UNIVERSIDADE ESTADUAL PAULISTA
"JULIO DE MESQUITA FILHO"
INSTITUTO DE GEOCIÊNCIAS E CIÊNCIAS EXATAS

Trabalho de Conclusão de Curso
Curso de Graduação em Física

Uso de Redes Neurais para Identificação do Decaimento do Bóson W

Gabriel Oliveira Dos Santos

Prof. Dr. Luiz Antônio Barreiro
Prof. Dr. Thiago Rafael Fernandez Perez Tomei

Rio Claro-SP
2024

UNIVERSIDADE ESTADUAL PAULISTA
Instituto de Geociências e Ciências Exatas
Câmpus de Rio Claro

Gabriel Oliveira Dos Santos

Uso de Redes Neurais para Identificação do Decaimento do Bóson W

Trabalho de Conclusão de Curso apresentado ao Instituto de Geociências e Ciências Exatas - Câmpus de Rio Claro, da Universidade Estadual Paulista Júlio de Mesquita Filho, para obtenção do grau de Bacharel em Física.

Rio Claro - SP
2024

S237u

Santos, Gabriel Oliveira dos
Uso de Redes Neurais para Identificação do Decaimento do Bóson W / Gabriel Oliveira dos Santos. -- Rio Claro, 2024
62 p.

Trabalho de conclusão de curso (Bacharelado - Física) - Universidade Estadual Paulista (UNESP), Instituto de Geociências e Ciências Exatas, Rio Claro
Orientador: Luiz Antonio Barreiro
Coorientador: Thiago Rafael Fernandez Perez Tomei

1. Física de Altas Energias. 2. Identificação de Partículas. 3. Machine Learning. 4. Colisões Hadrônicas. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

Gabriel Oliveira Dos Santos

Uso de Redes Neurais para Identificação do Decaimento do Bóson W

Trabalho de Conclusão de Curso apresentado ao Instituto de Geociências e Ciências Exatas - Câmpus de Rio Claro, da Universidade Estadual Paulista Júlio de Mesquita Filho, para obtenção do grau de Bacharel em Física.

Comissão Examinadora

Prof. Dr. Luiz Antonio Barreiro (orientador)

Prof. Dr. Thiago Rafael Fernandez Perez Tomei (coorientador)

Prof. Dr. Caetano Mazzoni Ranieri

Rio Claro, 14 de Novembro de 2024.

Assinatura do(a) aluno(a)

assinatura do(a) orientador(a)

Dedico este trabalho a Rex e Luli, cujas memórias e lealdade me deram força e conforto durante os desafios da pandemia de COVID-19. E ao Scooby, seu filho, cuja presença iluminou meus dias e trouxe uma alegria constante à minha vida.

Agradecimentos

Gostaria de expressar minha profunda gratidão a algumas pessoas e instituições que foram fundamentais na minha jornada acadêmica.

Primeiramente, agradeço aos meus pais e à minha família pelo apoio incondicional e pela motivação constante para seguir meu sonho de construir uma carreira científica.

Um agradecimento especial à Professora Fabíola, do meu ensino médio, cujo incentivo foi fundamental para minha decisão de ingressar na UNESP.

Sou extremamente grato ao meu orientador de iniciação científica, Thiago Rafael Fernandez Perez Tomei. Seu trabalho e orientação foram essenciais para a integração das áreas que eu tanto gosto, física e computação, permitindo-me aprimorar meu conhecimento e habilidades em ambas.

Agradeço também ao meu orientador de TCC, Luiz Antonio Barreiro, por me apresentar o universo da física de partículas.

Sou eternamente grato, especialmente aos meus amigos Isabella, Lucas, Leonardo e Ana Júlia, pelo apoio e pela companhia ao longo dessa jornada. As horas de estudo e diversão que compartilhamos são memórias que levarei para sempre comigo.

Esta pesquisa tornou-se possível graças aos recursos computacionais disponibilizados pelo Núcleo de Computação Científica (NCC/GridUNESP) da Universidade Estadual Paulista (UNESP).

O presente projeto foi desenvolvido com apoio da Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) através da concessão de bolsa de estudo de Iniciação Cientifica - Código de Financiamento 121762/2023-8

Por fim, agradeço à UNESP pela excelência no ensino público e de qualidade, que me proporcionou uma formação sólida e me preparou para o futuro.

"It doesn't matter how beautiful your theory is, it doesn't matter how smart you are. If it doesn't agree with experiment, it's wrong."

Richard P. Feynman

Resumo

Este trabalho estuda o uso de redes neurais artificiais para identificar o decaimento do Bóson W em experimentos de física de partículas, como o *Compact Muon Solenoid* (CMS) no *Large Hadron Collider* (LHC). O Bóson W é uma partícula mediadora da interação fraca, responsável por processos fundamentais como o decaimento de nêutrons e a troca de sabor entre quarks. Inicialmente, o estudo aborda conceitos do Modelo Padrão da física de partículas, descrevendo as interações entre partículas elementares e destacando a importância do Bóson W. O detector CMS é detalhado com foco nos sistemas de detecção de partículas, como os calorímetros eletromagnético e hadrônico, essenciais para medir a energia das partículas resultantes das colisões. O trabalho destaca o fenômeno de "jatos gordos", que ocorrem quando partículas altamente energéticas, decaem em pares de outras partículas que produzem jatos quase sobrepostos. Esses jatos são difíceis de distinguir por métodos tradicionais devido à proximidade das partículas geradas. A identificação correta desses jatos gordos é crucial para diferenciar eventos de decaimento do Bóson W de outros processos de fundo. Para identificar o Bóson W e seus jatos gordos, utilizou-se uma rede neural *multilayer perceptron* (MLP), que processa grandes volumes de dados experimentais, identificando padrões complexos. A rede foi treinada com dados simulados, focando na distinção entre eventos de decaimento do Bóson W e outros processos semelhantes. O tratamento de dados incluiu a seleção de características como a energia transversal e a pseudo-rapidez das partículas. A rede MLP foi estruturada com várias camadas ocultas, otimizadas para aumentar a precisão da identificação. Funções de ativação como ReLU foram usadas para melhorar o desempenho durante o treinamento, e algoritmos de otimização, como SGD com momento e ADAM, foram testados. A avaliação do desempenho foi feita através de curvas *receiver operating characteristic* (ROC), que mostraram a capacidade do modelo em reduzir falsos positivos e distinguir corretamente os decaimentos do Bóson W. Conclui-se que a aplicação de machine learning, especialmente redes neurais, é uma abordagem promissora na análise de dados do CMS, permitindo a identificação precisa de jatos gordos e eventos específicos de decaimento de partículas fundamentais. Este avanço destaca o potencial das técnicas de aprendizado de máquina em experimentos de alta energia, auxiliando no progresso da pesquisa em física de partículas.

Palavras-chave: Física de Altas Energias, Machine Learning, Identificação de Partículas, Bóson W, Colisões Hadrônicas

Abstract

This work studies the use of artificial neural networks to identify the decay of the W boson in particle physics experiments, such as the Compact Muon Solenoid (CMS) at the Large Hadron Collider (LHC). The W boson is a mediator of the weak interaction, responsible for fundamental processes such as neutron decay and flavor exchange between quarks. Initially, the study addresses concepts from the Standard Model of particle physics, describing interactions between elementary particles and highlighting the importance of the W boson. The CMS detector is detailed with a focus on particle detection systems, such as the electromagnetic and hadronic calorimeters, which are essential for measuring the energy of particles resulting from collisions. The work emphasizes the phenomenon of "fat jets,"which occur when highly energetic particles decay into pairs of other particles that produce nearly overlapping jets. These jets are difficult to distinguish using traditional methods due to the proximity of the generated particles. Correctly identifying these fat jets is crucial for differentiating W boson decay events from other background processes. To identify the W boson and its fat jets, a multilayer perceptron neural network (MLP) was employed, capable of processing large volumes of experimental data and identifying complex patterns. The network was trained with simulated data, focusing on distinguishing W boson decay events from other similar processes. Data processing included feature selection such as transverse energy and pseudo-rapidity of the particles. The MLP was structured with several hidden layers optimized to enhance identification accuracy. Activation functions like ReLU were used to improve performance during training, and optimization algorithms such as SGD with momentum and ADAM were tested. Performance evaluation was conducted through receiver operating characteristic (ROC) curves, which demonstrated the model's ability to reduce false positives and correctly distinguish W boson decays. It is concluded that the application of machine learning, particularly neural networks, is a promising approach in the analysis of CMS data, enabling precise identification of fat jets and specific events of fundamental particle decays. This advancement highlights the potential of machine learning techniques in high-energy experiments, contributing to the progress of research in particle physics.

Keywords: High-Energy Physics, Machine Learning, Particle Identification, W Boson, Hadronic Collisions

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

A física de altas energias, também conhecida como física de partículas, é uma das áreas mais avançadas da ciência moderna. Ela busca entender os constituintes fundamentais da matéria e das forças que governam suas interações. Além disso, explora os limites extremos de energia e matéria, recriando condições na existência do Big Bang. Para o estudo de partículas subatômicas, como os quarks, léptons e Bósons, muitas vezes se utiliza de grandes experimentos, como o *Large Hadron Collider* (LHC) do *Conseil européen pour la recherche nucléaire* (CERN), na Suíça. Por meio da colaboração internacional de físicos experimentais e teóricos, em 2012 foi confirmado o Bóson de Higgs, que é o fornecedor de massa para as outras partículas fundamentais.

No século XIX, o átomo era considerado a menor partícula do universo, similar a uma pequena esfera indivisível. No entanto, em 1897, J.J. Thomson, utilizando o experimento do tubo de raios catódicos, observou e concluiu a existência da primeira partícula elementar descoberta: o elétron, que é menor que o átomo e possui carga negativa. Com essa descoberta, Thomson propôs o modelo do "pudim de passas", no qual o átomo seria composto por uma esfera de carga positiva com elétrons incrustados em sua superfície (ver figura 1.1).

Em 1909, sob a orientação de Rutherford, Marsden realizou o experimento da folha de ouro, que consistia em lançar radiação alfa[1] sobre uma fina folha de ouro. Observou-se que muitas partículas alfa atravessavam a folha, mas algumas eram refletidas ou desviadas. Assim, em 1911, concluiu-se que o átomo possuía uma carga positiva concentrada em um núcleo, com elétrons de carga negativa ao redor, conforme ilustrado na figura 1.1. Também foi observado que quase toda a massa do átomo estava concentrada no núcleo, inicialmente, essa proposta enfrentou resistência, pois conflitava com a mecânica clássica. Segundo as leis do eletromagnetismo, um elétron em movimento ao redor do núcleo deveria espiralar para dentro, colidindo com a carga positiva. Essa incongruência indicava uma falha no modelo atômico da época.

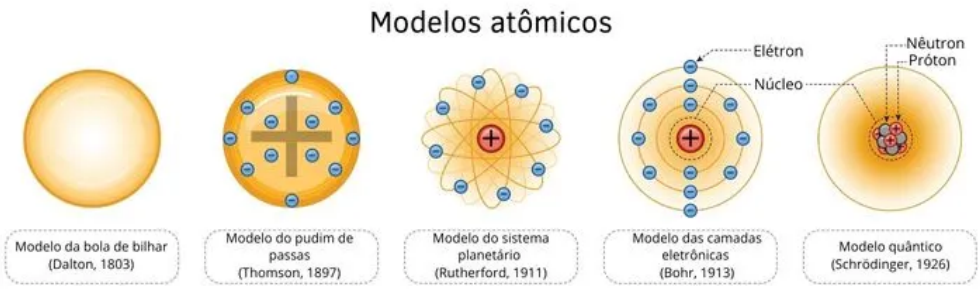

Figura 1.1: Ilustração dos modelos atômicos com passar dos anos, sendo o mais a esquerda o mais antigo, modelo de Dalton, e o ultimo da direita o mais moderno, o modelo de Schrödinger

[1]Radiação alfa consiste em núcleos de hélio

Niels Bohr, utilizando as ferramentas dos primórdios da mecânica quântica, como a quantização de energia de Max Planck e Albert Einstein, aprimorou o modelo de Rutherford. Em 1913, ele propôs o modelo conhecido como Rutherford-Bohr, que introduz níveis de energia para explicar a distribuição eletrônica. A eletrosfera é formada por camadas, sendo que as camadas mais distantes do núcleo possuem mais energia. Bohr também propôs que, quando um elétron recebe um quantum[2], ele salta para uma camada mais externa, liberando um fóton ao retornar à sua camada inicial.

Já em 1926, um modelo quântico começou a ser desenvolvido, abordando o comportamento dos elétrons e a delimitação da eletrosfera. De acordo com o princípio da incerteza de Heisenberg, é impossível determinar simultaneamente a posição e a velocidade de uma partícula. Assim, não é possível localizar exatamente os elétrons no átomo. No entanto, por meio da equação de Schrödinger, foi possível determinar a região de maior probabilidade de encontrar um elétron. Dessa forma, a eletrosfera passou a ser considerada como uma nuvem de elétrons probabilística.

Com o tempo, descobriu-se que o núcleo não é uma entidade indivisível, mas sim composto por prótons e nêutrons, que por sua vez são constituídos de quarks. No espectro de novas partículas elementares, foi encontrado o múon, uma espécie de elétron de maior massa, que não interage pela força forte, sendo apropriado para ser utilizado em detectores .A identificação e estudo de quarks e outras partículas subatômicas foi possível graças a uma série de experimentos avançados e teorias que contrapunham os modelos existentes. Esses avanços não apenas redefiniram o entendimento da estrutura da matéria, mas também evidenciaram a necessidade de novas abordagens analíticas para lidar com a complexidade crescente dos dados experimentais.

Hoje, o *Machine Learning* (ML) se tornou uma ferramenta importante na análise de grandes volumes de dados, especialmente na física de altas energias. Esse campo lida com dados experimentais complexos. A capacidade dos algoritmos de aprendizado de máquina de identificar padrões, e fazer previsões com base em dados extensivos vai além das capacidades humanas tradicionais de análise. Essa evolução tecnológica destaca como a interseção entre ciência da computação e a física moderna está moldando o futuro da pesquisa em física de altas energias e outros campos da ciência de fronteira.

Neste trabalho, serão abordados os seguintes temas: na seção 2, uma abordagem do experimento CMS, incluindo detalhes sobre o detector, o funcionamento dos calorímetros, que medem a energia das partículas, espectrômetros que medem o momento das partículas carregadas, o sistema de aquisição de dados e os *triggers*. Já na seção 3, apresentaremos uma abordagem mais teórica, explicando o Modelo Padrão, a interação das forças: forte, eletromagnética, fraca. E jatos de partículas. Em seguida, na seção 4, continuaremos com uma abordagem teórica, desta vez focada em ML, abordando conceitos como redes neurais artificiais, o processo de aprendizagem dos modelos, tipos de funções de ativação e otimização de modelos. Na seção 5, será detalhada a aplicação das técnicas de Machine

[2]Energia externa, como calor, luz ou eletricidade

Learning, incluindo a metodologia do trabalho e a discussão dos resultados. Por fim, a seção 6 trará a conclusão do estudo.

2 Compact Muon Solenoid

O *Compact Muon Solenoid* (CMS), está localizado no CERN [9], situado na fronteira entre a França e Suíça. Foi criado com o objetivo de promover uma colaboração entre os países europeus na área de pesquisa, visando o domínio da física de altas energias. Atualmente, cerca de 17 mil cientistas ao redor do mundo trabalham por meio de institutos conveniados, com 113 estados membros, sendo o Brasil o primeiro da América [10].

No CERN, são produzidas diversas partículas por meio de aceleradores, começando com a criação do acelerador linear *Linac1* [11] em 1959, projetado para acelerar prótons de baixas energias, cerca de 50 MeV. O primeiro colisor de prótons foi criado em 1971, o *Intersecting Storage Rings* (ISR) [12], com uma energia de 56 GeV e um diâmetro de 300 metros. Após esses aceleradores menores, foram construídos o *proton Synchrotron* (PS) [13], com 200 metros de diâmetro e energia de 25 GeV, o *super proton Synchrotron* (SPS) [14], com um diâmetro de 2,2 km, a 40 metros abaixo da superfície, e energia de 450 GeV. Os prótons não eram inicialmente acelerados nele, mas passavam primeiramente pelo PS. Com a utilização do *Antiproton Accumulator* (AA) em 1981, foi possível criar cerca de 10^{12} antipartículas de prótons [15]. Após alcançar a energia de 450 GeV, prótons e antiprótons se chocavam, vindos de direções diferentes. Com esse experimento, em 1983, foi possível observar a existência dos mediadores da força fraca [3]. Neste mesmo ano, começou a construção do *Large Electron Positron* (LEP) [16], exigindo a criação de um túnel a 100 metros abaixo da superfície, com um diâmetro de 27 km e 4 metros de altura, e energia de 200 GeV. A construção do LEP contribuiu significativamente para a teoria eletrofraca. Ao contrário dos colisores anteriores, o LEP foi construído para aceleração de elétrons (e^-) e pósitrons (e^+). Graças ao experimento, foi possível observar e obter a massa dos Bósons W e Z.

Com o objetivo de detectar o Bóson de Higgs, foi criado o maior e mais potente colisor do mundo, o LHC [17], utilizando o mesmo túnel criado para o LEP. Inaugurado em 2008, e projetado para obter uma energia total de 14 TeV, inicialmente se obteve uma energia total de 7 TeV, mas na terceira tomada de dados (*Run* 3) foi possível obter uma energia total de 13,6 TeV. Seu funcionamento se da pela aceleração de dois feixes de prótons em sentidos opostos, obtidos pela ionização do átomo de hidrogênio, ou por íons de chumbo.

Para compreender a vantagem de utilizar a aceleração de dois feixes em sentidos opostos, é necessário revisar alguns conceitos da cinemática relativística e os sistemas de referência envolvidos, como o sistema de centro de massa (CM) e o sistema laboratório (lab). Neste trabalho, foi adotado o uso de unidades naturais, ou seja, $c = \hbar = 1$.

No sistema de centro de massa (CM), o momento total das partículas é zero:

$$\vec{p}_a + \vec{p}_b = 0 \tag{2.1}$$

[3] Bóson de *gauge*

Em contraste, no sistema laboratório, frequentemente utilizado em experimentos com alvo fixo, uma das partículas está em repouso($\vec{p}_b = 0$).

Para ilustrar, consideremos o eixo z como a direção do movimento. No sistema CM, os quadrimomentos das partículas são descritos por:

$$\begin{aligned} p_a^{CM} &= (E_a^{CM}, 0, 0, p_a^{CM}) \\ p_b^{CM} &= (E_b^{CM}, 0, 0, -p_a^{CM}) \end{aligned} \tag{2.2}$$

No sistema laboratório, os quadrimomentos são:

$$\begin{aligned} p_a^{lab} &= (E_a^{lab}, 0, 0, p_a^{lab}) \\ p_b^{lab} &= (m_b, 0, 0, 0) \end{aligned} \tag{2.3}$$

A energia total após a colisão pode ser calculada pela equação abaixo:

$$E_T = \sqrt{(p_a + p_b)^2} \tag{2.4}$$

Para o sistema CM, substituindo pelos quadrimomentos na equação 2.4, obtemos:

$$E_T = \sqrt{(p_a^{CM})^2 + (p_b^{CM})^2} = \sqrt{(E_a^{CM} + E_b^{CM})^2 - (p_a^{CM} + p_b^{CM})^2} \tag{2.5}$$

Como $p_b^{CM} = -p_a^{CM}$, simplificamos para:

$$E_T = \sqrt{(E_a^{CM} + E_b^{CM})^2} \ \therefore \ E_T = E_a^{CM} + E_b^{CM} \tag{2.6}$$

No sistema laboratório, substituindo os quadrimomentos na equação 2.4, a equação para a energia total é:

$$E_T = \sqrt{(p_a^{lab})^2 + (p_b^{lab})^2} = \sqrt{(E_a^{lab} + m_b)^2 - (p_a^{lab} + 0)^2} \tag{2.7}$$

Fazendo a distributiva

$$\sqrt{E_a^{lab^2} + 2m_b E_a^{lab} + m_b^2 - p_a^{lab^2}} \tag{2.8}$$

Lembrando que $|p^2| = p_\mu p^\mu = E^2 - |\vec{p}|^2 = m^2$

$$E_T = \sqrt{E_a^{lab^2} + 2m_b E_a^{lab} + m_b^2 - (E_a^{lab^2} - m_a^2)} \ \therefore \ E_T = \sqrt{m_a^2 + 2m_b E_a^{lab} + m_b^2} \tag{2.9}$$

No LHC, feixes de prótons são acelerados a uma energia aproximada de 7 TeV, como a massa[4] do próton é aproximadamente 1 GeV. Substituindo esses valores na equação 2.9, obtemos:

$$E_T = \sqrt{1\,\mathrm{GeV}^2 + 1\,\mathrm{GeV}^2 + 2 \times 1\,\mathrm{GeV} \times 7\,\mathrm{TeV}} = \sqrt{2 + 14000} \approx 118,33\,\mathrm{GeV} \quad (2.10)$$

No sistema de centro de massa, usando a equação 2.6, temos:

$$E_T = 7\,\mathrm{TeV} + 7\,\mathrm{TeV} = 14\mathrm{TeV} \quad (2.11)$$

Observe que a energia total obtida em colisões onde ambas as partículas são aceleradas é significativamente maior, com um fator de grandeza de 10^2.

Antes de iniciar as colisões, as partículas precisam ser aceleradas, inicialmente em aceleradores lineares, depois em aceleradores síncrotrons, como o PS e SPS, que aceleram as partículas antes de entrarem no LHC. Na figura 2.1, é possível ver o complexo de aceleradores e sua trajetória até chegarem no LHC.

As partículas produzidas após o processo de colisão são detectadas em um dos quatro detectores: *A Toroidal LHC Apparatus* (ATLAS) [18], *Compact Muon Solenoid* (CMS) [2], *A Large Ion Collider Experiment* (ALICE) [19], e *Large Hadron Collider Beauty experiment* (LHCb) [20].

[4] $E = mc^2$. Ou seja, a energia está relacionada à massa, onde a massa é frequentemente expressa em elétron-volts (eV). A expressão para a massa fica, então, eV/c^2, mas c=1, temos que a massa é dada diretamente em eV

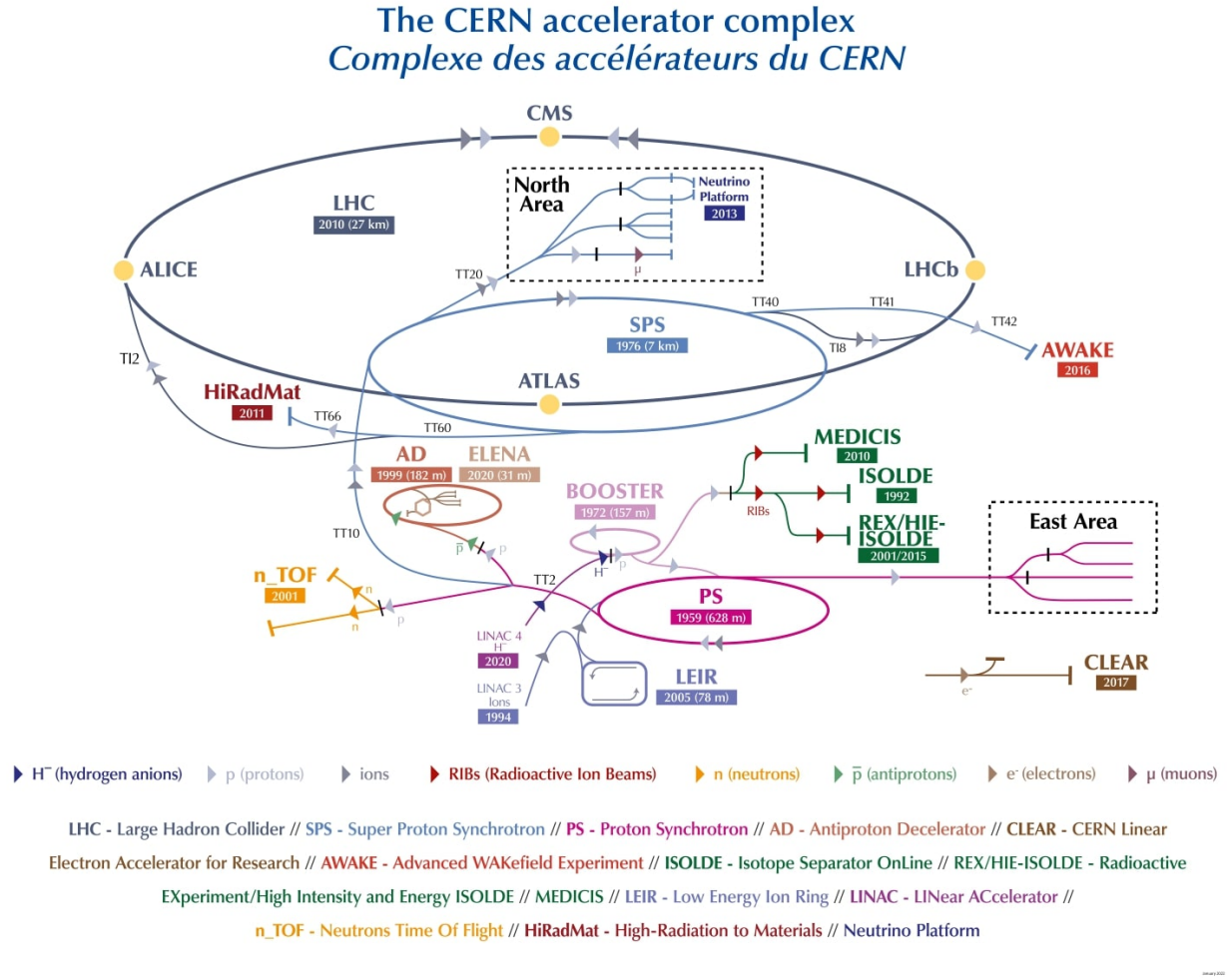

Figura 2.1: Complexo dos Aceleradores no CERN. Fonte: [1]

Neste trabalho, iremos focar apenas em um dos quatro detectores: o CMS.

2.1 O Detector CMS

O detector CMS foi construído em torno de um ímã de solenoide cilíndrico com um comprimento de 21 metros e 15 metros de diâmetro. Seus cabos supercondutores, foram projetados para gerar um campo magnético de 4 Tesla, o que é 100 mil vezes maior que o campo magnético terrestre, embora o campo magnético real alcançado seja de 3,8 Tesla. Esse campo magnético é essencial, pois, através da força de Lorentz, permite a determinação da carga, massa e momento das partículas. A força de Lorentz age sobre partículas carregadas em movimento, desviando sua trajetória e permitindo a análise de suas propriedades. Para aumentar a uniformidade e a intensidade do campo, ele é confinado por uma estrutura ferromagnética. Na figura 2.2 é possível ver o corpo do CMS.

Seu funcionamento pode ser considerado “análogo” ao de uma câmera gigante de alta velocidade, tirando "fotos" em 3D das colisões com uma velocidade de até 40 milhões de

"fotos" por segundo. A maioria das partículas produzidas após a colisão próton-próton é instável, mas permanece estável o suficiente para ser "fotografada" pelo detector.

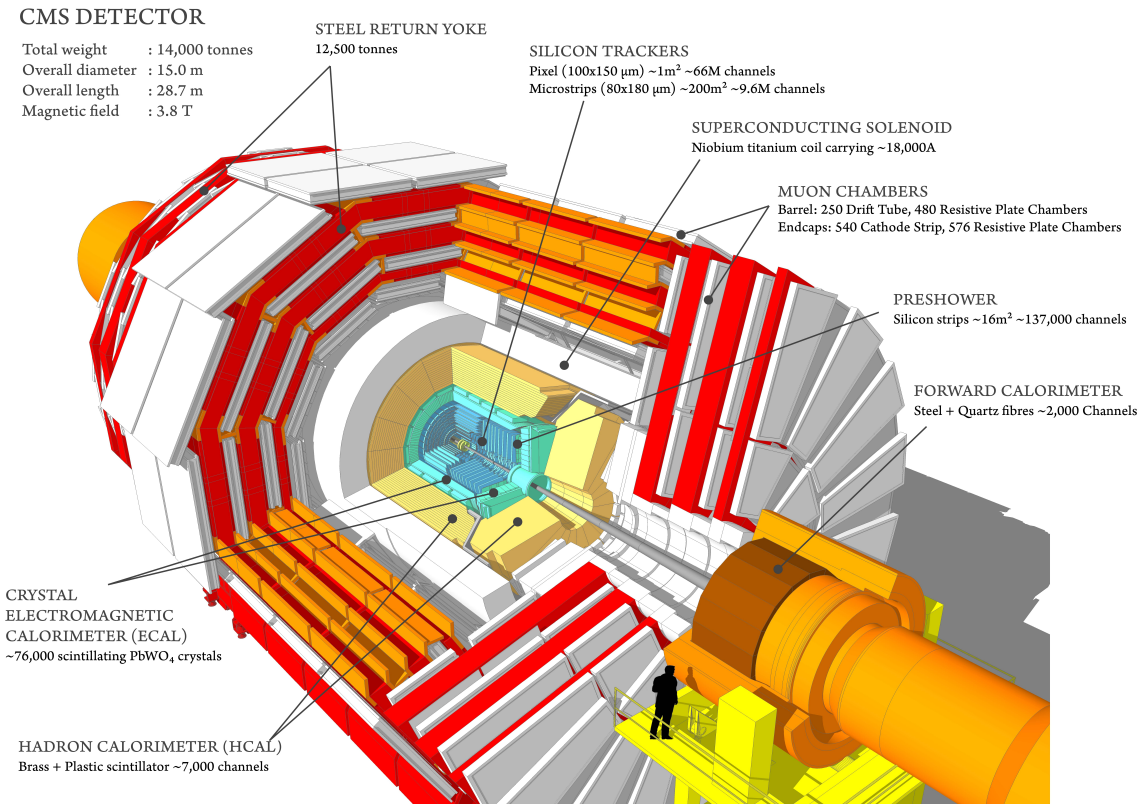

Figura 2.2: Corpo do CMS. Fonte: [2]

No trabalho, serão abordadas algumas partes do CMS e as novas atualizações da *Run* 3. Essas atualizações visam aumentar a precisão, eficiência e capacidade de lidar com o aumento de eventos, garantindo que o CMS continue a fornecer dados de alta qualidade.

2.1.1 Detecção de Traços

O traço representa a trajetória de uma partícula carregada registrada pelo detector, formando um conjunto de pontos ou sinais. A detecção desses traços, comumente chamada de *Inner Tracking System* (*Tracker*), está localizada na parte mais interna do CMS. Ela é responsável por identificar e reconstruir as trajetórias das partículas carregadas geradas nas colisões, utilizando técnicas de detecção de posição e algoritmos de reconstrução. Devido à sua posição interna, o *Tracker* é exposto a uma alta densidade de partículas carregadas e, consequentemente, a uma maior radiação. O sistema de detecção é dividido em três partes, incluindo dois detectores de micro-faixas de silício para as regiões entre 20 e 110 centímetros de raio:

- **Detector de Pixel:** Posicionado próximo ao ponto de interação, este detector é composto por três camadas de sensores de alta resolução. Ele é projetado para capturar as primeiras trajetórias das partículas logo após a colisão, em uma região com raio de cerca de 10 centímetros.

- **Detector de micro-faixas de Silício:** Este sistema é composto por múltiplas camadas de micro-faixas de silício dispostas em cilindros concêntricos ao redor do ponto de interação. Ele permite a reconstrução das trajetórias das partículas à medida que se afastam do ponto de colisão. Há dois detectores de micro-faixas de silício nas regiões: uma entre 20 e 55 centímetros de raio, e outra entre 55 e 110 centímetros de raio. Esse rastreador é utilizado para medições mais precisas dos momentos das partículas e para a identificação dos vértices primários e secundários.

A reconstrução das trajetórias é realizada por meio do reconhecimento de padrões, que inclui a identificação de interferências, ruídos, erros de medição e anomalias nas trajetórias. Após essa etapa, é feito um ajuste (*hits*) para otimizar a curva na direção onde o sinal foi detectado.

2.1.2 Calorímetro Eletromagnético

O calorímetro eletromagnético (ECAL) é projetado para medir a energia de partículas como elétrons e fótons, que são geradas em processos como: produção de pares, espalhamento Compton e *bremsstrahlung*, resultando em um jato de partículas eletromagnéticas. Este sistema é composto por cristais de tungstato de chumbo ($PbWO_4$), que são arranjados em grupos de cinco, chamados de submódulos, e são agrupados em conjuntos de cristais de 5×5, denominados super-cristais. Os super-cristais são, então, agrupados em dois semicírculos chamados *Dees*.

Quando uma partícula eletromagnética interage com os cristais de $PbWO_4$, ela produz um chuveiro de partículas secundárias que geram luz de cintilação. Essa luz é capturada pelos fotodetectores.

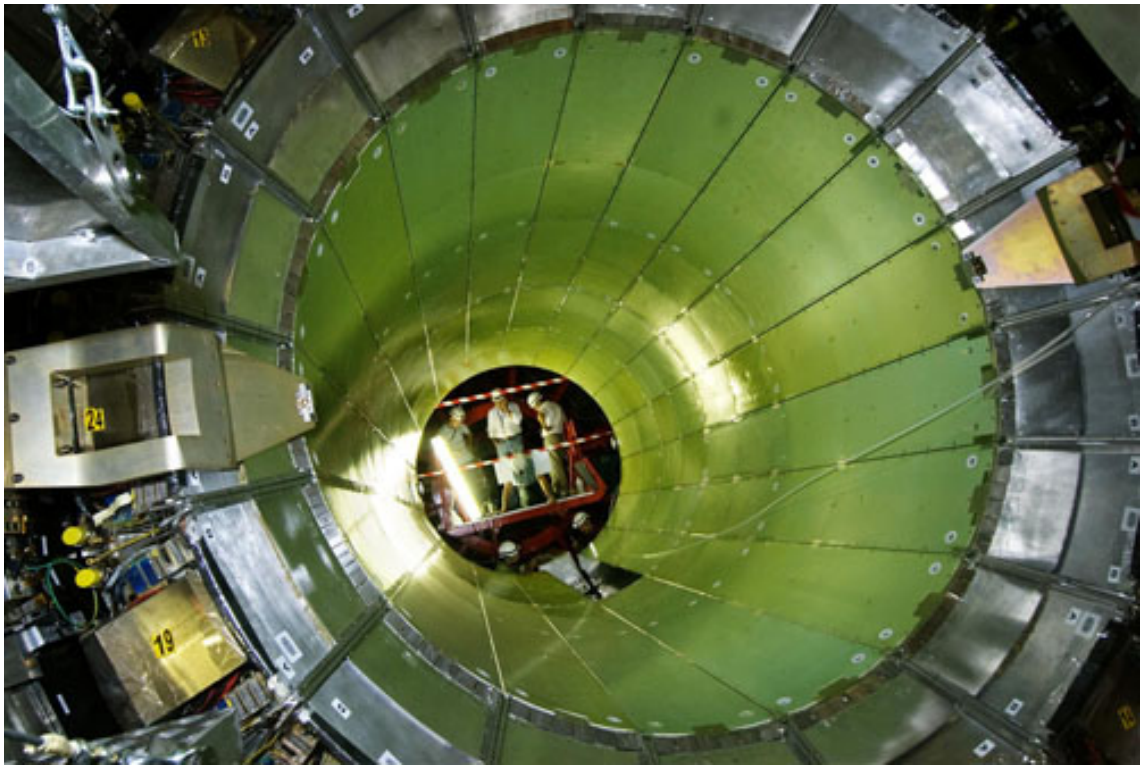

Figura 2.3: Visão interna do ECAL do detector CMS, exibindo a seção de cristais de $PbWO_4$.

Os cristais têm uma cintilação verde-azulado com uma amplitude de 420 nm. A variação da cintilação é de $-1,9\%$ por grau Celsius, com estabilidade a 18°C. Para manter a resolução do equipamento estável, o intervalo de temperatura pode variar apenas $\pm 0,05$°C.

Para a detecção da cintilação, são utilizados fotodetectores. Devido ao campo magnético de 3,8 T, na região do barril são usados Fotodiodos Avalanche (APDs), enquanto nas regiões das tampas são utilizados Fotomultiplicadores de Vácuo (VPTs). Ao interagir com os fotodetectores, a luz de cintilação é convertida em sinais elétricos, que são amplificados e digitalizados para processamento posterior.

2.1.3 Calorímetro Hadrônico

O calorímetro hadrônico (HCAL) é responsável pela medição das partículas hadrônicas, ou seja, partículas compostas por quarks. Ele complementa o ECAL, possibilitando a medição de energia de jatos. A estrutura do HCAL é composta por camadas alternadas de materiais absorventes e sensíveis à passagem de partículas. Vale ressaltar que os outros sistemas, como o *Tracker* e o ECAL, estão localizados dentro do HCAL.

O HCAL tem duas partes principais: o *Hadron Barrel* (HB) e o *Hadron Endcaps* (HE). Além disso, existem seções especializadas para jatos dianteiros, o *Hadron Forward* (HF), e outra na parte externa do solenoide, o *Hadron Outer* (HO), que aumenta a retenção de chuveiros centrais. Na figura 2.4, é possível observar a localização dessas partes.

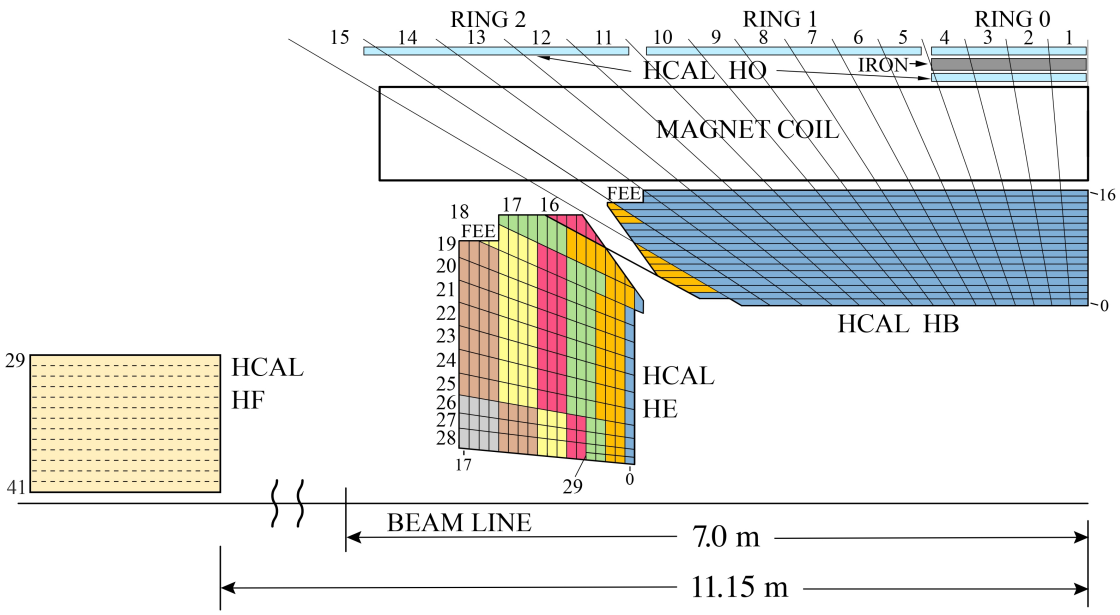

Figura 2.4: Seções do HCAL, incluindo: HB, HE, HF e HO. As divisões em η no HB e HE variam de 1 a 28, enquanto no HF as divisões são apenas horizontais, numeradas de 29 a 41. As diferentes cores indicam as camadas que medem a profundidade dos chuveiros hadrônicos, sendo mais relevantes em regiões de alto η, como no HE. O eixo "z" representado no diagrama é o "*BEAM LINE*" , que indica a direção do feixe de partículas.

As seções HF e HO têm uma construção um pouco diferente. O HF não utiliza cintiladores, mas sim absorção através de aço e fibras de quartzo para a detecção da radiação de Cherenkov[5]. Já o HO apresenta apenas uma camada de cintiladores, semelhante aos do ECAL.

O HCAL cobre uma ampla faixa angular em torno do ponto de colisão, cerca de 98%, garantindo que a energia de todas as partículas hadrônicas seja medida, independentemente da direção de emissão.

2.1.4 Sistema de Múon

O sistema de múons está localizado na camada mais externa do detector. Os múons são partículas carregadas que, devido à sua massa relativamente alta e à ausência de interações fortes, conseguem atravessar as camadas internas do CMS (rastreador e calorímetros) praticamente sem sofrer deflexão significativa. Ao passar pelo ECAL, os múons perdem pouca energia e continuam atravessando o HCAL sem dificuldades. Devido à sua capacidade de identificação, os múons são ideais para a criação de sistemas de *triggers* baseados em sua presença, permitindo a redução da taxa de eventos gravados.

[5]Emissão de luz azulada visível quando uma partícula carregada viaja através de um meio com velocidade superior à da luz nesse meio.

O sistema de múons mede a trajetória das partículas ao longo das camadas do detector, possibilitando a reconstrução da sua trajetória. Ele é composto por quatro tipos principais de detectores:

- ***Drift Tubes* (DT)**: Utilizadas principalmente na região central do detector (barril), onde o campo magnético é mais contido pelo núcleo de ferro. Nos DTs, fios centrais atuam como ânodos, atraindo elétrons liberados quando o múon ioniza o gás presente na câmara. As DTs são eficazes para medir com precisão a trajetória de múons que se movem quase paralelamente ao feixe de colisão.

- ***Cathode Strip Chambers* (CSCs)**: Estas câmaras são usadas nas extremidades (*endcaps*) do detector, onde o campo magnético é mais intenso e inomogêneo. As CSCs são formadas por segmentos de planos catódicos em faixas ortogonais aos planos dos fios ânodos. A interação do múon com o gás na câmara induz uma carga em várias faixas do plano catódico, permitindo a localização precisa das partículas em qualquer coordenada do plano.

- ***Resistive Plate Chambers* (RPCs)**: Operam em conjunto com as DTs e CSCs. Os RPCs têm eletrodos paralelos que apresentam baixa resolução espacial, mas alta velocidade de leitura, sendo, portanto, fundamentais para o sistema de disparo do *Level 1 trigger* , que requer respostas rápidas.

- ***Gas Electron Multipliers* (GEMs)**: Localizados na frente do anel interno das câmaras CSC na primeira estação dos *endcaps*. Os GEMs possuem uma resposta rápida e boa resolução espacial, complementando os CSCs em regiões de alto fluxo de partículas. As câmaras GEM são preenchidas com uma mistura de gases (Ar:CO_2) que, sob a influência de um campo elétrico, permite a ionização dos elétrons, melhorando a detecção de múons.

2.2 Sistema Computacional de Aquisição dos Dados

O sistema de aquisição de dados (DAQ) é um componente fundamental, que coleta, processa e armazena os dados gerados pelo detector. O CMS produz cerca de 10^9 eventos por segundo, o que pode gerar 1 Petabyte/s. No entanto, nem todos esses dados são armazenados; apenas 10 Gigabytes/s são retidos. Dado o longo período do experimento, uma das principais necessidades foi a utilização de computação em grade (*Grid*). Suas operações são organizadas através do projeto *Worldwide LHC Computing Grid* (WLCG) [21].

As principais funções do sistema computacional são:

- Pré-processar e selecionar os eventos a serem salvos.

- Organizar e transferir os dados processados, com uma catalogação que os torna acessíveis para a colaboração científica internacional.

- Fornecer ferramentas para análise de dados padronizada, garantindo a reprodutibilidade das análises.

Os *Grids* responsáveis pelo processamento do CMS são divididos em três partes, conforme mostrado na Figura 2.5, indo do Tier-0 ao Tier-2. Cada Tier possui responsabilidades distintas:

- **Tier-0:** Localizado no CERN, responsável pela aquisição dos dados brutos e sua reconstrução inicial. Esta etapa inclui a conversão de sinais analógicos para digitais e o processamento inicial.
- **Tier-1:** Centros localizados em vários países com instituições vinculadas ao CMS, responsáveis pela criação de réplicas dos dados brutos (Tier-1) para garantir redundância, além da re-reconstrução e armazenamento de longo prazo dos dados.
- **Tier-2:** Localizado em Instituições de pesquisa com menor poder computacional, que fornecem acesso computacional aos colaboradores para análise de dados, estudos de calibração e alinhamento, simulações e transferência de dados para o Tier-1 associado.

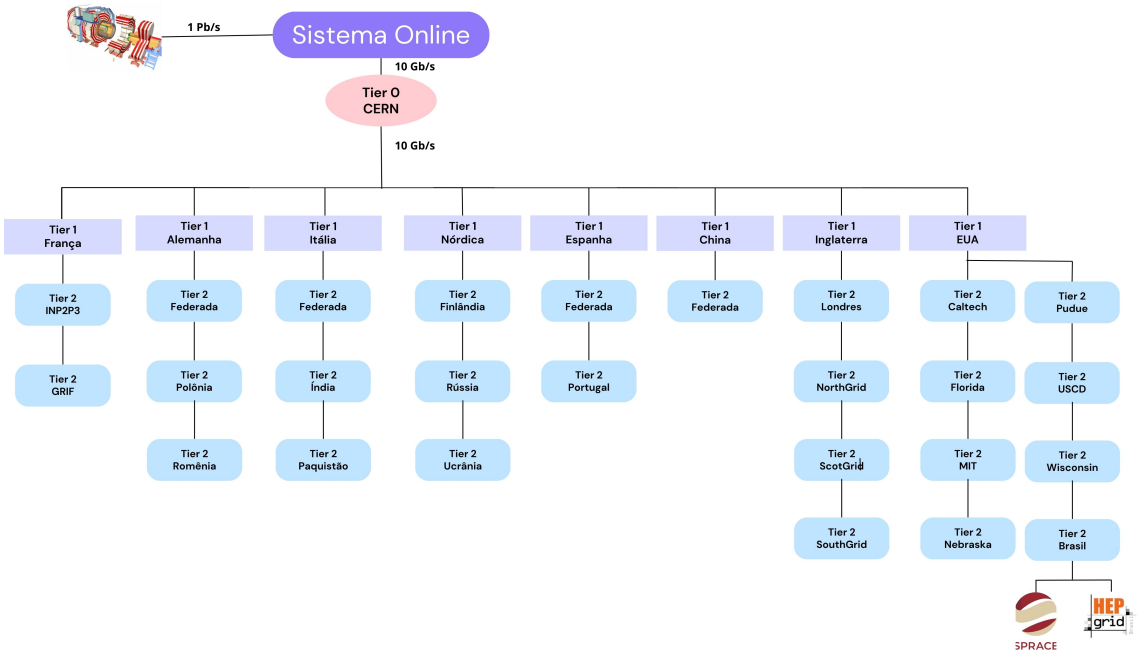

Figura 2.5: Complexo de *Grids* Espalhados pelo Mundo.

2.3 Trigger

O sistema de *trigger* é essencial para o funcionamento do CMS. Sem ele, todos os eventos gerados seriam registrados, o que é inviável devido às limitações da eletrônica atual e aos problemas significativos de armazenamento, dado que a taxa máxima de

armazenamento é na faixa de kHz, é necessário reduzir a taxa de eventos para essa faixa, para possibilitar o salvamento dos dados. Os *triggers* resolvem essa questão ao filtrar e selecionar os eventos mais relevantes para armazenamento. Isso permite reduzir a taxa de eventos detectados pelo *Level 1 trigger* (L1)[22] de 40 MHz para aproximadamente 100 kHz. Essa redução é alcançada por meio de micro análises, no caso do *High-Level trigger* (HLT)[23], a taxa é reduzida ainda mais, para valores entre 1 e 2 kHz. Os *triggers* do CMS são divididos em duas partes:

- **L1:** Responsável pelo primeiro nível de seleção, baseado em *hardware* especializado, como *Field Programmable Gate Arrays* (FPGAs).
- **HLT:** Diferente do L1, que é implementado em *hardware*, o HLT é uma implementação em *software*. Esse *software* é utilizado para a reconstrução e seleção dos eventos padrões do CMS.

2.4 Melhorias da Última Atualização

O LHC iniciou sua *Run* 3 com várias atualizações e melhorias no detector CMS. Essas melhorias abrangem diversos sistemas de detecção, incluindo a substituição dos fotodetectores antigos por novos modelos de silício, os *Silicon Photomultipliers* (SiPMs). Os SiPMs são mais eficientes, possuem uma melhor resposta linear e são menos sensíveis à radiação. Nos calorímetros, houve aprimoramentos tanto no ECAL quanto no HCAL. O ECAL recebeu uma nova eletrônica de leitura e um aprimoramento no algoritmo de *trigger*, enquanto o HCAL foi equipado com novos sensores e sistemas de leitura.

O DAQ também passou por melhorias, com o aumento da capacidade de processamento com a troca dos processadores por modelos lançados em 2021. Para aumentar a segurança, o sistema de monitoramento contínuo foi aprimorado e a redundância foi implementada para melhorar a confiabilidade.

O *triggers* L1 recebeu um aprimoramento para filtrar os eventos com mais precisão e velocidade, utilizando novos algoritmos. Já o HLT foi aprimorado com melhorias no *software*, permitindo uma análise mais complexa e em tempo real dos eventos.

3 Modelo Padrão

Na física de partículas, existem três interações fundamentais da natureza, cada uma com suas características:

- Força Eletromagnética: Atua na interação entre partículas carregadas, como os elétrons, e é descrita pela Eletrodinâmica Quântica (QED). O Fóton é o Bóson responsável por mediar essa força.
- Força Forte: Opera na reação hadrônica entre quarks, e é descrita pela Cromodinâmica Quântica (QCD). O Gluôn é o Bóson mediador.
- Força Fraca: Responsável pelo decaimento β, e em reações de fissão e fusão nuclear em estrelas, é descrita pela Dinâmica de Sabores Quânticos (QFD). É mediada pelos Bóson W e Z.

A gravidade, embora seja uma das quatro forças fundamentais, não está incluída nesta lista, pois sua interação em escala subatômica é irrelevante. A maioria dos fenômenos físicos em escala subatômica pode ser descrita em termos de elétrons, neutrinos do elétron, prótons e nêutrons, que participam apenas das interações forte, fraca e eletromagnética.

Tabela 1: Forças da natureza em relação a suas magnitudes

Força	**Magnitude**	**Bóson**	**Spin**	**Mass/GeV**
Forte	1	Gluôn **g**	1	0
Eletromagnética	10^{-3}	Fóton γ	1	0
Fraca	10^{-8}	Bóson W $\mathbf{W}^{\pm}$	1	80,4
		Bóson Z **Z**	1	91,2

Em escalas de altas energias, foi possível observar que prótons e nêutrons são estados ligados de partículas elementares chamadas quarks (q). O próton é formado por dois quarks up e um quark down (uud), enquanto o nêutron é composto por dois quarks down e um quark up (ddu).

Os quarks up, down, o neutrino do elétron e o elétron são chamados de partículas elementares de primeira geração. No entanto, experimentos de altas energias revelaram a existência de mais duas gerações para cada uma das partículas da primeira geração [24], diferenciando-se apenas pela massa. Na figura 3.1, é possível observar cada partícula elementar e sua respectiva geração. O múon (μ) é cerca de 200 vezes mais pesado que o elétron, mas sua carga e spin permanecem os mesmos: -1 para a carga e 1/2 para o spin.

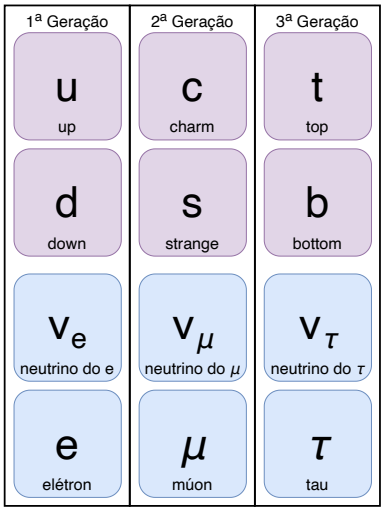

Figura 3.1: Partículas Elementares e Suas Gerações

Até o momento, os experimentos realizados no LHC e em outros colisores de partículas não encontraram evidências diretas de uma quarta geração de partículas. Isso não significa que não possa existir, mas sugere que, se existir, sua manifestação estaria além das capacidades tecnologias atuais.

Dessa forma, toda a matéria usual do universo é composta por apenas doze partículas fundamentais. Na tabela 2, fica evidente que os neutrinos não possuem uma massa bem definida. Isso se deve ao fato de ainda não conhecermos o valor exato da sua massa, embora se saiba que os neutrinos têm massa, a tecnologia atual não é capaz de determiná-la com precisão devido à sua extrema pequenez, sendo apenas possível estimar que sua massa é aproximadamente nove ordens de grandeza menor do que a das outras partículas.

Tabela 2: Gerações de Partículas Elementares

Partícula	**Carga**	**Massa (GeV)**	**Geração**
Léptons			
Elétron (e^-)	-1	$0,511 \times 10^{-3}$	$1^{\underline{a}}$
Neutrino Eletrônico (ν_e)	0	$< 10^{-9}$	$1^{\underline{a}}$
Múon (μ^-)	-1	0.106	$2^{\underline{a}}$
Neutrino Muônico (ν_μ)	0	$< 10^{-9}$	$2^{\underline{a}}$
Tau (τ^-)	-1	1.78	$3^{\underline{a}}$
Neutrino Tauônico (ν_τ)	0	$< 10^{-9}$	$3^{\underline{a}}$
Quarks			
Quark Up (u)	$+2/3$	$0,003$	$1^{\underline{a}}$
Quark Down (d)	$-1/3$	$0,005$	$1^{\underline{a}}$
Quark Charm (c)	$+2/3$	$1,3$	$2^{\underline{a}}$
Quark Strange (s)	$-1/3$	$0,1$	$2^{\underline{a}}$
Quark Top (t)	$+2/3$	174	$3^{\underline{a}}$
Quark Bottom (b)	$-1/3$	$4,5$	$3^{\underline{a}}$

Cada uma dessas doze partículas pode ser descrita utilizando uma aproximação da teoria quântica de campos, requerendo a utilização da equação de Dirac.

Para obter a equação de Dirac, partimos da equação de Klein-Gordon, que foi a primeira tentativa de unificar a relatividade com a mecânica quântica [25], mas aplicável apenas a partículas escalares, ou seja, partículas sem spin. A equação de Klein-Gordon fornece uma descrição relativística para essas partículas, mas não inclui o spin ou a previsão de antipartículas. A equação de Dirac, desenvolvida posteriormente, representa um avanço significativo, oferecendo uma descrição mais abrangente ao incluir tanto o spin das partículas quanto a existência de antipartículas, uma previsão que foi confirmada experimentalmente. Assim, a equação de Dirac pode ser considerada uma generalização da equação de Klein-Gordon. Isso é evidenciado ao elevar a equação de Dirac ao quadrado, o que resulta na equação de Klein-Gordon para partículas escalares. Portanto, a equação de Dirac amplia a descrição fornecida pela equação de Klein-Gordon.

A equação de Dirac permite visualizar que, para cada partícula, deve existir uma antipartícula, ou seja, uma partícula com a mesma massa e spin, mas com carga oposta. A notação de antipartícula para quarks e neutrinos é dada por uma barra sobre o símbolo, como no antiquark ($\bar{q}$) e no antineutrino do elétron ($\bar{\nu_e}$). Para os elétrons de carga negativa, usa-se o símbolo positivo, como no pósitron (antipartícula do elétron) e^+.

3.1 Força Forte

Devido à interação forte mediada pela QCD em baixas energias, é impossível encontrar um quark livre na natureza. À medida que os quarks se afastam uns dos outros, a energia potencial da interação forte aumenta linearmente, um fenômeno conhecido como confinamento de quarks. A energia potencial $U(r)$ entre dois quarks separados por uma distância r pode ser aproximada pelo potencial de Cornell, que é dado por:

$$U(r) = -\frac{4}{3}\frac{\alpha_s}{r} + \sigma r \tag{3.1}$$

onde α_s é a constante de acoplamento forte, sendo representada por:

$$\alpha_s(r) = \frac{12\pi}{(33 - 2n_f)\ln\left(1/\Lambda_{\mathrm{QCD}}\, r\right)} \tag{3.2}$$

Sendo Λ_{QCD} o fator de escala associado à QCD, a constante de acoplamento depende da escala de energia. À medida que a energia aumenta, a distância de acoplamento tende a zero, o que é um fenômeno conhecido como liberdade assintótica. r a distancia entre os quarks, 4/3 o fator de cor apropriado, e σ é a constante de tensão de cor, que tem um valor aproximado de 16 toneladas [26]. O termo $\frac{4}{3}\frac{\alpha_s}{r}$ representa a interação atrativa de curto alcance, enquanto o termo σr descreve a interação atrativa de longo alcance, que cresce linearmente com r. Este crescimento da energia potencial resulta na formação de novos hádrons, pois a energia envolvida no processo é suficiente para criar pares de quarks

e antiquarks, em vez de permitir que os quarks se afastem indefinidamente. Assim, quarks são encontrados exclusivamente confinados em hádrons.

Os hádrons são divididos em três categorias: bárions, mésons e hádrons exóticos. Partículas bariônicas consistem de três quarks, enquanto as mesônicas são formadas por um quark e um antiquark (ver 3.2). As partículas bariônicas mais conhecidas são o próton e o nêutron, e um exemplo de méson é o méson-π, descoberto pelo brasileiro César Lattes [27].

Além de serem constituídos por quarks, os hádrons devem seguir a regra do sabor, cada quark possui uma propriedade chamada "cor", que é uma forma de carga relacionada à interação forte. Os quarks podem ter uma cor vermelha, verde ou azul. Para manter a neutralidade de cor, um hádron deve ser composto por combinações de quarks e/ou antiquarks que resultem em um estado "sem cor" [28]. No caso dos antiquarks, eles possuem a propriedade oposta, chamada de "anticor", que pode ser representada como antiverde, antiazul e antivermelho, respectivamente.

Figura 3.2: Representação de hádrons - Bárion, Anti-Bárion e Méson.

Além da cor, os quarks também são caracterizados por outras propriedades, como a hiper carga (Y) e a estranheza (S). A hiper carga é obtida pela equação:

$$Y = B + S \tag{3.3}$$

Onde B é o número bariônico, e S o número da estranheza, com a hiper carga junto com o número de isospin I_3, é possível determinar a carga elétrica Q de uma partícula. Ficando com uma relação de :

$$Q = I_3 + \frac{Y}{2} \tag{3.4}$$

Onde I_3 pode assumir $+1/2$ e $-1/2$ para o quarks up e down respectivamente.

O grupo de simetria SU(3) é uma ferramenta importante para a organização dos hádrons. Ele é baseado na simetria de sabores dos quarks[6]: up (u), down (d) e strange (s). Esta simetria organiza os hádrons em multípletos, que podem ser visualizados em gráficos que relacionam a estranheza S com a componente do isospin I_3 Como mostra a figura3.3 .

[6]O trabalho abordará apenas os quarks leves (u, d, s).

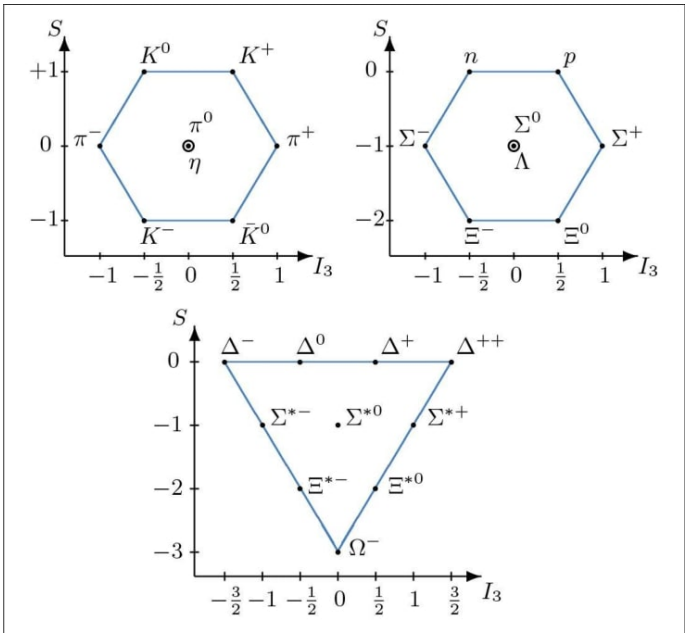

Figura 3.3: Diagramas de Multípletos de Partículas no Modelo de Quarks. Fonte: [3]

Essa organização e visualização no gráfico ajudam a compreender a estrutura das partículas e suas interações, proporcionando uma visão das simetrias envolvidas no modelo de quarks e hádrons.

3.2 Força Eletromagnética

A interação eletromagnética é responsável por uma ampla gama de fenômenos, desde a luz que percebemos até as forças que mantêm os elétrons no núcleo atômico. Essa interação é mediada pelo Fóton, que é o Bóson associado a força eletromagnética, transportando a interação entre as cargas elétricas. A QED descreve essas interações, mostrando como a luz interage com a matéria e explicando fenômenos como a polarização e a dispersão da luz. Por meio da interação eletromagnética, podendo descrever não apenas as propriedades da luz, mas também as formas como átomos e moléculas interagem e se organizam.

A força eletromagnética se entrelaça com outra força fundamental: a força fraca. Formando a teoria eletrofraca, essa teoria unifica as duas interações em uma única descrição. Mas para que essa unificação ocorra, é necessário compreender a quebra de simetria associada ao mecanismo de Higgs. A quebra de simetria de Higgs é um fenômeno que acontece em altas energias, quando partículas adquirem massa através do campo de Higgs. No estado original, antes da quebra, as interações eletromagnéticas e fracas podem ser vistas como simétricas. Contudo, após a interação com o campo de Higgs, essa simetria é quebrada, e algumas partículas, como os Bósons W e Z, passam a ser produzidas, enquanto o Fóton permanece sem massa.

3.3 Força Fraca

A interação fraca, além de ser responsável pelos processos de decaimento, também desempenha um papel na troca de sabor dos quarks. Como mostrado na tabela 1 o Bóson W tem duas variantes carregadas: W^+ e W^-. Esses Bósons são responsáveis por mudar o "sabor" dos quarks, ou seja, mudar a família de um quark. Isso significa que eles podem transformar um tipo de quark em outro tipo diferente (ou seja, mudar a família de um quark). Por exemplo, um quark down pode se transformar em um quark up por meio da troca de sabor mediada pelo Bóson W^-. Em contraste, o Bóson Z é neutro e não altera o sabor das partículas. Ele media processos de interação fraca onde não ocorre mudança no tipo de quark ou lépton, preservando assim o sabor das partículas.

3.3.1 Decaimento do Bóson W

O Bóson W tem uma vida útil muito curta e decai rapidamente em outras partículas. Os decaimentos mais comuns do Bóson W são: $q + \bar{q}$ e $lepton + \nu_{lepton}$. Por exemplo, o decaimento de um nêutron é um processo típico mediado pelo Bóson W^-. Após aproximadamente 15 minutos, um nêutron se transforma em um próton, emitindo um elétron e um antineutrino do elétron.

Para que um nêutron se transforme em um próton, um dos quarks d no nêutron precisa se transformar em um quark u. Essa transformação de sabor é mediada pelo Bóson W^-:

$$n(udd) \rightarrow p(uud) + W^- \tag{3.5}$$

Como o Bóson W^- decai rapidamente, o decaimento final do nêutron é:

$$n(udd) \rightarrow p(uud) + e^- + \bar{\nu}_e \tag{3.6}$$

Este processo preserva os princípios de conservação de carga, número bariônico e número leptônico.

Outro exemplo é o decaimento de um quark b, que troca de sabor para um quark c, novamente sendo mediado pelo Bóson W^-:

$$b \rightarrow c + W^- \tag{3.7}$$

O Bóson W^- decai em um par quark-antiquark, ao contrário do decaimento do nêutron onde o decaimento era em um elétron e um antineutrino do elétron. No quark b, o decaimento final é:

$$b \rightarrow c + d + \bar{u} \tag{3.8}$$

Assim, o decaimento do quark b para um quark c é seguido pela emissão de um quark d e um antiquark $\bar{u}$ preservando as leis de conservação.

3.4 Jatos de Partículas

Em colisões de prótons-prótons (p-p) em altas energias, ocorre o fenômeno de hadronização, que pode resultar na produção de uma grande quantidade de partículas. Esse fenômeno é descrito pelo conceito de jatos de partículas, onde um jato é formado por quarks que estão se movendo na mesma direção.

A hadronização ocorre quando um par de quarks é criado pela produção de partículas; neste caso, para fins de exemplo, utilizando o Bóson W em repouso no referencial de laboratório. Os quarks gerados se movem em direções opostas. Como explicado na seção 3.1, á medida que os quarks se afastam um do outro, a interação forte aumenta a energia potencial linearmente, o que leva à formação de um aglomerado de quarks. Esse processo resulta na criação espontânea de um par adicional de quarks entre os quarks que estão se afastando. Por ser uma colisão p-p surgem vários hádrons, que podem decair em outros hádrons ou diretamente em léptons, por isso são detectadas muitas partículas.

As partículas tendem a se propagar dentro de um formato cônico com vértice no ponto onde ocorreu a colisão, como ilustra a figura 3.4

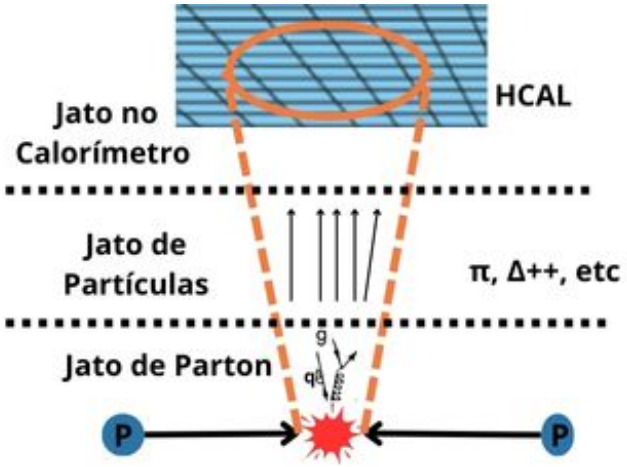

Figura 3.4: Sequência de Formação de Jatos: Do Parton ao Calorímetro.

Após a colisão p-p, um jato inicial de partons (quarks e glúons) são gerados. Esses partons passam por processos de fragmentação e hadronização, resultando na formação de partículas hadrônicas, como prótons, píons, entre outras. O jato de partículas interage com o calorímetro hadrônico para medir a energia das partículas produzidas.

3.4.1 Algoritmos de Detecção de Jatos

A definição de jatos de partículas deve atender a critérios que permitam uma comunicação clara e sem ambiguidades entre teóricos e experimentais. Esses critérios incluem a necessidade de um algoritmo simples e eficiente que possa ser utilizado tanto em análises experimentais quanto em cálculos teóricos.

No CMS o feixe de partículas vem na direção da coordenada z, pelo colisor ter um

formato cilíndrico (ver fig 3.5) é possível usar o sistema de coordenadas cilíndricas, ficando então com P, θ, ϕ. Pela simetria a coordenada x que tem o ângulo ϕ será constante.

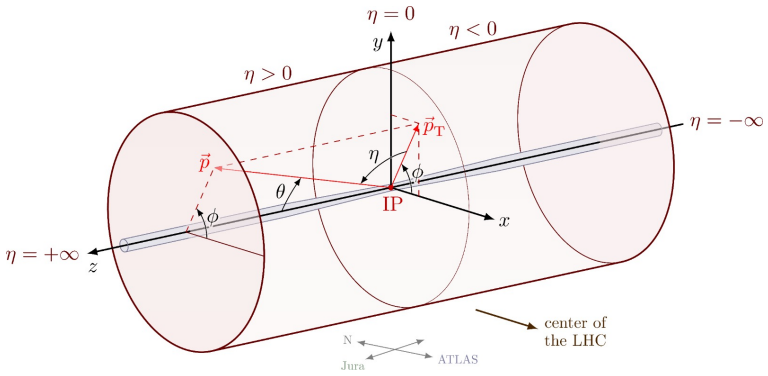

Figura 3.5: Sistema de coordenadas do CMS. Fonte [4]

Assim sobrando apenas P e θ, onde θ é o ângulo de espalhamento, é possível determinar o momento transversal (p_T):

$$p_T = P \sin\theta \tag{3.9}$$

Para encontrar a rapidez é preciso utilizar a transformação de Lorentz na forma matricial, onde:

$$\beta = \tanh y; \gamma = \cosh y; \gamma\beta = \sinh y$$

$$\begin{pmatrix} E' \\ p'_{\parallel} \end{pmatrix} = \begin{pmatrix} \cosh y & -\sinh y \\ -\sinh y & \cosh y \end{pmatrix} = \begin{pmatrix} E \\ p_{\parallel} \end{pmatrix}; \; p'_{\perp} = p_{\perp} \tag{3.10}$$

Que inversamente, já com o y isolado, fica:

$$y = \tanh^{-1}\beta \tag{3.11}$$

Lembrando que relação de $\tanh^{-1}$:

$$\tanh^{-1} x = \frac{1}{2}\ln\left(\frac{1+x}{1-x}\right)$$

Aplicando a relação na equação 3.11:

$$y = \tanh^{-1}\beta = \frac{1}{2}\left(\frac{1+\beta}{1-\beta}\right) \tag{3.12}$$

Relacionando com o momento:

$$y = \frac{1}{2}\left(\frac{E+p_{\parallel}}{E-p_{\parallel}}\right) \tag{3.13}$$

Por se tratar de altas energias, a rapidez pode ser aproximada pela pseudo-rapidez (η), assim tendo uma relação simples com o ângulo de espalhamento θ :

$$\eta = \frac{1}{2}\left(\frac{1 + cos\theta}{1 - \cos\theta}\right) = -\ln\left(\tan\frac{\theta}{2}\right) \tag{3.14}$$

Para descobrir uma aproximação da rapidez. Só depende do ângulo de espalhamento θ, a pseudo-rapidez é extremamente útil quando não se sabe o momento ou massa de uma partícula. Também é utilizada para a parametrização da cobertura dos detectores, por exemplo nas células de calorímetro que são dadas em (η,ϕ).

As equações 3.9 e 3.14 podem ser utilizadas nos algoritmos de jatos, porém, com uma certa mudança utilizando a energia transversal (E_T) que é proporcional ao p_T, a pseudo-rapidez, e o p_T, porem escrito em forma do componente do momento no plano transverso (x,y), ficando no final com as seguintes variáveis:

$$\eta = -\ln\left(\tan\frac{\theta}{2}\right) \quad E_T = E\sin\theta \quad p_T = \sqrt{p_x^2 + p_y^2} \tag{3.15}$$

Existem dois tipos de algoritmos para encontrar jatos [29]:

- Algoritmos de cone
- Algoritmos de recombinação

3.4.1.1 Algoritmos de Cone

No algoritmo Cone, pontos conhecidos como sementes são escolhidos com base na condição de que a energia acumulada nesses pontos seja maior ou igual à energia de corte (E_0). Para cada ponto que satisfaz essa condição, calcula-se a área do cone ao redor dele, incluindo todas as partículas dentro de uma distância ΔR_{is}, dada pela equação:

$$\Delta R_{is} = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2} \leq R_{cone} \tag{3.16}$$

Onde R_{cone} é o raio do cone no espaço (η,ϕ), e $\Delta\eta$, $\Delta\phi$ as diferenças de pseudo-rapidez e ângulo azimutal, respectivamente, em relação à semente. Usualmente se utiliza o valor de $R_{cone} = 0,7$.

3.4.1.2 Algoritmos de Recombinação

Os algoritmos de recombinação utilizam a soma dos quadrivetores de partículas para separar os objetos em duas classes: proto-jatos e jatos. Inicialmente, o algoritmo trabalha apenas com proto-jatos. Para cada proto-jato, calcula-se sua grandeza d_{iB} como:

$$d_{iB} = P_{Ti}^2 \tag{3.17}$$

Em seguida, para cada par de proto-jatos i, j calcula-se a distância d_{ij} dada por:

$$d_{ij} = \min\left(P_{T_i}^{2v}, P_{T_j}^{2v}\right)\frac{\Delta R_{ij}^2}{R^2} \tag{3.18}$$

O algoritmo então compara os valores de d_{iB} e d_{ij}. Se d_{ij} for o menor valor, os proto-jatos i e j são combinados em um novo proto-jato. Se d_{iB} for menor, o proto-jato é promovido à classe de jatos e movido para a lista de jatos. Esse processo continua até que todos os proto-jatos tenham sido processados e não restem mais elementos na lista.

Para cada valor de v (sendo $v = 1$, 0 ou -1) associado aos elementos d_{iB} e d_{ij}, é utilizado um algoritmo de reconstrução diferente. Para $v = 1$, emprega-se o algoritmo k_t. Neste caso d_{ij} é proporcional ao quadrado do p_T dos proto-jatos, e, para um mesmo ΔR_{ij}, o algoritmo prioriza a combinação de proto-jatos com menor troca de momento. Para $v = 0$, utiliza-se o algoritmo *Cambridge/Aachen*, onde d_{ij} é proporcional a ΔR_{ij}^2, favorecendo a combinação de proto-jatos mais próximos no espaço (η,ϕ). Com $v = -1$ o algoritmo anti-k_t é utilizado, onde d_{ij} é proporcional a $\frac{1}{p_{T,i}^2}$, resultando na combinação de proto-jatos com maior p_T. Na figura 3.6 é possível observar os diferentes formatos de jatos para cada valor de v. Observa-se que o algoritmo anti-k_t é o qual mais se tem semelhanças com a estrutura de cone, já que tem uma forma mais arredondada.

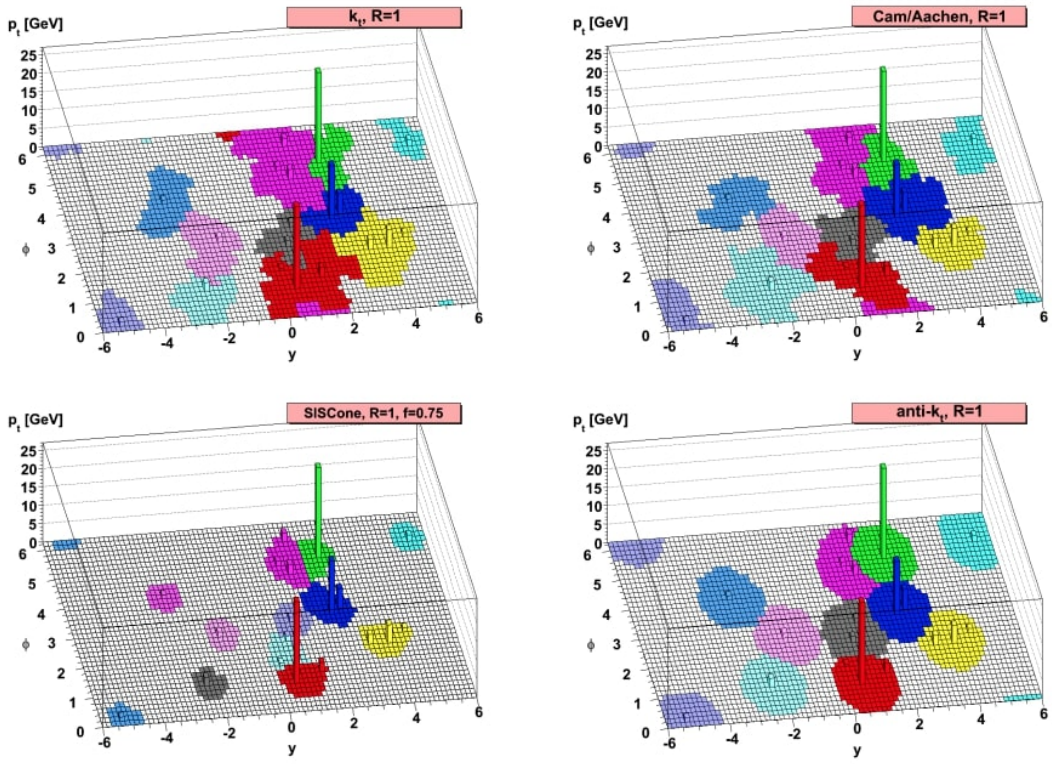

Figura 3.6: Diferentes algoritmos de reconstrução de jato. Fonte: [5]

3.4.2 Jatos Gordos

Ao analisar o decaimento de uma partícula no referencial do centro de massa (CM), onde a partícula inicial decai em duas partículas, elas tendem a se afastar em direções opostas [30]. No entanto, no referencial do laboratório (lab), para partículas com alto p_T, o ângulo entre a velocidade final das partículas em relação à partícula antes do decaimento é dado pela expressão:

$$\cos(\theta) = \frac{E - E_0\sqrt{1-\vec{\beta^2}}}{\vec{\beta}\sqrt{E^2 - m^2}} \tag{3.19}$$

Onde E é a energia de uma das partículas resultantes do decaimento no referencial lab, E_0 é a energia da partícula no referencial CM, $\vec{\beta}$ representa a velocidade da partícula antes do decaimento, e m é a massa da partícula após o decaimento em referencial lab. Assim, quando a velocidade da partícula inicial é igual à velocidade da luz, ou seja, $\beta = 1$ o ângulo θ tende a zero. Isso implica que, para partículas com altas velocidades, o decaimento resulta em duas partículas com velocidades quase paralelas.

O Bóson W pode decair em um par quark-antiquark, conforme ilustrado na Figura 3.7), representado pela reação: $W \to q\overline{q}$. Durante o processo de hadronização, esse par de quarks pode produzir dois jatos no sistema. Isso cria uma dificuldade na identificação, pois pode ser difícil distinguir se as partículas detectadas provêm de um único jato ou de dois jatos muito próximos um do outro.

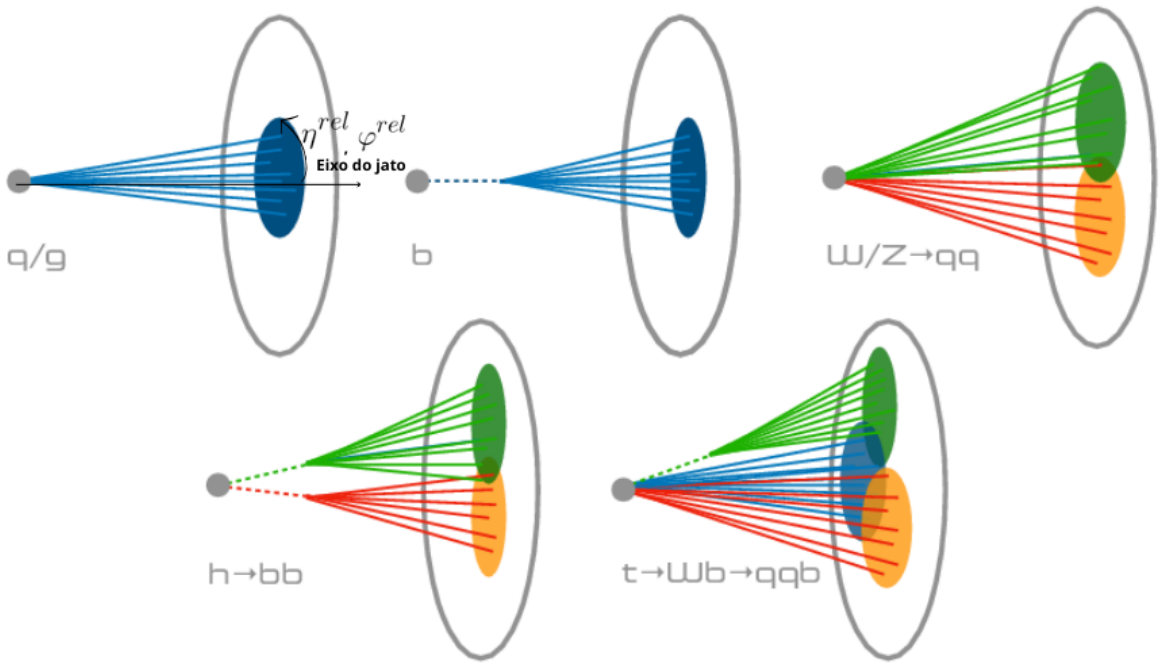

Figura 3.7: Representação Esquemática de Jatos em Colisões de Partículas, e Seus Jatos Após o Decaimento. Modificada. Fonte: [6]

Nesse contexto, as redes neurais têm se mostrado ótimas ferramentas. Esses modelos são capazes de processar grandes volumes de dados e identificar padrões sutis que são difíceis de identificar com algoritmos tradicionais.

4 Rede Neural Artificial

As técnicas de *Machine Learning* (ML) podem ser aplicadas em diversas áreas. Ao invés de desenvolver manualmente um programa para cada tarefa específica, adota-se uma abordagem onde se utiliza uma amostragem de dados com muitos exemplos que indicam a saída desejada para cada entrada possível. Um algoritmo de ML recebe esses exemplos e gera um modelo capaz de realizar a tarefa. Esse modelo pode ser bastante diferente de um programa convencional, frequentemente consistindo de centenas a milhões de parâmetros.

Após o treinamento, o modelo pode prever a saída mesmo com novos conjuntos de dados. Além disso, nas condições atuais, com a redução dos custos computacionais como o uso de computação em nuvem, a disponibilidade de *Big Data* e o processamento em GPUs, são ideais para impulsionar o avanço do *Machine Learning*.

As redes neurais são uma subárea da inteligência artificial (IA). Para a classificação da identificação do bóson W, será utilizada essa técnica, que é uma das disponíveis na área de IA. O foco principal do trabalho será na arquitetura de rede neural de *Multilayer perceptron* (MLP). No entanto, como mostrado na Figura 4.1, existem outros tipos de redes neurais, como: redes neurais convolucionais (CNN), redes neurais recorrentes (RNN), rede adversária generativa(GANs) entre outras.

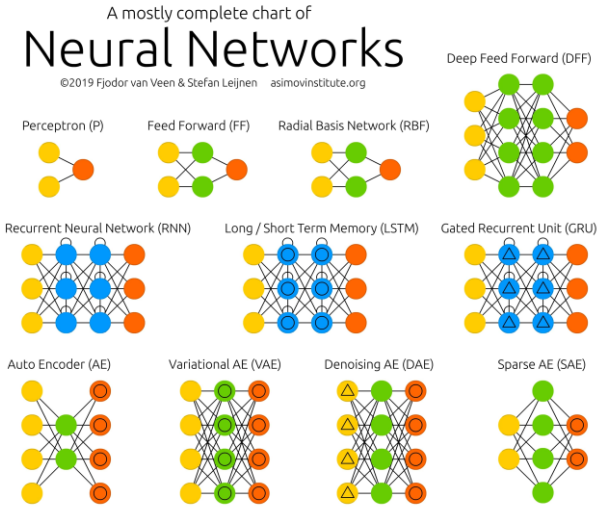

Figura 4.1: Arquitetura de Redes Neurais. Fonte:[7]

As CNNs são amplamente utilizadas em problemas de visão computacional, como classificação e detecção de imagens. Elas são especialmente eficientes na identificação de padrões espaciais, sendo capazes de processar grandes volumes de dados visuais com alta performance. Já as GANs são empregadas na geração de novos dados, como imagens, músicas ou textos. As GANs consistem em duas redes: uma geradora, responsável pela criação de dados novos, e uma discriminadora, que avalia a qualidade desses dados gerados. As RNNs são ideais para o processamento de dados sequenciais, como séries temporais, texto ou fala. Sua principal vantagem é a capacidade de "lembrar"informações de entradas

anteriores, o que as torna adequadas para tarefas como reconhecimento de fala e previsão de séries temporais. Por fim, as MLPs, que serão utilizadas neste trabalho, são redes neurais *feedforward* tradicionais, ideais para dados estruturados, como os encontrados em planilhas ou bancos de dados. As MLPs são amplamente aplicadas em problemas de classificação e regressão, especialmente quando os dados podem ser representados como vetores.

4.1 MLP

O modelo de aprendizagem supervisionada é o modelo mais comum e muitas vezes o mais utilizado em redes neurais. Sua concepção se iniciou na década de 1940, com o estudo do cérebro, no qual tentaram descrever um neurônio por meio de modelos matemáticos. Com esse trabalho inicial foi possível a criação da ideia do perceptron (figura 4.2).

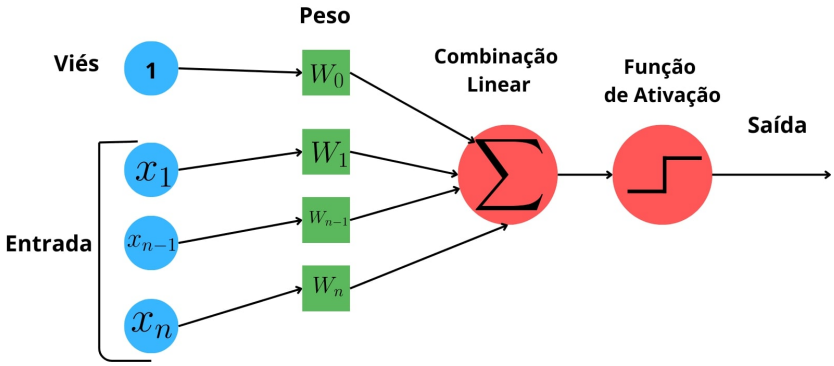

Figura 4.2: Modelo do Perceptron.

Porém, não se pode fazer uma rede neural apenas com um perceptron (neurônio). Para isso, são necesários vários perceptrons conectados, assim formando uma rede neural.

O perceptron tem três estruturas principais, sendo elas: entrada, núcleo e saída. A entrada recebe as informações (se for a primeira camada, recebe os dados do *dataset*). Cada entrada tem um peso associado, variável para a calibração do modelo. No núcleo, temos a soma ponderada das interações entre as entradas e a multiplicação dos pesos. Quanto à saída, é uma função de ativação que determina se o neurônio será ativado ou não. Após o processamento, a saída pode ser final ou ser transmitida para outro neurônio, que repetirá o mesmo processo. Isso continuará até que os dados cheguem ao neurônio final, resultando em um valor para a saída. Em outras palavras, os dados de entrada são vetores nos quais cada componente está associado a um peso (w) correspondente. Assim, a soma ponderada se dá pela equação:

$$S = \sum_{0}^{i} x_i w_i + b \tag{4.1}$$

Tal que x_i é a linha correspondente do vetor, w_i o peso para cada x_i, e b o viés. Em geral a rede pode ser separada por 3 partes, como mostra a figura 4.3.

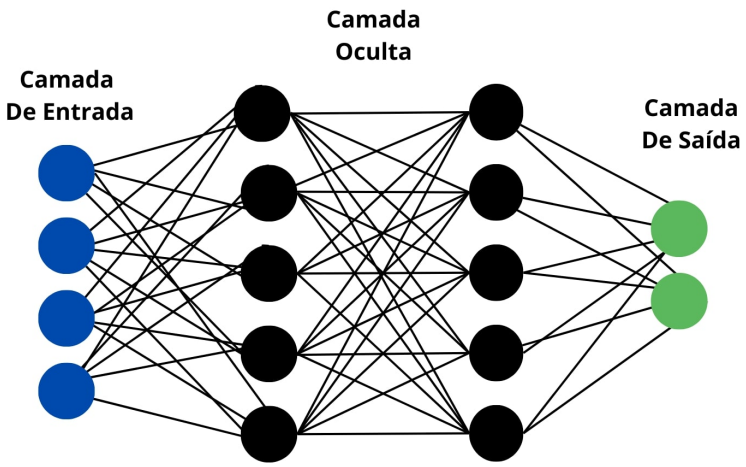

Figura 4.3: Exemplo Rede Neural.

A primeira camada é a camada de entrada, responsável por receber os dados obtidos, ou seja, os dados que serão inseridos no modelo. Por exemplo, se for uma imagem com resolução de 12x12 *pixels*, será necessária uma camada de entrada com 144 neurônios, um para cada *pixel*. A segunda camada, frequentemente chamada de camada oculta ou intermediária, está entre a camada de entrada e a camada de saída. Os neurônios nessa camada recebem informações de outros neurônios. O número de camadas ocultas e a quantidade de neurônios em cada camada dependem da tarefa em questão. Geralmente, quanto mais camadas, mais profundo é o aprendizado da rede. Após passar pelas camadas ocultas, o processamento chega à camada de saída, que é a última camada. Esta, processa o aprendizado obtido e retorna o valor da resposta ($\hat{y}$). É possível ter mais de um neurônio na camada de saída.

4.1.1 Exemplo Dataset MNIST

Um exemplo clássico é o banco de dados 'MNIST', que contém várias imagens de números escritos à mão . O objetivo é treinar um modelo para identificar esses números. Para este exemplo, é utilizada uma camada de entrada com 784 neurônios, correspondendo à resolução da imagem de 28x28 *pixels*, onde cada neurônio representa um *pixel* da imagem. A camada oculta pode ter um número menor ou maior que 784 neurônios, dependendo do método utilizado. Já a camada de saída precisa ter 10 neurônios, cada um responsável por estimar a probabilidade de o número ser um dígito de 0 a 9.

4.2 Função de Ativação

Uma função de ativação em ML, é uma função matemática que é aplicada em cada perceptron da rede neural, seja em uma camada oculta ou na camada de saída. Ela determina se um neurônio deve ser ativado ou não, com base na entrada ponderada que recebe.

Como é possível observar pela a equação 4.1, os neurônios têm um comportamento de um hiperplano: $y = mx + b$. Se não existisse a utilização de funções de ativação, o modelo seria reduzido a apenas prever modelos de regressão linear, oque seria um desperdício computacional, já que a equação de raiz do erro quadrático médio (RMSE), faz isso com mais facilidade. A função de ativação introduz a não-linearidade na saída de cada neurônio, Com isso o modelo consegue aprender sistemas mais complexos que apenas hiperplanos. Com as funções de ativação torna-se viável a propagação dos gradientes (o gradiente será tratado melhor na seção 4.3.2), assim sendo possível ter uma atualização nos pesos e viés.

Neste trabalho, serão apresentadas apenas duas das dez funções de ativação mais utilizadas em redes neurais [31]. Começando pela função sigmoide.

4.2.1 Sigmoide

A função de ativação sigmoide, definida pela equação 4.2, transforma entrada entre $(-\infty; \infty)$ para uma entrada no intervalo de $[0, 1]$. Essa função é muito utilizada como o último neurônio em classificação binária, já que os resultados da sua probabilidade normalizada é entre 0 e 1.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{4.2}$$

E sua derivada sendo:

$$\sigma'(x) = \frac{e^{-x}}{(1 + e^{-x})^2} = \sigma(x)(1 - \sigma(x)) \tag{4.3}$$

Plotando a equação 4.2, é possível ver o seu comportamento na figura 4.4

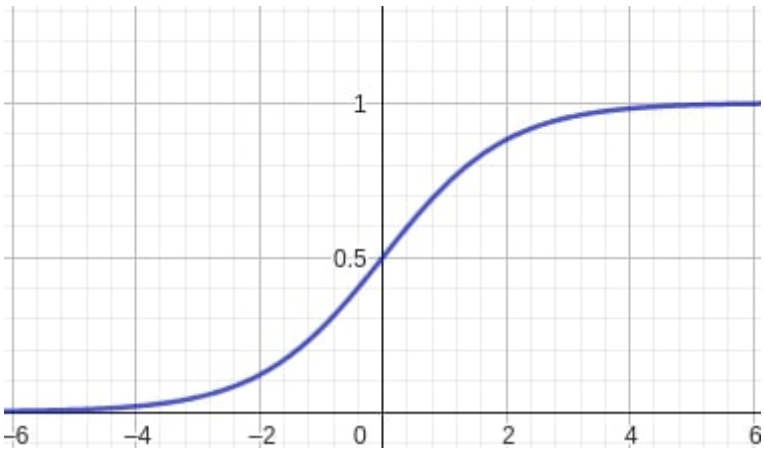

Figura 4.4: Função Sigmoide.

A maior limitação do uso da função sigmoide ocorre em redes neurais profundas (*deep learning*), devido à sua tendência à saturação, o que resulta em problemas com gradientes muito pequenos (o conceito de gradiente será explicado na seção 4.3.2). Uma alternativa

viável é substituí-la pela função *Rectified Linear Unit* (ReLU).

4.2.2 ReLU

ReLU é a função de ativação mais utilizada atualmente quando se envolve *deep learning*, devido a seu desempenho superior durante o treinamento. Sua equação é simples e definida como:

$$ReLU(x) = \sigma(x) = \max(0, x) \tag{4.4}$$

E derivada :

$$\sigma'(x) = \begin{cases} 0, & \text{se } x \leq 0 \\ 1, & \text{se } x \geq 0 \end{cases} \tag{4.5}$$

Essencialmente, o ReLU funciona como uma função identidade ($b = 0$), retornando zero para todos os valores negativos (ver figura 4.5). A sua faixa de saída é $[0, \infty)$.

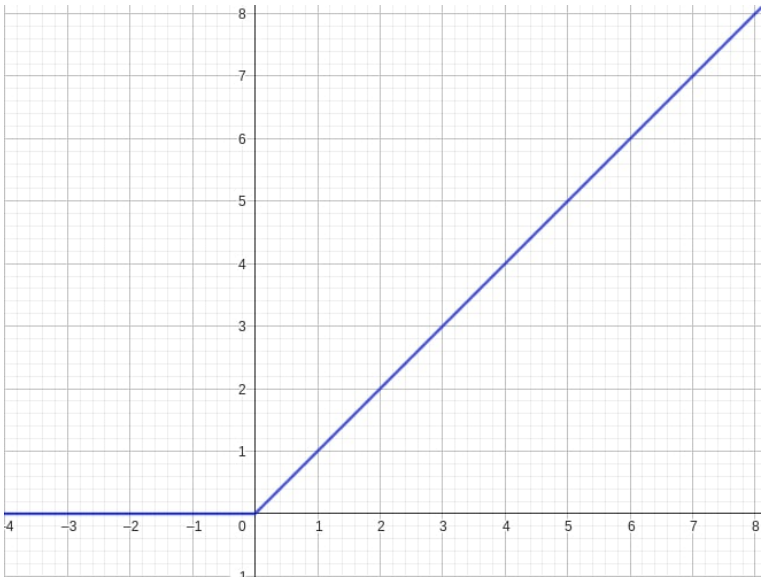

Figura 4.5: Função ReLU.

Uma das principais vantagens da ReLU é sua capacidade de introduzir um zero absoluto, permitindo ativações eficazes nas camadas ocultas e aumentando a diversidade representacional da rede. Porém a sua desvantagem é que para valores negativos os neurônios ficaram desligados para sempre, já que o ReLU não tem mecanismos para reverter.

4.3 Como a Rede Aprende

4.3.1 Função de Custo

As funções de custo, também conhecidas como funções de perda, são utilizadas para avaliar o desempenho de um modelo em relação ao seu conjunto de dados de treinamento, comparando-o com os dados reais esperados.

Para exemplificar, uma função de custo comumente utilizada em regressão linear, é erro quadrático médio (MSE), representado por $J(w, b)$:

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})^2 = \frac{1}{2m}\sum_{i=0}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2 \tag{4.6}$$

Onde $\hat{y}$ é o valor de saídas, ou seja, o valor previsto pelo modelo; y ou $f_{w,b}(x^{(i)})$ o valor real da função; w os pesos; b o viés, e m o tamanho do conjunto de dados [32]. No trabalho, foi utilizada a função conhecida como *Binary Cross Entropy* (BCE), já que o objetivo é fazer uma classificação (1,0), sendo 1 para a representação do Bóson W, e 0 o ruído de fundo.

$$J(w,b) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \times log(\hat{y}^{(i)}) + (1 - y^{(i)}) \times log(1 - \hat{y}^{(i)}) \tag{4.7}$$

É possível observar que a função BCE (eq: 4.7), tem um alto grau de "punição", já que se o modelo prever erroneamente $\hat{y} = 1$, sendo que o alvo é $y = 0$. O resultado da função de custo ficará $-1/m \times log(0)$, e $log(0)$ tende ao infinito, o que mostra que o modelo tem um auto grau de punição.

O objetivo final do modelo é minimizar ao máximo a função de custo $J(b, w)$ ajustando os pesos (w) e o viés (b), mas para isso ser possível é preciso utilizar métodos de otimização, como o do gradiente descendente.

4.3.2 Gradiente Descendente

O gradiente descendente tem como objetivo a otimização, utilizado para encontrar os mínimos de funções de custo. Como dito na seção 4.3.1, para cada entrada há uma resposta esperada y. Após cada época (ciclo de treinamento do modelo), há uma função de custo responsável por atualizar os pesos do modelo. O conceito de gradiente($\nabla\cdot$) é um vetor que aponta a direção de maior crescimento de uma função. E multiplicando por -1 é possível obter a direção para diminuir a função de custo.

Em uma função simples, basta calcular a derivada parcial da equação 4.6, resultando em:

$$\frac{\partial}{\partial w}J(w,b) = \frac{1}{m}\sum_{i=1}^{m}(J_{w,b}(x^{(i)}) - y^{(i)}) \tag{4.8}$$

$$\frac{\partial}{\partial b}J(w,b) = \frac{1}{m}\sum_{i=1}^{m}(J_{w,b}(x^{(i)}) - y^{(i)}) \tag{4.9}$$

Após várias iterações, é possível chegar a um peso w mais adequado, demonstrando que o modelo "aprendeu". Em aplicações mais complexas, não se utiliza apenas um neurônio, mas sim centenas ou milhares.

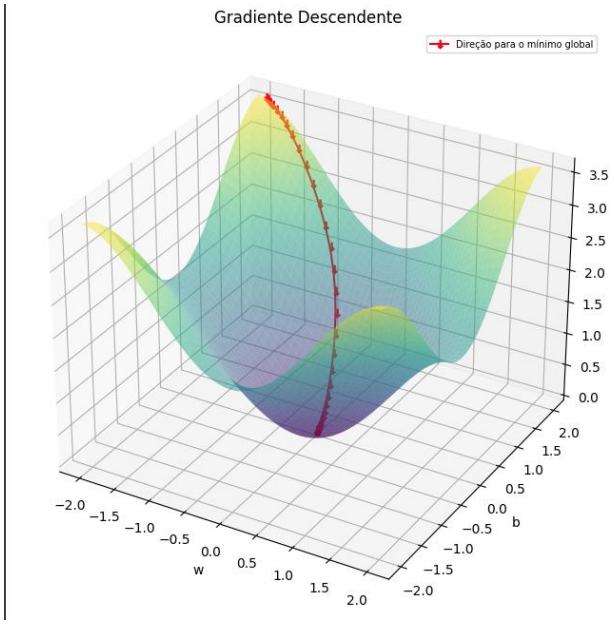

Figura 4.6: Gráfico de uma função de custo com vários mínimos locais.

É possível encontrar vários mínimos locais, e o gradiente descendente frequentemente converge para um mínimo local ao contrário do mínimo global, na figura 4.6 é possível observar que a função de custo conseguiu encontrar o mínimo global. Porém, para sistemas mais complexos, a fim de lidar com essa situação, foram desenvolvidos diversos métodos para mitigar esse tipo de erro e a lentidão do gradiente descendente. Neste trabalho, será abordado apenas dois dos dez otimizadores mais utilizados [33].

4.3.2.1 SGD Com Momento

O *Stochastic Gradient Descent*(SGD). Esse método combina o gradiente descendente com uma abordagem estatística clássica conhecida como "*random walk*", que se assemelha ao caminhar de um bêbado. Em cada época do modelo, ele se move em direção a um mínimo local de forma aleatória, o que pode resultar na descoberta de um mínimo local diferente, possivelmente melhor ou pior que o anterior, porém caso a função tenha pontos planos, o SGD acaba tendo uma certa demora para convergir para o mínimo global, assim, foi criada a técnica de SGD com Momento:

$$\begin{aligned} v_{t+1}^{w} &= \boldsymbol{\gamma} v_{t-1}^{w} + \alpha \nabla_{w} \cdot \mathbf{J}(\mathbf{w}, \mathbf{b}) \\ w_{t+1} &= w_t - v_{t+1}^{w} \\ v_{t+1}^{b} &= \boldsymbol{\gamma} v_{t-1}^{b} + \alpha \nabla_{b} \cdot \mathbf{J}(\mathbf{w}, \mathbf{b}) \\ b_{t+1} &= b_t - v_{t+1}^{b} \end{aligned} \tag{4.10}$$

Para cada atualização de passo, uma fração do vetor $\boldsymbol{\gamma}$ é atualizado, tendendo a mover para a direção correta mais rapidamente. v é o vetor momento, α a taxa de aprendizado, e $\nabla \cdot \mathbf{J}(\theta)$ o gradiente da função de custo com os parâmetros de peso e viés.

O momento é análogo ao lançamento de uma bola, como se estivesse jogando uma bola de uma montanha, a bola vai acumular momento, até chegar na velocidade terminal, ou seja $\boldsymbol{\gamma} < 1$. Essa analogia funciona perfeitamente para explicar o SGD com momento, pois ele aumenta nas dimensões onde o gradiente aponta para a mesma direção. Assim reduzindo a atualização de dimensões as quais os gradientes não mudam de direção, essa é a sua maior vantagens contra o SGD, já que tende a convergir mais rápido.

4.3.2.2 ADAM

O otimizador *adaptive moment estimation* (ADAM) é um dos algoritmos de otimização mais utilizados em *deep learning*. É uma mistura entre os otimizadores RMSprop e Adadelta. Uma das suas vantagens é ter um gasto computacional menor que SGD, com isso é possível um modelo treinado em menos tempo. Com ADAM, a primeira ordem de gradientes conhecido como momento é mantida por uma média, muito similar ao SGD com momento. Isso facilita o aprendizado e suaviza oscilações de treinamento.
A equação do momento é dada por :

$$\begin{aligned} m_t^w &= \beta_1 m_{t-1}^w + (1-\beta_1)\nabla_w \cdot \mathbf{J}(\mathbf{w}, \mathbf{b}) \\ m_t^b &= \beta_1 m_{t-1}^b + (1-\beta_1)\nabla_b \cdot \mathbf{J}(\mathbf{w}, \mathbf{b}) \end{aligned} \tag{4.11}$$

Onde β_1 é o coeficiente de decaimento da média móvel, geralmente definido como 0,9, m_{t-1} é a média móvel do momento no tempo $t-1$, e g_t é o gradiente no tempo t. Porém não se utiliza apenas os gradientes de primeira ordem, mas também os de segunda, assim tendo uma adaptação da taxa de aprendizado.

$$\begin{aligned} v_t^w &= \beta_2 v_{t-1}^w + (1-\beta_2)\nabla_w \cdot \mathbf{J}(\mathbf{w}, \mathbf{b}) \\ v_t^b &= \beta_2 v_{t-1}^b + (1-\beta_2)\nabla_b \cdot \mathbf{J}(\mathbf{w}, \mathbf{b}) \end{aligned} \tag{4.12}$$

Em outras palavras, m_t pode ser considerado como a média dos gradientes, e v_t a sua variância não centralizada. Nos primeiros passos, os momentos m_t,e v_t, tendem a ir para zero no inicio do treinamento, para isso ser evitado é também utilizado uma correção de viés:

$$\begin{aligned} \hat{m_t^w} &= \frac{m_t^w}{1-\beta_1^t} \\ \hat{v_t^w} &= \frac{v_t^w}{1-\beta_2^t} \\ \hat{m_t^b} &= \frac{m_t^b}{1-\beta_1^t} \\ \hat{v_t^b} &= \frac{v_t^b}{1-\beta_2^t} \end{aligned} \tag{4.13}$$

Agora com as médias ajustadas é possível a atualização dos parâmetros:

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{v}_t{}^w + \epsilon}} \hat{m}_t{}^w \tag{4.14}$$
$$b_{t+1} = b_t - \frac{\alpha}{\sqrt{\hat{v}_t{}^b + \epsilon}} \hat{m}_t{}^b$$

Tal que w_t é o peso, b_t o viés, α a taxa de aprendizado, ϵ termo para estabilização. Esta correção evita a divisão por zero, e β_1 e β_2 são os fatores de decaimento dos momentos de primeira e segunda ordem. respectivamente.

4.3.3 Taxa de Aprendizado

Na figura 4.6, é possível ver o vetor vermelho que representa a taxa de aprendizado. Quanto menor for essa taxa, mais lento será o aprendizado do modelo. Em outras palavras, uma taxa de aprendizado menor resulta em uma descida mais suave na derivação, levando mais tempo para alcançar um mínimo local. Essa taxa é dada pelas equações.

$$w = w - \alpha \frac{\partial}{\partial w} J(w, b) \tag{4.15}$$

$$b = b - \alpha \frac{\partial}{\partial b} J(w, b) \tag{4.16}$$

Onde w é o peso, b é o viés, e α é a taxa de aprendizado.

Na figura 4.7 é possível observar a diferença entre cada tipo de taxa de aprendizado, indo desde muito pequena, ideal, e muito grande. Quando não se sabe qual taxa de aprendizado utilizar no modelo, é prudente evitar uma taxa muito grande, pois essa pode resultar no fenômeno de "desaprendizado". Ao contrário de convergir para o mínimo local, a função de custo pode aumentar continuamente.

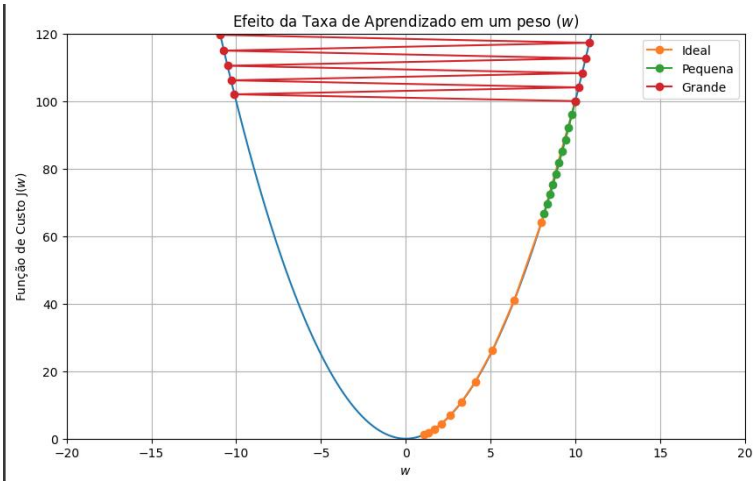

Figura 4.7: Exemplo de várias taxas de aprendizado, e seu comportamento em um neurônio.

Isso ilustra uma situação em que a escolha inadequada da taxa de aprendizado pode acarretar em problemas. No exemplo dado, com um peso inicial de $w = 1$ e uma taxa de aprendizado $\alpha = 10$, supondo que a derivada parcial da função de custo $J(w, b)$ seja $0,5$, o novo valor do peso seria -4. Isso representa uma mudança muito grande, levando a uma atualização excessiva, e a função de custo acaba aumentando ao invés de diminuir.

Em contrapartida, com uma taxa de aprendizado menor ou ideal, como $0,8$, a atualização do peso resultaria em uma nova função de custo de $0,6$, aproximando-se do mínimo local, observe que esse é um exemplo ilustrativo.

Não existe uma regra universal para a escolha da taxa de aprendizado. Muitas vezes, valores comuns como 0.001 são adotados por padrão, mas essa escolha pode variar consideravelmente dependendo do problema específico, e da arquitetura da rede neural utilizada. É comum realizar experimentações e ajustes para encontrar a taxa de aprendizado mais adequada para um determinado problema.

4.3.4 Backpropagation

O *backpropagation* é fundamental no treinamento de redes neurais, permitindo que o modelo aprenda com exemplos rotulados ao ajustar os pesos para minimizar a diferença entre previsões e valores reais. Embora criado na década de 1970, ganhou destaque em 1986 com o artigo "*Learning representations by back-propagation errors*"[34], demonstrando sua eficácia em resolver problemas de forma mais rápida para a época. Essencialmente, o algoritmo de *backpropagation* é uma técnica para calcular derivadas de maneira eficiente.

O *backpropagation* pode ser dividido em duas partes:

1. *Forward*
2. *Backward*

A etapa de *Forward* é a primeira fase do algoritmo, na qual os dados de entrada são propagados pela rede, permitindo assim o cálculo da saída prevista ($\hat{y}$). Já o *Backward* envolve o cálculo do gradiente da função de custo na última camada (camada de previsão) e sua retropropagação até a primeira camada escondida da rede. Isso é feito utilizando a regra da cadeia para os gradientes, o que resulta na atualização dos pesos da rede.

4.3.4.1 Forward

A utilização do *Forward* é propagar a rede por meio de multiplicação de vetores e ativações até chegar na camada de saída. O resultado da propagação $\sigma(X^T \odot w^{(1)})$ é considerado a saída do neurônio, que receberá os valores em um vetor $a^{(i)}$, onde i é o número da camada escondida. Multiplicando o valor recebido de $a^{(1)}$ com o peso $w^{(2)}$, e passando pela função de ativação, temos o resultado final da previsão. Para melhorar o aprendizado é passado pelo *Backward* que será discutido na próxima subseção.

4.3.4.2 *Backward*

O processo de *Backward*, é responsável pelo cálculo dos gradientes de perda em relação aos pesos da rede neural. Isso permitirá ajustar os pesos da rede durante o treinamento para minimizar a perda.

Quando a rede chegar na última camada, é feita uma verificação. Se realmente o resultado previsto é resultado real, e como discutido na seção 4.3.1, isso se dá pela função de custo. Mas para melhorar o aprendizado, também foi visto na seção 4.3.2, que se usam os gradientes. A equação para a última camada, onde começa a retro-propagação para ajustar os pesos, é dada por:

$$\delta^{(n)} = \nabla \cdot \mathbf{J}(\theta)\sigma'(z^{(n)}) \tag{4.17}$$

Após sair da última camada, a retro-propagação continua. Porém com a equação 4.17 um pouco modificada, ficando:

$$\delta^{(n-1)} = \left(w^{(n)}\right)^T \delta^{(n)}\sigma'(z^{(n-1)}) \tag{4.18}$$

Nas equações acima, $z^{(i)}$ representa o valor da camada antes de passar pela função de ativação. Esse processo é repetido iterativamente durante o treinamento da rede para minimizar a perda e melhorar suas previsões. Portando, o algoritmo de *backpropagation* calcula o gradiente da função para cada peso da rede de trás para frente até a primeira camada ocultada. Após o calculo dos gradientes, é utilizado o gradiente descendente para a sua atualização.

5 Aplicação de Machine Learning

Para o treinamento do modelo, foi utilizado o conjunto de dados *HLS4ML LHC Jet dataset* [35] com 30 partículas. Esses dados são feitos por meio de simulações de partículas com momento transversal (p_T) de jato de 1 TeV, que são produzidos por colisões próton-próton de 13 TeV. Os jatos são agrupados a partir da reconstrução das partículas individuais, utilizando o algoritmo anti-k_t com o parâmetro $R = 0,8$. Para esse *dataset* foram selecionadas apenas cinco categorias de partículas: Quarks leves[7], Quark top, Gluôn, Bóson W e Bóson Z. O *dataset* é dividido em 3 conjuntos de dados, sendo:

1. Lista de 16 jatos relacionados em alto nível de *features* (HLFs), sendo elas: $\Sigma \log(z); C_1^0; C_1^1; C_1^2; C_2^1; C_2^2; D_2^1; D_2^2; D_2^{(1,1)}; D_2^{(1,2)}; M_2^1; M_2^2; N_2^1; N_2^2;$ massa de jato, e multiplicidade. Onde: C, D, N, são observáveis "de forma", e M "de massa". Para uma melhor compreensão das subestrutura de jato, pode ser obtido na referência [36]

2. Imagens de 100×100 *pixel*. a "imagem" é derivada de um quadrado com pseudo-rapidez e distâncias azimutes de $\Delta\eta = \Delta\varphi = 2R$, centralizado no eixo do jato. Esse agrupamento de 100×100 pode ser considerado análogo ao tamanho a célula do ECAL, porém muito mais grosseiro que a resolução obtida no equipamento. Cada *pixel* é preenchido com a soma escalar dos p_T das partículas naquela região, onde cada imagem é representada com as 30 partículas com maior energia.

3. Uma lista com até 30 partículas, onde cada partícula é representada por 16 *features*, calculadas a partir do quadrimomento da partícula. Essas *features* incluem: as três coordenadas cartesianas do momento (p_x, p_y, p_z), energia absoluta E, p_T, pseudo-rapidez η, ângulo azimutal φ, distância do centro do jato $\Delta R = \sqrt{\Delta\eta^2 + \Delta\varphi^2}$ e as energias relativa: $E^{rel} = E^{particle}/E^{jet}$; $p_T^{rel} = p_T^{particle}/p_T^{jet}$, representando a relação entre a quantidade de partículas e a quantidade do jato; as coordenadas relativas são dadas por : $\eta^{rel} = \eta^{particle} - \eta^{jet}$; $\varphi^{rel} = \varphi^{particle} - \varphi^{jet}$ definido em relação ao eixo do jatos $\cos\theta$ e θ^{rel} tal que $\theta^{rel} = \theta^{particle} - \theta^{jet}$, calculado em relação ao eixo do jato e às coordenadas relativas η e φ da partícula, após aplicar uma transformação de Lorentz adequada. Sempre que menos de 30 partículas são reconstruídas, a lista é preenchida com zeros.

Para esse trabalho, foi utilizado o item 3 da descrição dos dados, ou seja será, utilizado a lista com as 30 partículas, para uma melhor visualização, na figura 5.1 é possível observar o plot para cada uma das 16 *features*.

[7] up , down e strange

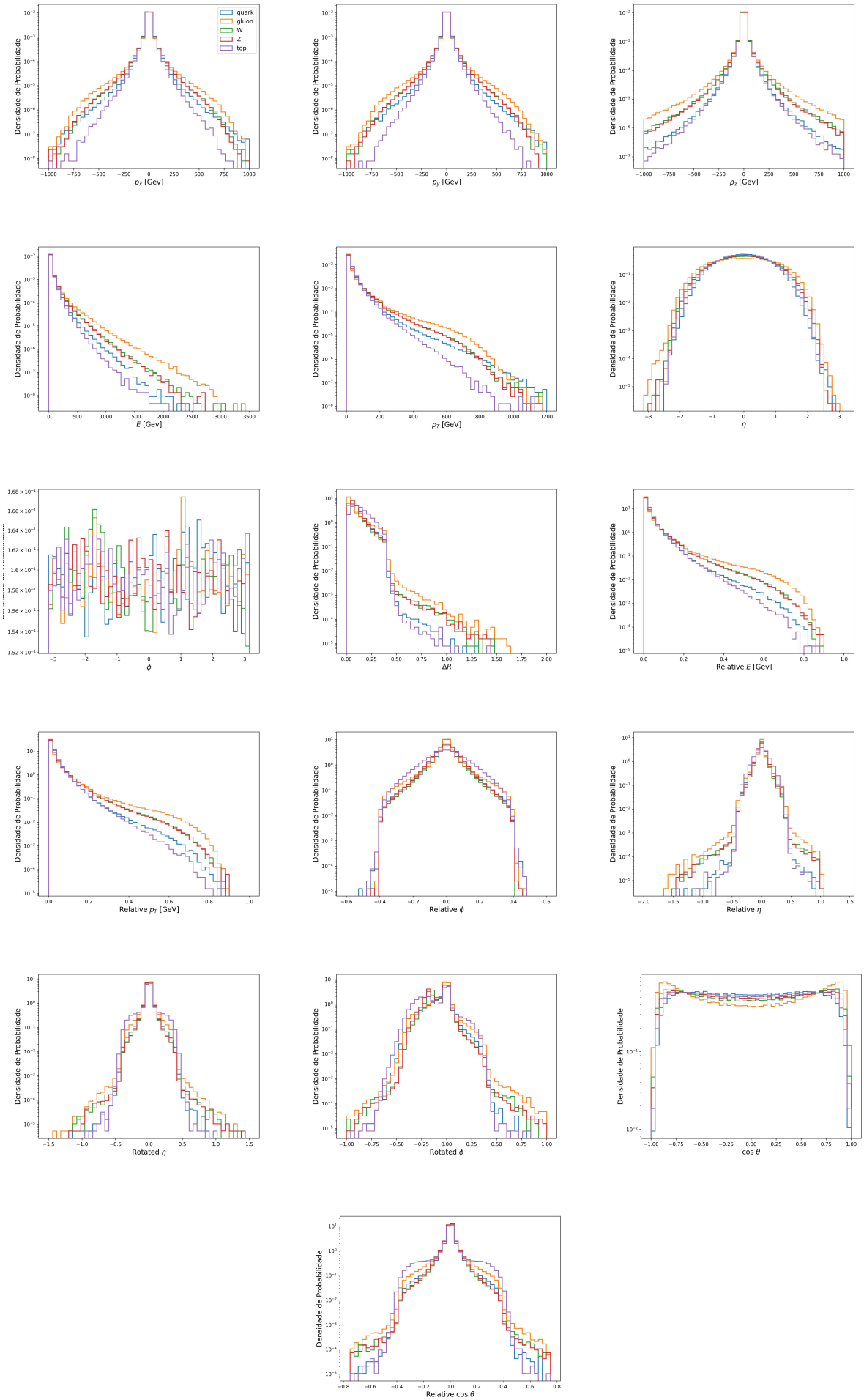

Figura 5.1: Distribuições das *features* descritas no item 3 para as 30 partículas de maior p_T em cada jato.

5.1 Tratamento de Dados

Para o tratamento de dados, foi utilizada a linguagem de programação Python 3 [37]. Inicialmente, os dados compactados nos arquivos **hls4ml_LHCjet_30p_train.tar.gz** e **hls4ml_LHCjet_30p_val.tar.gz**, foram descompactados, permitindo a observação de múltiplos arquivos no formato **.h5**, organizados em 63 arquivos para treino e 27 para teste. Esse tipo de arquivo é um contêiner de dados hierárquico que facilita o armazenamento e a manipulação de grandes conjuntos de dados.

Para unificar todos esses arquivos em um único[8], foi utilizada a biblioteca **h5py** [38], que permite a leitura e manipulação dos arquivos **.h5**. Todos os arquivos foram lidos e concatenados em um array usando a biblioteca **numpy** [39], consolidando os dados em uma única estrutura e facilitando o processo de tratamento dos dados.

Após a concatenação, foram removidas as partículas que possuem energia nula. Em seguida, as partículas foram separadas em suas respectivas categorias: Quarks, Quark Top, Glúons, Bósons W e Bósons Z. Após a separação, todas as categorias foram concatenadas novamente em um único array, com a adição de uma nova coluna chamada "Bóson W". Esta coluna indica se a partícula é um Bóson W (valor 1) ou não (valor 0), (ver figura 5.2), facilitando a identificação e o uso dessas informações no treinamento de modelos posteriores. Assim, tornou-se possível a criação de um modelo baseado em classificação binária.

[8]Unificando todos os arquivos de teste em um único arquivo de teste e o mesmo para o de treino

p_x [GeV]	2.716500	-3.137305	-20.227770	-17.873404
p_y [GeV]	-1.497397	-23.386082	-7.045567	-21.527420
p_z [GeV]	-0.282358	-3.083958	-1.607983	12.576772
E [GeV]	3.117762	23.796682	21.479950	30.677078
Relativo E [GeV]	0.002861	0.023649	0.021478	0.026453
p_T [GeV]	3.101866	23.595583	21.419680	27.980143
Relativo p_T [GeV]	0.002874	0.023617	0.021664	0.026831
η	-0.090903	-0.130331	-0.075000	0.435584
Relativo η	0.023891	-0.041595	-0.094999	-0.027211
rotação η	0.121607	-0.030947	-0.076160	0.014791
ϕ	-0.503782	-1.704153	-2.806423	-2.263719
Relativo ϕ	0.119099	-0.031060	-0.071893	0.084461
rotação ϕ	-0.017555	0.007268	0.019024	-0.172302
ΔR	0.121472	0.051912	0.119136	0.088736
cos θ	-0.090654	-0.129598	-0.074860	0.409977
Relativo cos θ	0.121011	-0.030937	-0.076013	0.014789
Boson W	1.000000	1.000000	0.000000	0.000000

Figura 5.2: Exemplificação dos dados apos o tratamento

Após a concatenação final, o *dataset* de treino ficou com dezoito milhões de linhas e o de teste com sete milhões. Os dados modificados foram então salvos como um tensor do **PyTorch** [40] no formato **.pt**, o que oferece uma economia significativa de memória, tanto em disco quanto em RAM. Para garantir a replicabilidade do trabalho, todos os passos foram implementados no código, que está disponível para consulta em [41], e os dados já tratados, disponíveis em [42]. O código é criado por meio de funções que encapsulam cada etapa do tratamento, desde a leitura dos arquivos até o salvamento dos dados finais, tornando o processo reprodutível e acessível para futuros estudos ou aprimoramentos.

5.2 Construção do Modelo

Inicialmente, foi testada a normalização dos dados utilizando a biblioteca **StandardScaler** do **sklearn** [43]. Essa normalização visa garantir que as variáveis estejam na mesma escala, ajustando os dados para que tenham uma média de 0 e um desvio padrão de 1. Esse processo assegura que todas as variáveis contribuam igualmente para o modelo, evitando que variáveis com magnitudes maiores dominem as menores. A fórmula utilizada pelo **StandardScaler** é dada por:

$$x_{\text{padrão}} = \frac{x - \mu}{\sigma} \tag{5.1}$$

Onde x é o valor antes da padronização, μ a média, e σ o desvio padrão.

No entanto, após a normalização, observou-se que não houve mudanças significativas no desempenho do modelo, apenas um aumento na complexidade do código. Portanto, após os testes iniciais, a normalização foi descartada e os dados foram utilizados conforme apresentado na Figura 5.2.

Como mencionado na seção anterior, o trabalho tem como foco a utilização da técnica de *Multilayer perceptron* (MLP), esse modelo é considerado um dos mais simples no conceito de ML, e pode ser empregado nas tarefas de aprendizagem de dados tabulares, porém é exigido um esforço maior para encontrar os hiperparâmetros certos. Com o objetivo de tentar encontrar uma melhor arquitetura, foi travado a taxa de aprendizagem em $0,001$, e as épocas em 150 (ou seja o modelo será treinado com os dados 150 vezes), utilizando a função **shuffle** do **PyTorch** para embaralhar os dados a cada época, assim tentando evitar o *overfitting*. A Função de ativação nas camadas ocultas utilizada foi a ReLU, e na camada de saída a sigmoide. A função de custo utilizada foi a **BCELoss** dada pela equação 4.7. Com esses dados fixos, já é possível treinar os modelos para observar qual será a melhor arquitetura e otimizador. Foram treinados três modelos, variando suas camadas ocultas, como mostrado na figura 5.3.

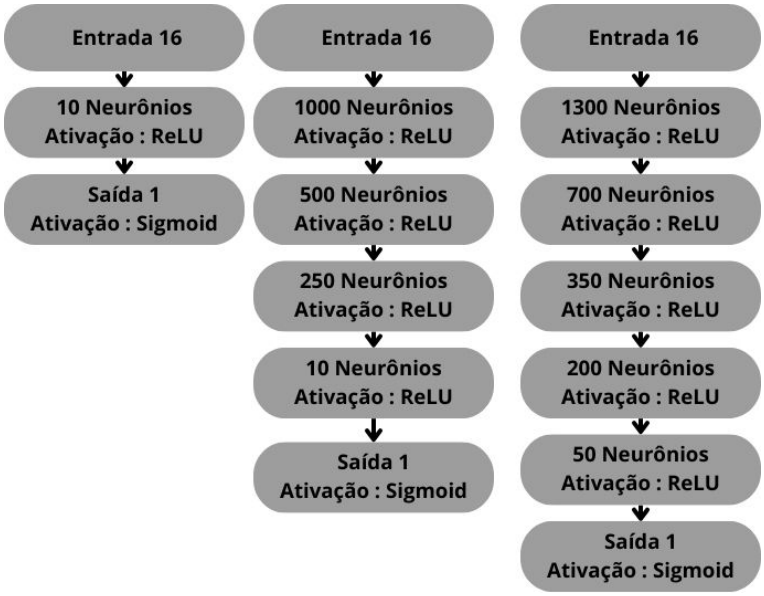

Figura 5.3: Arquiteturas com diferentes camadas, neurônios.

Com uma visão geral da modelagem dos modelos, agora é possível criar o código. Utilizou-se a biblioteca **PyTorch** e suas sub-bibliotecas para a construção dos modelos.Foi selecionado a biblioteca pois permite aproveitar o poder computacional das placas de vídeo (GPUs). Como as redes neurais envolvem operações intensivas de multiplicação de matrizes, as GPUs são eficientes nesse tipo de tarefa. Além disso, considerando o uso de dezoito milhões de dados, o uso da GPU acelera significativamente o processo, tornando-o muito mais rápido.

Primeiramente, os dados provenientes do tensor do **PyTorch** foram organizados em um *dataset* customizado, otimizado para integração com o **PyTorch**. Esses dados foram

então divididos em conjuntos de treino e teste.

O **DataLoader** foi utilizado para carregar os dados em lotes, com um *batch size* de 128, isso significa que, do total de dezoito milhões de linhas, cada **DataLoader** processará apenas 128 linhas por vez, até que todas as dezoito milhões sejam processadas. Esse processo é aplicado tanto aos dados de treino quanto aos dados de teste. Desta forma, a memória utilizada durante o treinamento é reduzida, e os pesos finais de cada época são ajustados com base na média dos gradientes de cada *batch*.

Após o tratamento inicial dos dados, é possível começar a definir a arquitetura do modelo utilizando a biblioteca **nn** do **PyTorch**. Para isso, cria-se uma classe que configurará a arquitetura e o tipo do modelo. No caso específico de MLP, a função **nn**.**Sequencial**() é utilizada dentro do construtor (método de inicialização da classe) para definir a sequência de camadas do modelo. Essa abordagem permite a construção modular e organizada da arquitetura, facilitando a implementação e o ajuste do modelo.

Após a criação da classe, o próximo passo é instanciar o modelo em uma variável e verificar o *device* em uso, seja CPU ou GPU. Em seguida, é necessário utilizar a biblioteca **optim** do **PyTorch** para escolher o otimizador a ser utilizado. Neste caso, foram escolhidos dois otimizadores para testar, **Adam** e **SGD** (utiliza-se um de cada vez). É necessário passar a função **parameters()** da variável onde o modelo foi instanciado, e definir a taxa de aprendizagem, que foi configurada como 0,001. Além disso, utilizando a biblioteca **nn**, deve-se selecionar a função de custo adequada, que, neste caso, foi a **BCELoss**.

Finalmente, é necessário implementar as funções de treino e teste para o modelo, e realizar o treinamento pelo número de épocas desejado. No trabalho em questão, foram realizadas 150 épocas. O código utilizado para essas etapas está descrito e pode ser encontrado em [41].

5.3 Resultados

A curva *Receiver Operating Characteristic* (ROC) é uma ferramenta utilizada para avaliar o desempenho de um modelo de classificação. Ela ilustra a relação entre a *true positive rate* (TPR) e *false positive rate* (FPR).

A TPR, também conhecida como sensibilidade ou *recall*, representa a proporção de positivos reais corretamente identificados pelo modelo. Se da pela equação:

$$\text{TPR} = \frac{\text{Verdadeiros Positivos}}{\text{Verdadeiros Positivos} + \text{Falsos Negativos}} \tag{5.2}$$

A FPR indica a proporção de negativos que foram incorretamente classificados como positivos. É calculada pela equação:

$$\text{FPR} = \frac{\text{Falsos Positivos}}{\text{Falsos Positivos} + \text{Verdadeiros Negativos}} \tag{5.3}$$

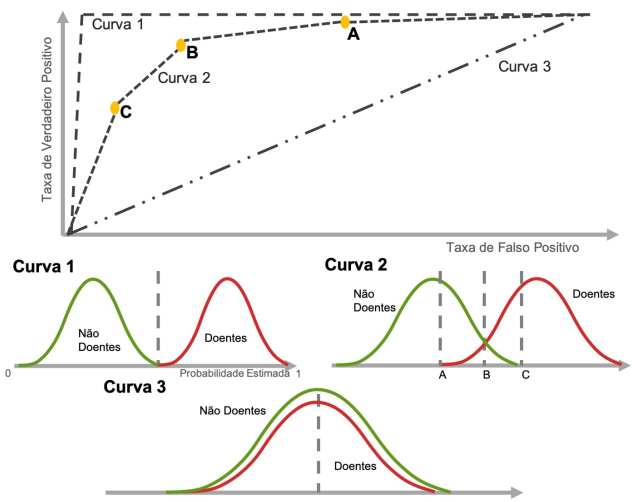

Figura 5.4: Exemplos do comportamento da Curva ROC. Fonte [8]

A curva plota TPR versus FPR; quanto mais distante da diagonal, melhor o desempenho, como é possível observar na figura 5.4.

A AUC é a área sob essa curva ROC. Ela é uma medida quantitativa de como o modelo está se saindo em termos de discriminação entre as classes. AUC pode ser calculada por meio de integração contínua [44].

$$AUC(f) = \int_{-\infty}^{\infty} TPR(FPR)\, dFPR = \int_{0}^{1} ROC(u)du \tag{5.4}$$

Para gerar a curva ROC e comparar os modelos, foi utilizada a biblioteca **matplotlib** [45] em conjunto com o módulo **metrics** da biblioteca **sklearn**.

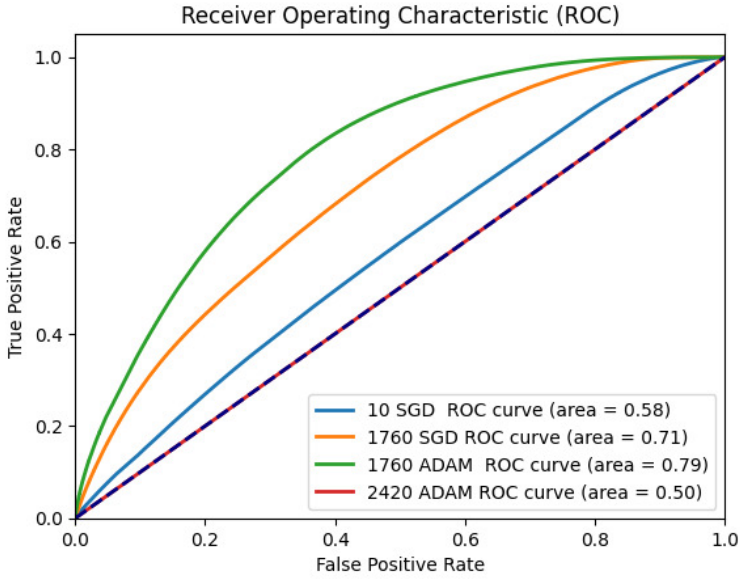

Figura 5.5: Curva ROC comparando o desempenho de modelos com diferentes neurônios e otimizadores (SGD vs. ADAM).

A figura 5.5 ilustra a curva ROC, e revela como o desempenho dos modelos MLP varia com a complexidade da arquitetura e o otimizador utilizado. Na Tabela 3, é possível observar de forma clara as diferenças entre os modelos.

Tabela 3: Desempenho dos modelos MLP com diferentes configurações.

Modelo	**Neurônios**	**Otimizador**	**AUC**	**Observação**
Modelo simples	10	SGD	0,58	Desempenho limitado
Modelo intermediário	1760	SGD	0,71	Aumento de complexidade
Modelo intermediário	1760	ADAM	0,79	Superior ao otimizador SGD
Modelo complexo	2420	SGD	0,50	Indica possível overfitting

O modelo mais simples, com 10 neurônios, e otimizador SGD, apresentou uma AUC de 0,58, indicando um desempenho limitado. Com o aumento da complexidade, a AUC subiu para 0,71 utilizando o SGD e para 0,79 com o otimizador ADAM, destacando a superioridade do ADAM. No entanto, o modelo mais complexo, com 2420 neurônios, obteve uma AUC de apenas 0,50, o que sugere um possível *overfitting* e uma perda na capacidade de generalização.

6 Conclusão

A utilização de redes neurais em experimentos como o CMS, tem-se mostrado importante para o avanço das análises no campo da física de partículas, com o aumento da complexidade e do volume de dados gerados pelas colisões no LHC. Essas técnicas permitem a identificação e classificação de partículas, como o bóson W, com uma melhor precisão, automatizando processos que seriam inviáveis de serem realizados manualmente devido à enorme quantidade de eventos registrados.

Os resultados demonstram que a performance dos modelos MLP é fortemente influenciada pela escolha correta de seus hiperparâmetros (taxa de aprendizagem, épocas, batch) e pela arquitetura (neurônios, camadas, otimizadores). Modelos excessivamente complexos, sem técnicas de regularização, podem sofrer de *overfitting*, comprometendo a generalização dos resultados. Em contrapartida, otimizadores modernos, como o ADAM, têm se mostrado eficazes em melhorar a AUC e aumentar a eficiência em termos de tempo e recursos computacionais, aproveitando melhor a infraestrutura disponível e tornando o processamento mais rápido e preciso.

Com a atualização do LHC para *High-Luminosity Large Hadron Collider* (HL-LHC), o número de colisões e a quantidade de dados processados pelo CMS aumentarão consideravelmente. As melhorias implementadas no CMS, como a substituição dos fotodetectores e a atualização do sistema de aquisição de dados, são fundamentais para manter a qualidade das medições e lidar com o volume massivo de informações. Nesse contexto, o uso de técnicas de *machine learning*, como redes neurais, se tornará ainda mais relevante, permitindo uma análise mais rápida e detalhada dos dados e contribuindo para uma maior e melhor compreensão da física de partículas.

Referências

[1] Ewa Lopienska. The cern accelerator complex, layout in 2022. Technical report, 2022.

[2] The CMS Collaboration, Chatrchyan, et al. The cms experiment at the cern lhc. *Journal of Instrumentation*, 3(08):S08004–S08004, August 2008.

[3] Stefan Petschauer and et al. Haidenbauer. Hyperon-nuclear interactions from su(3) chiral effective field theory. *Frontiers in Physics*, 8, 02 2020.

[4] Izaak Neutelings. Cms coordinate system, Oct 2023.

[5] Matteo Cacciari, Gavin P Salam, and Soyez. et al. The anti-ktjet clustering algorithm. *Journal of High Energy Physics*, 2008(04):063–063, April 2008.

[6] Eric A. Moreno and et al. Cerri. Jedi-net: a jet identification algorithm based on interaction networks. *The European Physical Journal C*, 80(1), January 2020.

[7] Stefan Leijnen and Fjodor van Veen. The neural network zoo. In *IS4SI 2019 Summit*, IS4SI 2019. MDPI, May 2020.

[8] Wlademir Ribeiro Prates. Curva ROC e AUC em Machine Learning - Ciência e Negócios — cienciaenegocios.com. `https://cienciaenegocios.com/curva-roc-e-auc-em-machine-learning/`. [Accessed 10-09-2024].

[9] Armin Hermann, John Krige, Ulrike Mersits, and Dominique Pestre. *History of CERN*, volume 1. North-Holland Amsterdam, 1987.

[10] Fernando Evans. Brasil é 1o país das américas com status de "estado membro" do cern, que abriga maior colisor de partículas do mundo; entenda. *https://g1.globo.com*, 2024.

[11] L Arnaudon, M Magistris, M Paoluzzi, and et al Hori. Linac4 technical design report. Technical report, 2006.

[12] Kjell Johnsen. Cern intersecting storage rings (isr), 1973.

[13] Donald Cundy and Simone Gilardoni. Wsp: The proton synchrotron (ps): At the core of the cern accelerators. *Adv. Ser. Direct. High Energy Phys.*, 27:39–85, 2017.

[14] Niels Doble and et al. Gatignon. Wsp: The super proton synchrotron (sps): A tale of two lives. *Adv. Ser. Direct. High Energy Phys.*, 27:135–177, 2017.

[15] Maria Cristina Batoni Abdalla. *O discreto charme das partículas elementares.* Unesp, 2006.

[16] D Brandt, H Burkhardt, M Lamont, S Myers, and J Wenninger. Accelerator physics at lep. *Reports on Progress in Physics*, 63(6):939, 2000.

[17] Lyndon Evans and Philip Bryant. Lhc machine. *Journal of instrumentation*, 3(08):S08001, 2008.

[18] The ATLAS Collaboration, Aad, et al. The atlas experiment at the cern large hadron collider. *Journal of Instrumentation*, 3(08):S08003–S08003, August 2008.

[19] The ALICE Collaboration, Aamodt, et al. The alice experiment at the cern lhc. *Journal of Instrumentation*, 3(08):S08002–S08002, August 2008.

[20] The LHCb Collaboration, Alves, et al. The lhcb detector at the lhc. *Journal of Instrumentation*, 3(08):S08005–S08005, August 2008.

[21] Ian Bird, F Carminati, R Mount, B Panzer-Steindel, J Harvey, Ian Fisk, B Kersevan, P Clarke, M Girone, P Buncic, et al. Update of the computing models of the wlcg and the lhc experiments. Technical report, 2014.

[22] Joao Varela. Cms l1 trigger control system. Technical report, CERN-CMS-NOTE-2002-033, 2002.

[23] The CMS Trigger and Data Acquisition Group. The cms high level trigger. *arXiv preprint hep-ex/0512077*, 2005.

[24] Mark Thomson. *Modern Particle Physics*. Cambridge University Press, September 2013.

[25] Tom Lancaster and Stephen J Blundell. *Quantum field theory for the gifted amateur*. OUP Oxford, 2014.

[26] David Griffiths. *Introduction to elementary particles*. John Wiley & Sons, 2020.

[27] Leonardo Gariboldi et al. Lattes'contribution to the discovery of the π meson in bristol. In *Atti del 22. Congresso Nazionale di Storia della Fisica e dell'Astronomia*, pages 215–235. Citeseer, 2003.

[28] Francis Halzen and Alan D Martin. *Quark & Leptons: An introductory course in modern particle physics*. John Wiley & Sons, 2008.

[29] S. V. Chekanov. Jet algorithms: a minireview, 2002.

[30] Cedrick Miranda Mello. *Estudo da detecção de quarks top no LHC*. PhD thesis, Universidade de São Paulo, 2012.

[31] Andrinandrasana David Rasamoelina and et al. Adjailia. A review of activation function for artificial neural network. In *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, January 2020.

[32] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[33] Sebastian Ruder. An overview of gradient descent optimization algorithms, 2016.

[34] David E. Rumelhart and Hinton. et al. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.

[35] Pierini, Maurizio1, Duarte, et al. Hls4ml lhc jet dataset (30 particles). 2020. `https://zenodo.org/records/3601436`.

[36] E. Coleman, M. Freytsis, A. Hinzmann, M. Narain, J. Thaler, N. Tran, and C. Vernieri. The importance of calorimetry for highly-boosted jet substructure. *Journal of Instrumentation*, 13(01):T01003–T01003, January 2018.

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

[38] Andrew Collette. *Python and HDF5*. O'Reilly, 2013.

[39] Charles R. Harris and K. et al. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[40] Paszke, Adam, Gross, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[41] Gabriel Oliveira. gabriels3t/uso-de-redes-neurais-com-dataset-hls4ml-lhc-jet-dataset: Firt release, 2024. `https://zenodo.org/doi/10.5281/zenodo.13647045`.

[42] Gabriel Oliveira. Dados tratados do hls4ml 30 partículas, voltado para as 30 partículas (3 opção do dataset), 2024. `https://zenodo.org/doi/10.5281/zenodo.13684927`.

[43] F. Pedregosa and et al Varoquaux. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[44] Tianbao Yang and Yiming Ying. Auc maximization in the era of big data and ai: A survey. *ACM Computing Surveys*, 55(8):1–37, December 2022.

[45] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.