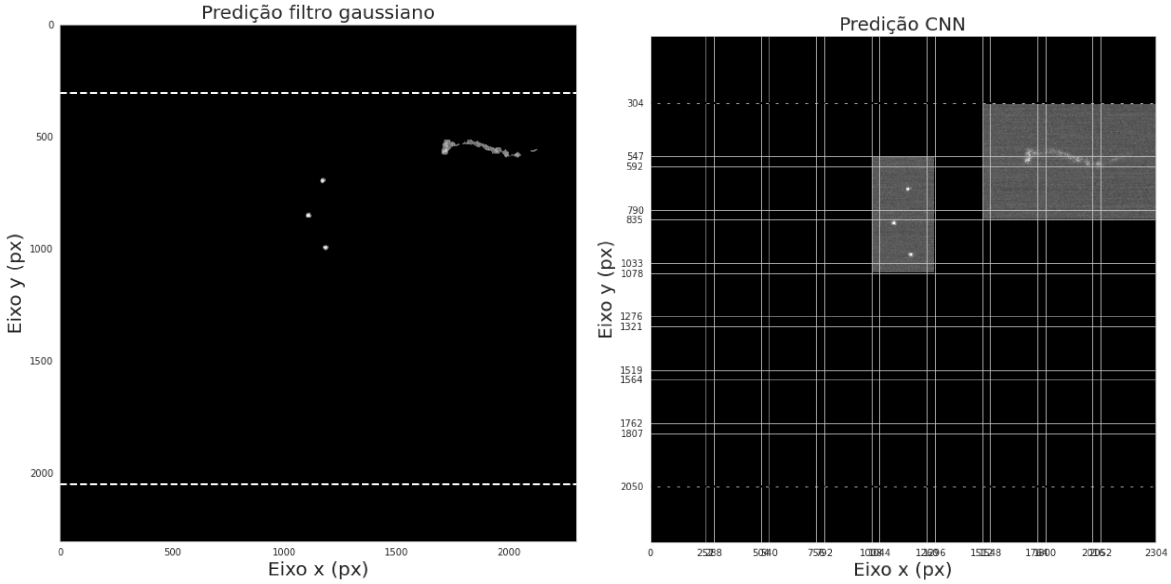
para ser processada pelo algoritmo de reconstrução. Em contraste, os métodos de *trigger* independem da quantidade de rastros presentes na imagem e fazem sua classificação em um tempo igual ao apresentado na Seção 5.3, demonstrando maior eficiência para aplicações que requerem resposta rápida.

Figura 59 – Imagem do banco de dados de sinais reais.



Fonte: Elaborada pelo autor (2024).

Figura 60 – Pixels acima do limiar do filtro gaussiano (esquerda) e subimagens acima do limiar da CNN com 9 camadas (direita) em uma imagem do banco de dados de sinais reais.



Fonte: Elaborada pelo autor (2024).

6 Conclusão

Uma tarefa fundamental para o experimento CYGNO, especialmente na fase de aquisição contínua prevista para a última etapa do projeto, é decidir quais imagens devem ser salvas para uma análise posterior. Neste contexto, este trabalho propôs e avaliou dois algoritmos de *trigger*: um baseado em filtragem e outro utilizando redes neurais convolucionais (CNN). Ambos os métodos demonstraram uma alta performance na detecção de sinal e na rejeição de ruído eletrônico, com resultados consistentes tanto em imagens simuladas quanto em imagens reais.

Os resultados mostraram que, após uma etapa de treinamento, os filtros gaussiano e casado foram capazes de detectar aproximadamente 80% dos sinais simulados de NR e ER com 0.25 keV, com uma taxa de falso alarme de 10%. A aplicação do filtro gaussiano levou cerca de 200 milissegundos utilizando um processador Intel Xeon e 20 milissegundos com uma GPU Tesla T4. Em contraste, as arquiteturas de CNN testadas alcançaram a mesma taxa de detecção de cerca de 80%, mas com uma taxa de falso alarme reduzida para cerca de 1%. No entanto, a detecção com a CNN exigiu aproximadamente 550 milissegundos com o processador e 200 milissegundos com a GPU para a CNN com 9 camadas de convolução. Esses resultados destacam que há um balanço entre a performance de detecção e o tempo de processamento, sendo que melhorias em um aspecto podem exigir concessões no outro.

Por fim, a comparação entre os métodos propostos e o algoritmo de reconstrução do experimento CYGNO mostrou que os algoritmos de *trigger* detectaram todos os sinais identificados pela reconstrução, além de alguns sinais adicionais que a reconstrução não conseguiu captar. Além disso, esses métodos alcançaram uma taxa de falso alarme significativamente menor. Isso indica que, ao adotar os algoritmos de *trigger*, o experimento CYGNO poderá descartar de forma mais eficiente imagens contendo apenas ruído eletrônico, preservando as que possuem sinais relevantes e otimizando os recursos de processamento e armazenamento.

REFERÊNCIAS

1 Rafael C. Gonzalez and Richard E. Woods. *Processamento Digital de Imagens*. Pearson Education, 3rd edition, 2008.

2 M. Marafini et al. Optical readout of a triple-gem detector by means of a cmos sensor. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 824:562–564, 2016. Frontier Detectors for Frontier Physics: Proceedings of the 13th Pisa Meeting on Advanced Detectors.

3 Fernando Domingues Amaro et al. Directional dark matter searches with cygno. *Particles*, 4(3):343–353, 2021.

4 Fernando Domingues Amaro et al. The cygno experiment. *Instruments*, 6(1), 2022.

5 G. S. P. Lopes. Impact of filtering on cygno experiment. Dissertação de mestrado, Federal University of Juiz de Fora, 2022.

6 I. A. Costa. *Optimization of the clustering algorithm of the CYGNO experiment*. Tese de doutorado, Federal University of Juiz de Fora, 2020.

7 G. Mazzitelli et al. 50 litres tpc with scmos-based optical readout for the cygno project. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 1045:167584, 2023.

8 William K. Pratt. *Digital Image Processing*. John Wiley & Sons, 3rd edition, 2001.

9 V. Elamaran et al. Fpga implementation of spatial image filters using xilinx system generator. *Procedia Engineering*, 38:2244–2249, 2012. INTERNATIONAL CONFERENCE ON MODELLING OPTIMIZATION AND COMPUTING.

10 E. R. Davies. *Computer and machine vision theory, algorithms, practicalities*. Academic Press, 2012.

11 Satinderjit Singh. An alternate algorithm for (3x3) median filtering of digital images. *INTERNATIONAL JOURNAL OF COMPUTERS AND TECHNOLOGY*, 1, 05 2012.

12 Jean Baptiste Joseph Fourier. *Théorie analytique de la chaleur*. Firmin Didot, Paris, France, 1822.

13 John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

14 S. Butterworth. On the theory of filter amplifiers. *Wireless Engineer*, 7(6):536–541, 1930.

15 L. Cutrona et al. Optical data processing and filtering systems. *IRE Transactions on Information Theory*, 6(3):386–400, 1960.

16 A.V. Lugt. Signal detection by complex spatial filtering. *IEEE Transactions on Information Theory*, 10(2):139–145, 1964.

17 Adam Kozma and David Lee Kelly. Spatial filtering for detection of signals submerged in noise. *Appl. Opt.*, 4(4):387–392, Apr 1965.

18 Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943. Reprinted in Anderson and Rosenfeld (1988).

19 Bernard Widrow and Marcian E Hoff. Adaptive switching circuits. In *IRE WESCON Convention Record*, volume 4, pages 96–104, 1960. Reprinted in Anderson and Rosenfeld (1988).

20 Frank Rosenblatt. *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*. Spartan Books, 1962.

21 David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. In David E Rumelhart, James L McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, 1986. Reprinted in Anderson and Rosenfeld (1988).

22 Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

23 Yann LeCun et al. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

24 Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, 2016.

25 Yann LeCun et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

26 Hassan Ramchoun et al. Multilayer perceptron: Architecture optimization and training. *International Journal of Interactive Multimedia and Artificial Inteligence*, 4:26–30, 01 2016.

27 Kevin Jarrett et al. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th International Conference on Computer Vision*, pages 2146–2153, 2009.

28 Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, ICML'10, pages 807–814, Madison, WI, USA, 2010. Omnipress.

29 Lawrence K. Saul, Tommi Jaakkola, and Michael I. Jordan. Mean field theory for sigmoid belief networks. *J. Artif. Int. Res.*, 4(1):61–76, mar 1996.

30 Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

31 Nitish Srivastava et al. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

32 Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

33 Steven J. Nowlan and Geoffrey E. Hinton. Simplifying neural networks by soft weight-sharing. *Neural Computation*, 4(4):473–493, 1992.

34 Tom W. B. Kibble. The standard model of particle physics, 2014.

35 A. Boyarsky et al. Sterile neutrino dark matter. *Progress in Particle and Nuclear Physics*, 104:1–45, Jan 2019.

36 Planck Collaboration, Ade, P. A. R., et al. Planck 2015 results - xiii. cosmological parameters. *A&A*, 594:A13, 2016.

37 J. R. Bond, G. Efstathiou, and J. Silk. Massive Neutrinos and the Large Scale Structure of the Universe. *Phys. Rev. Lett.*, 45:1980–1984, 1980.

38 P. SIKIVIE. Dark matter axions. *International Journal of Modern Physics A*, 25(02n03):554–563, Jan 2010.

39 Marc Schumann. Direct detection of wimp dark matter: concepts and status. *Journal of Physics G: Nuclear and Particle Physics*, 46(10):103003, Aug 2019.

40 Mark W. Goodman and Edward Witten. Detectability of certain dark-matter candidates. *Phys. Rev. D*, 31:3059–3063, Jun 1985.

41 E. Aprile et al. Dark matter results from 225 live days of xenon100 data. *Physical Review Letters*, 109(18), Nov 2012.

42 Paolo Gondolo and Moqbil Alenazi. Directional recoil rates for wimp direct detection. *Physical Review D*, 77, 02 2008.

43 R. Bernabei et al. First results from dama/libra and the combined results with dama/nai. *The European Physical Journal C*, 56(3):333–355, Aug 2008.

44 C. E. Aalseth et al. Search for an annual modulation in three years of cogent dark matter detector data, 2014.

45 Francis Froborg and Alan R Duffy. Annual modulation in direct dark matter searches. *Journal of Physics G: Nuclear and Particle Physics*, 47(9):094002, jul 2020.

46 Jay N. Marx and David R. Nygren. The Time Projection Chamber. *Phys. Today*, 31N10:46–53, 1978.

47 Fabio Sauli. Micro-pattern gas detectors. *Nuclear Instruments and Methods in Physics Research Section A Accelerators Spectrometers Detectors and Associated Equipment*, 477, 01 2002.

48 F. Sauli. Principles of operation of multiwire proportional and drift chambers. *CERN-77-09*, 5 1977.

49 LCTPC Collaboration. Working principle of a tpc. Disponível em: `https://www.lctpc.org/e8/e57671`. Acesso em: 12 de julho de 2024, 2024.

50 A. Oed. Position-sensitive detector with microstrip anode for electron multiplication with gases. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 263(2):351–359, 1988.

51 F. Sauli. Gem: A new concept for electron amplification in gas detectors. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 386(2):531–534, 1997.

52 S. Lami et al. A triple-gem telescope for the totem experiment. *Nuclear Physics B - Proceedings Supplements*, 172:231–233, Oct 2007.

53 C. Altunbas et al. Construction, test and commissioning of the triple-gem tracking detector for compass. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 490(1):177–203, 2002.

54 Laura Baudis. Direct dark matter detection: The next decade. *Physics of the Dark Universe*, 1(1-2):94–108, Nov 2012.

55 Peter Schneider. *Extragalactic Astronomy and Cosmology.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

56 Stefan Knirck et al. Directional axion detection. *Journal of Cosmology and Astroparticle Physics*, 2018(11):051–051, Nov 2018.

57 Ciaran A.J. O'Hare. Can we overcome the neutrino floor at high masses? *Physical Review D*, 102(6), Sep 2020.

58 E. Baracchini et al. Negative ion time projection chamber operation with SF6at nearly atmospheric pressure. *Journal of Instrumentation*, 13(04):P04022–P04022, apr 2018.

59 Michela Marafini et al. **High granularity tracker based on a Triple-GEM optically read by a CMOS-based camera**. *JINST*, 10(12):P12010, 2015.

60 D. Pinci et al. High resolution TPC based on optically readout GEM. *Nucl. Instrum. Meth. A*, 936:453–455, 2019.

61 Davide Pinci et al. Cygnus: development of a high resolution tpc for rare events. In *Proceedings of The European Physical Society Conference on High Energy Physics — PoS(EPS-HEP2017)*, page 077, 10 2017.

62 Fernando Domingues Amaro et al. A 50 l cygno prototype overground characterization. *The European Physical Journal C*, 83(10):946, 2023.

63 Hamamatsu. *Photomultiplier Tube R7378*, 2 1998.

64 Hamamatsu. *ORCA-fusion*, 9 2022.

65 S. Agostinelli et al. Geant4—a simulation toolkit. *Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250–303, 2003.

66 S. INCERTI et al. The geant4-dna project. *International Journal of Modeling, Simulation, and Scientific Computing*, 01(02):157–178, 2010.

67 John Smith and Jane Doe. Violin plots were employed to compare the performance of various machine learning models, providing insight into the variability and central tendency of their performance metrics. *IEEE Transactions on Neural Networks and Learning Systems*, 31(10):1234–1245, 2020.

68 F D Amaro et al. Directional idbscan to detect cosmic-ray tracks for the cygno experiment. *Measurement Science and Technology*, 34(12):125024, sep 2023.

69 Martin Ester et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

70 Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2nd edition, 2009.

71 Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media, Inc., 2nd edition, 2019.

72 Ted Dunning and Ellen Friedman. *Practical Machine Learning: Innovations in Recommendation.* O'Reilly Media, Inc., 2014.

73 Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual.* CreateSpace, Scotts Valley, CA, 2009.

74 Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

75 Olga Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

76 Alex Krizhevsky et al. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

77 Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

78 Christian Szegedy et al. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

79 Kaiming He et al. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

80 Gao Huang et al. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

81 Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, July 2019.

82 Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *ArXiv*, abs/1012.2599, 2010.

83 Mohammad Masum et al. Bayesian hyperparameter optimization for deep neural network-based network intrusion detection. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 5413–5419, 2021.

84 Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

85 Alexandre Défossez, Leon Bottou, Francis Bach, and Nicolas Usunier. A simple convergence proof of adam and adagrad. *Transactions on Machine Learning Research*, 2022.

86 Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. ROC Analysis in Pattern Recognition.

87 Skipper Seabold and Josef Perktold. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010.

88 Lawrence D. Brown, T. Tony Cai, and Anirban DasGupta. Interval Estimation for a Binomial Proportion. *Statistical Science*, 16(2):101 – 133, 2001.

89 Intel Corporation. *Intel Xeon Processor (Cascadelake) Datasheet, Volume 1*, 2019. `https://www.intel.com/content/www/us/en/products/docs/processors/xeon/cascade-lake-datasheet-vol-1.html`.

90 NVIDIA Corporation. *NVIDIA Tesla T4 GPU Architecture*, 2018. Accessed: 2024-08-04.

Lista de Publicações

1. Publicação em revista

   Amaro, F.D. et al. (2023). **Directional iDBSCAN to detect cosmic-ray tracks for the CYGNO experiment**. *Measurement Science and Technology*, **34**(12), 125024. DOI: 10.1088/1361-6501/acf402.

   The CYGNO experiment aims to study rare events related to the search for low-mass dark matter and solar neutrino events. One of the main components of background comes from cosmic rays that generate long tracks in the detector's images. The interaction of such particles with the gas releases a variable energy profile along its trajectory to form tracks with multiple cores that can be easily reconstructed erroneously by being split into more than one cluster. Thus, this work offers a newly adapted version of the well-known density-based spatial clustering of applications with noise (DBSCAN) algorithm, called iDDBSCAN, which exploits the directional characteristics of the clusters found by the DBSCAN to improve its clustering efficiency when dealing with multi-core tracks. This paper provides a detailed explanation of this algorithm, covering its parameter validation and evaluating its influence when integrated into the experiment's event selection routine. To generate background events, data acquisition was performed with the detector installed in an overground laboratory, leaving it exposed to natural radiation. To produce signals in the energy range of interest for the experiment, a 55Fe radioactive source was used. The achieved results showed that the iDDBSCAN algorithm is capable of improving the background rejection of the experiment, through a more accurate reconstruction of the tracks produced by natural radiation such as cosmic rays, without deteriorating its signal detection efficiency and energy estimation.

2. Publicação em revista

   Almeida, B.D. et al. (2023). **Noise assessment of CMOS active pixel sensors for the CYGNO Experiment**. *Measurement Science and Technology*, **34**(12), 125145. DOI: 10.1088/1361-6501/acf7e1.

   Active Pixel sensors play a crucial role in enabling successful low-light scientific experiments due to their inherent advantages and capabilities. Such devices not only offer high spatial resolution but also feature individual pixels with integrated amplifiers, allowing for direct signal amplification at the pixel level. This results in reduced readout noise and improved signal-to-noise ratio (SNR), which are particularly vital when dealing with limited photon counts in low-light environments. This holds particularly true for scientific CMOS (sCMOS) sensors, acknowledged as an advanced evolution of Active Pixel sensors. However, despite their advantages, such sensors can still exhibit limitations such as higher cost and presence of noise

artifacts that should be closely investigated. In particular, CYGNO project fits in a global effort aimed at direct detection of Dark Matter particles. CYGNO collaboration intends to build a detector based on a Time Projection Chamber making use of Gas Electron Multipliers for the amplification of ionization electrons. The GEM multiplication process produces photons that can be readout by a high-resolution sCMOS sensor. Such detection system is being designed to have enough sensitivity to detect low-energy particles and to measure released energy with enough granularity so to reconstruct direction and energy profile along their trajectories. The image sensor has an important role in the detector performance, having a direct impact on the SNR of the experiment. This work proposes a study on the performance of three different sCMOS sensors with respect to their sensitivity to low-energy particles and their intrinsic noise, which are of the utmost importance for various scientific experiments.