



Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas

Uso de redes neuronales para el estudio de rayos cósmicos

Tesis presentada al

Colegio de Física

como requisito parcial para la obtención del grado de

LICENCIADO EN FÍSICA APLICADA

por

Fernando Mendez Alonso

Asesorado por

Dr. Enrique Varela Carlos

Dr. Humberto Salazar Ibargüen

Puebla Pue.
28 de marzo de 2023



Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas

Uso de redes neuronales para el estudio de rayos cósmicos

Tesis presentada al

Colegio de Física

como requisito parcial para la obtención del grado de

LICENCIADO EN FÍSICA APLICADA

por

Fernando Mendez Alonso

Asesorado por

Dr. Enrique Varela Carlos

Dr. Humberto Salazar Ibargüen

Puebla Pue.
28 de marzo de 2023

Título: Uso de redes neuronales para el estudio de rayos cósmicos
Estudiante: FERNANDO MENDEZ ALONSO

COMITÉ

Dr. Oscar Martínez Bravo
Presidente

Dr. Marco Antonio Arroyo Ureña
Secretario

Dr. Lorenzo Díaz Cruz
Vocal

Dr. Jorge Cotzomi Paleta
Vocal

Dr. Enrique Varela Carlos
Asesor

Dr. Humberto Salazar Ibargüen
Coasesor

Agradecimientos

A mis padres Crisanto Cerón y Hortencia Alonso por el apoyo incondicional durante todos estos años, su confianza y comprensión cuando decidí estudiar física; gracias por tantos años de orientación y lecciones de vida, que no les quepa duda de que estaré siempre para ustedes, así como ustedes lo han estado para mí.

A mis amigos por ser una valiosa compañía durante todos estos años.

A mi jurado por su valiosa retroalimentación, Dr. Oscar Martínez Bravo, Dr. Marco Antonio Arroyo Ureña, Dr. Jorge Cotzomi Paleta y Dr. Lorenzo Díaz Cruz.

Finalmente le agradezco al Dr. Enrique Varela Carlos y al Dr. Humberto Salazar Ibargüen por su paciencia, apoyo y dirección en el desarrollo de este trabajo que hoy es una pieza importante en mi formación.

A todos ustedes, muchas gracias.

Índice general

Resumen	XI
Introducción	XIII
1. Rayos cósmicos	1
1.1. Espectro de energía de rayos cósmicos	1
1.1.1. Características en el espectro de energía de los rayos cósmicos	2
1.2. Origen de los rayos cósmicos	4
1.3. Procesos de interacción	6
1.3.1. Ionización y excitación	6
1.3.2. Radiación de Bremsstrahlung	7
1.3.3. Creación de pares	7
1.4. Cascadas atmosféricas extensas (CAE)	8
1.5. Modelización de la cascada	9
1.6. El eje y la dirección de la lluvia	10
2. Observatorio Pierre Auger	11
2.0.1. Consideraciones del Observatorio Pierre Auger	11
2.1. Detector de superficie	12
2.1.1. Detector Cherenkov	13
2.2. Detector de fluorescencia	13
2.2.1. Luz de fluorescencia	14
2.3. Sistema óptico	14
2.4. AERA	15
2.5. Extensiones al Observatorio Pierre Auger	15
3. Simulaciones de cascadas de partículas	17
3.1. Simulaciones de Monte-Carlo	17
3.1.1. Modelos de cascadas de partículas y CONEX	17
3.1.2. Aspectos de la simulación de una cascada de partículas	18
3.2. Producción de los eventos simulados	18
3.3. Offline	20
4. Aprendizaje automático y redes neuronales artificiales	21
4.1. Tipos de aprendizaje automático	22
4.2. Aprendizaje automático supervisado	22
4.3. Redes neuronales artificiales	23
4.3.1. El perceptrón	24
4.3.2. Redes neuronales multicapa	25
4.4. Algoritmo de propagación hacia atrás	26
4.5. Funciones de activación	30

4.6. Función de pérdida	31
5. Análisis y resultados	33
5.1. ROOT	33
5.1.1. Uproot	34
5.2. Estrategia de entrenamiento	35
5.2.1. Selección de los datos a utilizar	35
5.2.2. Conjunto de validación, entrenamiento y de prueba	37
5.2.3. Correlación con X_{max}	38
5.2.4. Preparación de los datos	39
5.3. Arquitectura de la red neuronal artificial desarrollada	40
5.4. Resultados	43
Conclusión	51
5.4.1. Trabajo a futuro	52
A. Instalación de ROOT	53
B. Instalación de CONEX	55
C. Instalación de APE	57
D. Instalación de Offline	59
Bibliografía	61

Índice de figuras

1.	Algoritmo DeepDream aplicado a una ilustración artística de una cascada de partículas medida en el Observatorio Pierre Auger. Imagen de [7].	xv
1.1.	Se muestra el espectro de energía de rayos cósmicos obtenido por varios experimentos. El flujo de rayos cósmicos, F , en el eje vertical se multiplica por $E^{2.6}$ con la intención de que sea más visible el espectro de la rodilla, la segunda rodilla y el tobillo. Imagen tomada de [1].	2
1.2.	Espectro de energía de partículas de los rayos cósmicos, medido por el Observatorio Pierre Auger a energías alrededor y por encima del tobillo. Los índices espectrales numerados γ_i corresponden a las regiones numeradas. Imagen tomada de [3]. . . .	3
1.3.	Diagrama de Hillas actualizado, donde se muestran los tamaños y magnitudes de campos magnéticos de posibles objetos astronómicos de aceleración de partículas. Imagen tomada de [3].	6
1.4.	Representación del proceso de ionización; una partícula cargada colisiona con un electrón y lo arranca del átomo, lo cual resulta en un electrón libre y un átomo ionizado [1].	7
1.5.	Representación del proceso de excitación; una partícula cargada colisiona con un electrón transfiriéndole energía que lo lleva a un estado de mayor energía. [1]. . . .	7
1.6.	Representación de la radiación de bremsstrahlung; una partícula cargada interactúa con el campo eléctrico de un núcleo atómico y pierde energía emitiendo un fotón.	8
1.7.	Representación del proceso de creación de pares.	8
1.8.	Desarrollo de una CAE inducida por un rayo cósmico. Se muestran las diferentes componentes de la cascada: electromagnética, hadrónica y muónica [5].	10
2.1.	Representación de los detectores del Observatorio Pierre Auger. Los detectores de superficie se representan como puntos negros, las líneas azules muestran el rango de visión de los seis detectores de fluorescencia en cada una de las cuatro estaciones: Los Morados, Los Leones, Coihueco y Loma Amarilla. Se muestran las extensiones HEAT, AERA y AMIGA.	12
2.2.	Fotografía de una estación de superficie del Observatorio Pierre Auger [17] y bosquejo esquemático del mismo, ambos mostrando sus principales componentes [7]. .	13
2.3.	Edificio FD en la estación Los Leones, detrás del edificio hay una torre de comunicación. La fotografía fue tomada cuando se abrieron las persianas por mantenimiento. Imagen tomada de [16].	14
2.4.	Fotografías de la cámara, el espejo y la apertura de un telescopio de fluorescencia. Imagen tomada de [7].	15
2.5.	Fotografía de una estación de radio que forma parte de AERA. Imagen tomada de [59].	16

2.6. Detector de superficie actualizado, posee un detector de centelleo y una antena de radio en la parte superior [57].	16
3.1. Fragmento del script.	19
4.1. Arquitectura de un perceptrón; los datos de entrada son multiplicados por los pesos w_i , se realiza una suma ponderada y se aplica una función de paso dando lugar al valor de salida.	25
4.2. Visualización de un perceptrón con entradas (1.0, 2.0, 3.0, 2.5) asociadas con sus respectivos pesos (0.2, 0.8, -0.5, 1.0) además de un sesgo con valor (2.0). El resultado de computar la suma ponderada y añadirle el valor de sesgo es 4.8.	25
4.3. Arquitectura básica de una red neuronal multicapa. La red tiene dos capas ocultas y una de salida. Se muestra la red con los perceptrones y la representación vectorial de la misma.	26
4.4. Ilustración gráfica de la notación usada para los pesos, sesgos y activaciones en un perceptrón en diferentes capas.	27
4.5. Imagen ilustrativa de cómo un cambio en un peso afecta al perceptrón al que está conectado ocasionando un cambio en su salida de activación, este a su vez afecta a todas las salidas de activación en la siguiente capa y este cambio se propaga hasta la salida ocasionando, finalmente, un cambio en C	28
4.6. (a) Función Sigmoide. (b) Función tangente hiperbólica. (c) Función Relu o Rectified Linear Unit.	30
5.1. Parámetros obtenidos en las simulaciones con CONEX.	36
5.2. Ejemplos de algunos histogramas realizados y una gráfica de X_{max} vs lgE	36
5.3. Ejemplo de la gráfica X_{max} contra lgE del modelo EPOS LHC utilizando como primario protón y la correspondiente distribución de X_{max} utilizando todos los datos de la simulación.	39
5.4. Ejemplo de la gráfica X_{max} contra lgE del modelo EPOS LHC utilizando como primario protón y la correspondiente distribución de X_{max} después de haber quitado algunos elementos.	39
5.5. Representación de nuestro modelo de red neuronal, incluyendo la capa de entrada, salida, capas ocultas y sus perceptrones correspondientes.	41
5.6. Representación del segundo modelo de red neuronal artificial, incluyendo la capa de entrada, salida, capas ocultas y sus perceptrones correspondientes.	42
5.7. X_{max} contra lgE obtenidas para protón y núcleos de hierro en cada uno de los modelos hadrónicos utilizados.	44
5.8. Comparación de la distribución de X_{max} del conjunto de prueba con los valores predichos por la red.	47
5.9. Se muestran las correlaciones halladas entre los valores de la simulación contra los predichos para los primarios y modelos hadrónicos empleados en este trabajo.	48
5.10. X_{max} contra logE para núcleos de hierro y protón en las simulaciones de Real Monte-Carlo, se puede apreciar que la red es capaz de reproducir los quiebres.	49
5.11. Comparación de la distribución de X_{max} del conjunto de prueba generado con Offline contra los valores predichos por la red para el modelo hadrónico QGSJETII-04 con núcleos de hierro y protón.	50
5.12. Se muestran las correlaciones halladas entre los valores de las simulaciones de Offline contra los predichos.	50

Resumen

En los últimos años hemos presenciado un avance significativo en el campo de la inteligencia artificial. Las técnicas de aprendizaje automático basadas en redes neuronales artificiales han mostrado resultados sobresalientes en la solución de problemas ya no solo del área de las ciencias computacionales y la ingeniería, sino también en otras áreas, en particular han demostrado ser herramientas útiles en la investigación científica en las áreas duras como la física y las matemáticas. En este trabajo se plantea explorar la capacidad de predicción de las redes neuronales artificiales para estudiar y de ser posible predecir características de las partículas más energéticas provenientes del espacio exterior conocidas como *Rayos Cósmicos*, cuyo estudio ha sido de gran relevancia en el descubrimiento de partículas elementales y el desarrollo de la física de altas energías, así mismo los rayos cósmicos son de gran interés porque pueden ayudarnos a revelar muchos misterios del universo que aún no tienen respuesta y comprender mejor qué objetos astronómicos se encuentran en el espacio exterior que tienen la capacidad de acelerar estas partículas a tan altas energías. Para este trabajo generamos un gran conjunto de datos de simulaciones de Monte-Carlo, posteriormente desarrollamos un modelo de una red neuronal artificial y utilizamos como datos de entrenamiento algunas características de los rayos cósmicos que se extraen de las simulaciones con el fin de predecir la profundidad atmosférica del máximo desarrollo de una cascada de partículas X_{max} que genera el o los rayos cósmicos simulados. Utilizamos como primario núcleos de hierro y protones y tres distintos modelos hadrónicos: EPOS LHC, QGSJETII-04 y SIBYLL.

Palabras clave: *Redes neuronales artificiales, Rayos cósmicos, Simulaciones de Monte-Carlo, Profundidad atmosférica, Máximo desarrollo de partículas X_{max} , Modelos hadrónicos.*

Introducción

Desde el descubrimiento de la existencia de los rayos cósmicos se han estudiado a lo largo de los años propiedades como:

1. Las energías que poseen
2. Sus direcciones de arribo
3. Su composición

Actualmente existen colaboraciones alrededor del mundo con cientos de científicos dedicados al estudio sobre estos temas, esto con el fin de obtener más información.

Los rayos cósmicos con energías iguales o mayores a 10^{18} eV (1 EeV) se denominan rayos cósmicos ultra energéticos (UHECR por sus siglas en inglés) y son las partículas más energéticas encontradas hasta ahora en la naturaleza, superando con algunos órdenes de magnitud las energías alcanzadas en colisionadores artificiales, su origen continúa siendo fuente de investigación, por lo que a estas energías, mediciones más precisas y estadísticamente significativas podrían brindar una nueva comprensión de las fuentes y la propagación de los rayos cósmicos, además también mediciones precisas de su masa (es decir de la composición) ayudaría a identificar las fuentes que aceleran a estos UHECR. En el ámbito de análisis de datos obtenidos de experimentos, debemos pensar que, una mejor reconstrucción de la energía, la dirección de arribo y la composición de los rayos podría ser clave para responder a las preguntas que permanecen abiertas. Por suerte para nosotros, estas propiedades pueden ser extraídas mediante mediciones de las cascadas de partículas ocasionadas por los rayos cósmicos de alta energía cuando entran en la atmósfera e interactúan con los átomos presentes en esta. Dichas mediciones a estas energías son realizadas por observatorios como el Observatorio Pierre Auger el cual utiliza un sistema de medición híbrido conformado por el muestreo de las partículas secundarias que llegan al suelo con su Detector de Superficie y evaluación calorimétrica del desarrollo longitudinal de la lluvia a lo largo de la atmósfera con su Detector de Fluorescencia.

El aprendizaje automático ha demostrado ser una herramienta poderosa en diversos campos de las ciencias, ingeniería y tecnologías como el reconocimiento de voz, procesamiento de lenguaje natural, visión por computadora, máquinas autónomas y particularmente se perfilan como una herramienta de gran valor en las ciencias tanto aplicadas como puras y muy particularmente en la física de astropartículas.

El desarrollo de nuevas tecnologías en los últimos años en el área de cómputo y supercómputo también han contribuido al avance del campo de la Inteligencia Artificial al cual pertenece el aprendizaje de máquina o aprendizaje automático, ahora tenemos frente a nosotros y a nuestro alcance nuevas tecnologías y técnicas que nos pueden ayudar, por ejemplo, para la reconstrucción de las características de las cascadas de partículas como la profundidad atmosférica del máximo desarrollo X_{max} , en la que nos enfocamos en este trabajo debido a su relación con la masa del primario para una determinada energía.

Desarrollamos una red neuronal artificial con el objetivo de predecir la profundidad atmosférica del máximo desarrollo de partículas X_{max} para dos primarios diferentes: núcleos de hierro y protones en un rango de energía de $10^{16.85}$ a $10^{18.625}$ eV y en tres modelos hadrónicos distintos: EPOS LHC, SIBYLL y QGSJETII-04. Para entrenar nuestro modelo generamos un conjunto de simulaciones de Monte-Carlo utilizando el software CONEX de las cuales extraemos las características de las lluvias que sirven como datos de entrenamiento para la red neuronal artificial. Posteriormente usamos los datos obtenidos usando el modelo QGSJETII-04 para su reconstrucción en el software Offline y entrenamos una nueva red neuronal artificial con estos datos, nuevamente tuvimos como objetivo predecir el valor de X_{max} .

En el capítulo 1 podemos encontrar una breve introducción a los rayos cósmicos, así como a una herramienta para su estudio conocida como espectro de energía de rayos cósmicos, algunas hipótesis sobre su origen y propagación y lo que es la profundidad atmosférica del máximo desarrollo conocida como x_{max} , también se habla sobre los principales procesos de interacción de las partículas que derivan en las cascadas atmosféricas extensas.

En el capítulo 2 hablamos sobre el Observatorio Pierre Auger, sus métodos de muestreo, su sistema híbrido, la capacidad de rangos de energía en la que opera y sus extensiones para poder estudiar un rango de energías más amplio.

En el capítulo 3 veremos el software y equipo que utilizamos para realizar las simulaciones, describimos brevemente estos programas y cómo lo utilizamos.

Proseguimos en el capítulo 4 a revisar lo que es el aprendizaje automático y los principales elementos de una red neuronal artificial.

En el capítulo 5 mostramos el trabajo realizado y los resultados obtenidos. Al final tendremos nuestras conclusiones y el trabajo a futuro.

Objetivos

Objetivo general

Crear modelos de redes neuronales artificiales, con el fin de que aprendan a predecir la profundidad atmosférica del máximo desarrollo de una lluvia de partículas X_{max} , como un primer paso para el uso de estas técnicas de aprendizaje automático, aplicadas en el estudio de los rayos cósmicos. Para esto, es necesario obtener primero un conjunto de datos de entrada a través del uso de software especializado en la generación de simulaciones de cascadas de partículas que se usan en el entrenamiento y prueba de desempeño de los modelos de redes neuronales artificiales desarrollados en este trabajo.

Objetivos específicos

1. Generar un conjunto de simulaciones de cascadas de partículas.
2. Extraer las características más relevantes de las simulaciones para predecir la variable de interés, X_{max} , las cuales sirven como datos de entrenamiento para la red neuronal.
3. Generar un nuevo conjunto de simulaciones denominadas Real Monte-Carlo utilizando el software Offline.
4. Desarrollar, entrenar y probar el rendimiento de una red neuronal artificial.

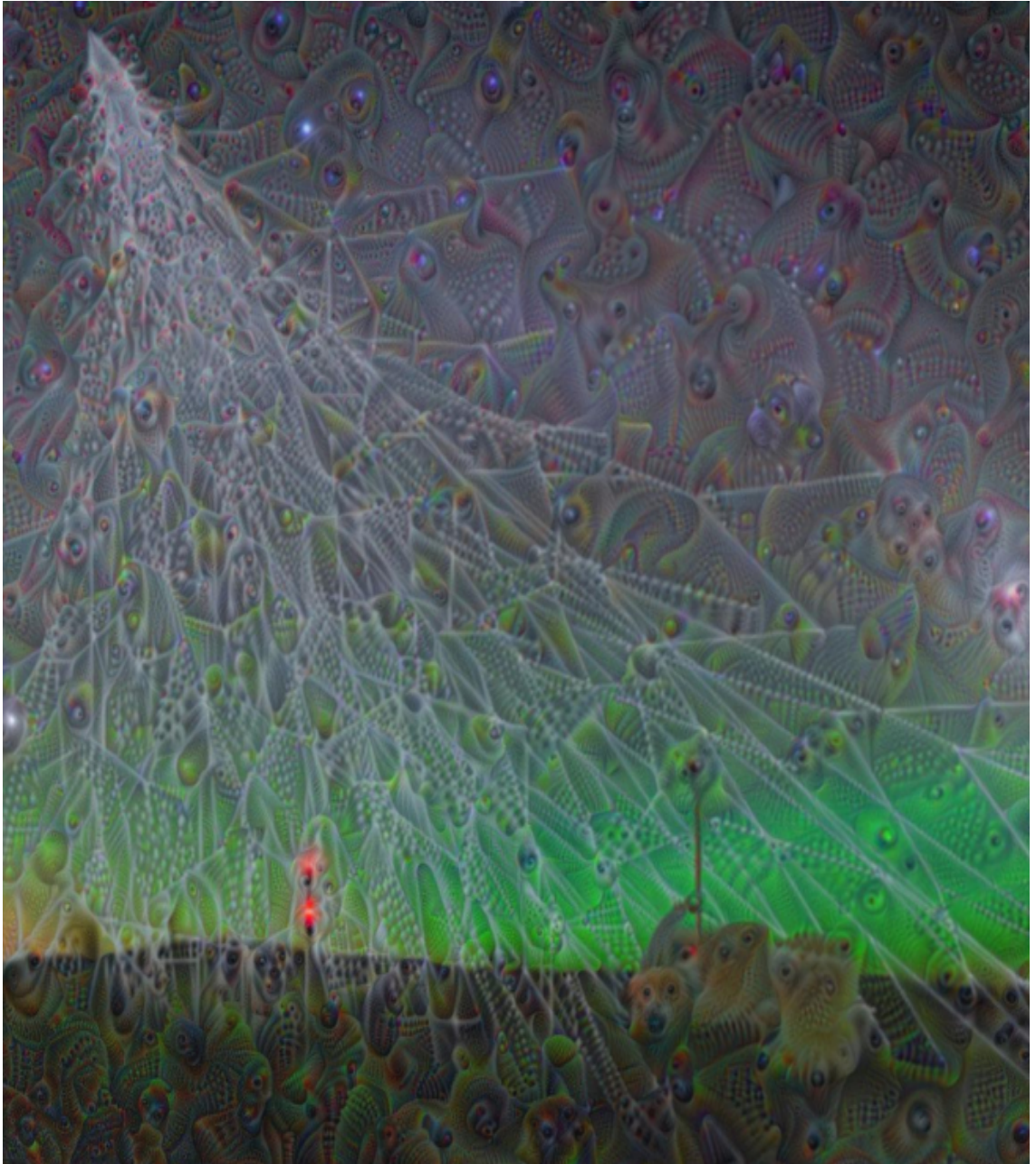


Figura 1: Algoritmo DeepDream aplicado a una ilustración artística de una cascada de partículas medida en el Observatorio Pierre Auger. Imagen de [7].

Capítulo 1

Rayos cósmicos

Desde que los rayos cósmicos fueron detectados comenzó un campo de investigación que ha revelado muchos secretos de la naturaleza. Incluso hoy en día nos proporcionan partículas que tienen mucha más energía de las que se han alcanzado en aceleradores de partículas artificiales. Si somos capaces de detectar con precisión a los rayos cósmicos, podemos poner a prueba nuestra comprensión de la naturaleza a las más altas energías observadas en el universo. Sin embargo, los rayos cósmicos de ultra alta energía solo se pueden observar indirectamente mediante las cascadas de partículas que provocan al interactuar con la atmósfera terrestre.

La mayoría de los rayos cósmicos son partículas cargadas de muy alta energía que están constantemente golpeando a la tierra desde todas las direcciones y cubren un rango de energías que va desde los 10^8 eV a 10^{20} eV. La mayoría de ellos son núcleos atómicos, principalmente protones, pero también se incluyen electrones, positrones, partículas con núcleo pesado y otras partículas subatómicas, mientras los fotones y neutrinos son considerados aparte. Las fuentes de los rayos cósmicos son desconocidas aún, pero se espera que sean de origen tanto galáctico como extragaláctico. Su estudio permite asociar las posibles fuentes celestes que son capaces de permitirles adquirir esas altas energías, identificar y profundizar sobre los modelos teóricos de los procesos ligados a los mecanismos de aceleración y dilucidar sobre su naturaleza.

Los rayos cósmicos ultra energéticos resultan ser misteriosos por las energías tan altas que alcanzan, pero también lo es el cómo se propagan que resulta ser difícil de explicar con los mecanismos tradicionales de aceleración de la astrofísica y con las leyes físicas conocidas que explican la propagación de partículas en el universo. Varios fenómenos violentos, como los núcleos galácticos activos que albergan agujeros negros masivos, los estallidos de rayos gamma o las estrellas de neutrones jóvenes que giran rápidamente, podrían ser candidatos para acelerar partículas hasta estas energías. Sin embargo, no existe evidencia por el momento de que estos objetos pueden acelerar partículas, protones u otros núcleos hasta las energías detectadas. Además, los datos actuales de los UHECR son demasiado escasos para correlacionarse con fuentes conocidas.

1.1. Espectro de energía de rayos cósmicos

El espectro de energía de los rayos cósmicos relaciona el flujo de los rayos cósmicos que llegan a la atmósfera terrestre con su energía. Es la forma en cómo cambia el flujo de partículas que llegan a la atmósfera terrestre con el aumento de energía. Se cree que las fuentes extragalácticas son las responsables de los rayos cósmicos de muy alta energía.

El flujo de rayos cósmicos depende fuertemente de la energía y se puede notar que disminuye drásticamente con el aumento de esta. El espectro tiene tres características principales denominadas rodilla, segunda rodilla y tobillo.

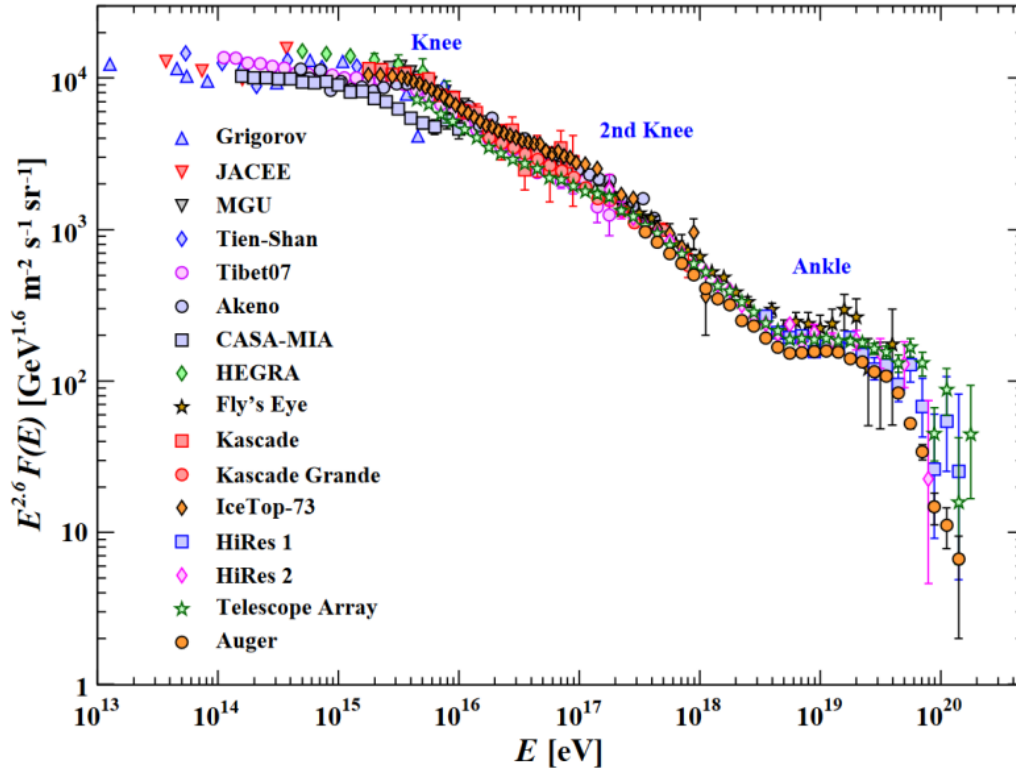


Figura 1.1: Se muestra el espectro de energía de rayos cósmicos obtenido por varios experimentos. El flujo de rayos cósmicos, F , en el eje vertical se multiplica por $E^{2.6}$ con la intención de que sea más visible el espectro de la rodilla, la segunda rodilla y el tobillo. Imagen tomada de [1].

Generalmente el flujo de los rayos cósmicos se puede describir mediante una ley de potencia, donde cada parte del espectro se comporta de la forma:

$$\frac{dN}{dE} \propto E^{-\gamma} \quad (1.1)$$

Donde γ es conocido como índice espectral, se piensa que los cambios en γ en las diferentes regiones energéticas del espectro se producen debido a cambios en la composición, las fuentes o la propagación.

1.1.1. Características en el espectro de energía de los rayos cósmicos

El índice espectral γ es diferente en las tres regiones previamente mencionadas (rodilla, segunda rodilla y tobillo) de esta forma se suele clasificar el espectro de rayos cósmicos en tres regiones principales:

A saber, la región anterior a la rodilla, cubre un rango de bajas energías ($E \approx 10^8 - 10^{15}$), en este rango de energías el flujo de rayos cósmicos es lo suficientemente alto como para ser observado directamente a grandes altitudes o en el espacio exterior, esto permite mediciones precisas de la composición hasta energías en el rango de GeV, en este rango la mayoría de los rayos cósmicos detectados se originan en el sol, ya que los rayos cósmicos galácticos de baja energía son bloqueados en el sistema solar debido, principalmente, al choque del viento solar cuando se encuentra con el medio interestelar. Más allá de unos pocos GeV, el sol alcanza su máxima potencia de aceleración

para los rayos cósmicos y su contribución al espectro disminuye. Con el aumento de energía las mediciones directas de la composición de rayos cósmicos han demostrado que las fuentes galácticas dominan el flujo, a energías aún más altas se cree que las fuentes son supernovas y remanentes de supernovas [5].

Alrededor de 3×10^{15} eV, se produce una disminución del flujo en el espectro, esta región se conoce como la rodilla. El comportamiento en esta región es menos estable, pero puede describirse bien mediante la ley de potencia con $\gamma \approx 3.1$ [5].

Una segunda disminución en el flujo se produce alrededor de los 10^{17} eV, a esta región se le otorga el nombre de segunda rodilla, el índice espectral es $\gamma = 3.3$. A estas energías los posibles candidatos a fuentes galácticas se vuelven pocos y los rayos cósmicos de fuentes extragalácticas probablemente comienzan a contribuir con una fracción significativa del flujo observado [5].

En la región de energía de alrededor de 5×10^{18} eV el índice espectral toma un valor de $\gamma \approx 2.5$, se presenta una drástica disminución en el flujo, esta región se conoce como tobillo, se cree que las fuentes extragalácticas dominan el flujo en esta zona. En la Figura 1.2 se muestra claramente este comportamiento del flujo en el tobillo. Son visibles características adicionales, con un nuevo cambio del espectro de $\gamma \approx 3.05$ a alrededor de 13 EeV, seguido de otro cambio aún más fuerte a 46 EeV a un índice espectral de $\gamma \approx 5.1$, que se conoce comúnmente como el corte [5].

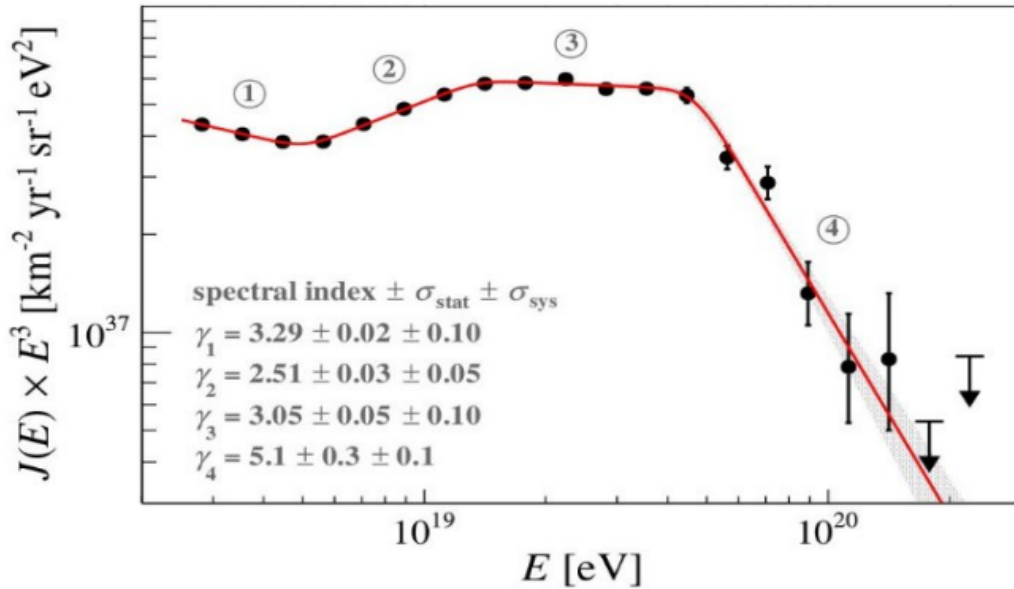


Figura 1.2: Espectro de energía de partículas de los rayos cósmicos, medido por el Observatorio Pierre Auger a energías alrededor y por encima del tobillo. Los índices espectrales numerados γ_i corresponden a las regiones numeradas. Imagen tomada de [3].

Es muy probable que ambas rodillas se deban a las escasas fuentes de rayos cósmicos dentro de nuestra galaxia que puedan producir partículas con tales energías, para energías mayores al tobillo los rayos cósmicos ya no pueden ser confinados por el campo magnético de la Vía Láctea y provienen de todas las direcciones por igual. Esto indica que los rayos cósmicos proceden de fuera de nuestra galaxia, de esta manera la rodilla podría interpretarse como la máxima energía a la que se pueden acelerar los protones en nuestra galaxia, la segunda rodilla podría considerarse como la máxima energía de aceleración para los núcleos de hierro. El tobillo podría entenderse entonces como un componente restante de rayos cósmicos extragalácticos. Otra teoría sospecha que la causa de la reducción del flujo es el llamado efecto GZK, el cual nos dice que la repentina caída en las energías más altas se debe a que los rayos cósmicos por encima de 5×10^{19} eV interactúan con

la radiación de fondo cósmico de microondas (CMB), perdiendo energía y produciendo fotones y neutrinos en el proceso, por lo tanto, los protones restantes se desplazarán hacia energías más bajas acercándose al tobillo [5].

A bajas energías el flujo es abundante, en promedio se pueden medir aproximadamente 1,000 rayos cósmicos por m^2 lo que facilita su estudio mediante experimentos con globos o satélites a gran altitud. Por encima de la rodilla, a energías superiores a 10^{15} eV, la detección directa se vuelve impráctica o incluso imposible. A energías de 10^{18} eV solo es posible detectar una partícula por km^2 por año, para energías de 10^{20} eV solo se puede detectar una partícula por km^2 por siglo. De modo que su estudio es de manera indirecta a través de la observación de las partículas secundarias producidas por interacciones entre los rayos cósmicos primarios y las partículas presentes en la atmósfera.

1.2. Origen de los rayos cósmicos

Hasta principios del siglo XX, la única forma de la que disponíamos para conocer el cosmos era la luz proveniente de los objetos celestes, el resto del espectro electromagnético era inaccesible, debido principalmente a la absorción atmosférica. Con el descubrimiento de los rayos cósmicos obtuvimos una nueva fuente de información extraterrestre de cuyo estudio derivaron algunos aspectos de la física nuclear, la física de altas energías y la física de partículas elementales. La investigación de los rayos cósmicos estimuló la creación de modelos estelares u otros objetos celestes; fue en la radiación cósmica donde se descubrieron muchas de las llamadas partículas elementales, de la misma manera, en las colisiones de los rayos cósmicos con los núcleos atmosféricos fue en donde se estudiaron por primera vez las interacciones nucleares. Por lo tanto, los rayos cósmicos son una ventana al Universo, ¡la única que trae hasta la tierra materia proveniente de objetos celestes lejanos! [8]

Desde el descubrimiento de los rayos cósmicos hasta hoy en día han habido múltiples intentos por explicar cómo es posible que las partículas cargadas con tan altas energías son producidas en el espacio exterior y el cómo llegan hasta nosotros, Millikan, les dio el nombre de el "grito primigenio" de elementos pesados que están siendo creados continuamente en el universo y creía que eran rayos γ que se producían cuando los protones y electrones formaban núcleos de hielo en el espacio interestelar; muchos otros intentos de explicar su procedencia involucran diversas hipótesis que hacen alusión a los más diversos mecanismos de aceleración que van desde tormentas estelares hasta el mismo origen del universo [8].

Es muy probable que la generación de rayos cósmicos se dé en los objetos astrofísicos de donde adquieren parte de la energía liberada; así pues, la dificultad para proponer modelos sobre su origen reside no tanto en inventar hipótesis, sino en verificar si ellas se ajustan a las observaciones. Las principales preguntas para responder sobre los rayos cósmicos residen en comprender el origen, propagación y naturaleza de estas partículas con tan alta energía. Si pensamos que las partículas con mayor energía detectadas poseen una energía en varios órdenes de magnitud mayor que la obtenida a través del más poderoso acelerador de partículas construido nos vienen a la mente una serie de preguntas: ¿En dónde se producen los rayos cósmicos?, ¿Cómo son acelerados?, ¿Cuándo fueron producidos?, ¿Cómo es su propagación en el espacio exterior?, ¿De qué tipo de partículas primarias se trata?, ¿Se podrán detectar partículas con energías superiores?

Las respuestas a estas preguntas aún son temas de investigación y están en busca de ser respondidas por la actual comunidad de científicos que estudia a los rayos cósmicos, para responderlas es necesario medir la energía y dirección de arribo, así como identificar a la partícula primaria, es decir la composición.

Hoy en día sabemos que los rayos cósmicos con energías menores a GeV provienen del sol, mientras que los rayos cósmicos con energías correspondientes a la zona de la primera y segunda rodilla deberían ser de origen galáctico. Por otro lado, se piensa que los rayos cósmicos con energías dentro y más altas que las que corresponden a la región del tobillo son de origen extragaláctico esto

debido a que los objetos astronómicos conocidos hasta el momento de nuestra galaxia generalmente no poseen el tamaño y magnitud de campo magnético que pueda contener partículas con estas energías. Sin embargo, la mayoría de las fuentes que aceleran estas partículas de muy altas energías siguen siendo desconocidas y existen diferentes mecanismos que tratan de explicar el proceso de aceleración de rayos cósmicos.

Los modelos de mecanismos de aceleración propuestos se pueden clasificar en dos: los modelos *Bottom-Up* y los modelos *Top-Down*.

Los modelos *Bottom-Up* nos dicen que las partículas cambian de un estado de menor energía a otro de mayor energía a través de un proceso de aceleración en objetos astrofísicos. Entre los mecanismos de aceleración *Bottom-Up*, el más aceptado es el mecanismo de aceleración en shocks difusos, basado en el modelo de Fermi, donde una partícula cargada que se encuentra confinada en un objeto astrofísico es acelerada lentamente debido a su movimiento aleatorio. La energía máxima que puede alcanzar la partícula depende tanto de la magnitud del campo magnético (B) como de la extensión (R) del objeto astrofísico, limitando así las posibles fuentes.

Existen dos formas que tratan de explicar estos procesos de aceleración:

1. Las partículas van ganando energía conforme se encuentran en su camino regiones de campos magnéticos que las van acelerando.
2. Las partículas son aceleradas directamente por un campo eléctrico (por ejemplo fem que surgen en conductores magnetizados que giran rápidamente, como estrellas de neutrones u objetos supermasivos).

En la figura 1.3 se muestra una actualización del diagrama propuesto por Hillas en 1984 donde se muestran diferentes candidatos de fuentes que podrían tener la capacidad de acelerar UHECR, tales como: estrella de neutrones, núcleos activos de galaxias (AGN), brotes de rayos gamma (GRBs) y choques de acreción en el medio intergaláctico (IGM shocks), que según su extensión y magnitud de campo magnético pueden llegar a acelerar partículas a 10^{20} eV [3].

Los modelos *Top-Down*, por su parte, proponen la existencia de partículas supermasivas que decaen en rayos cósmicos ultra energéticos. Una de las fuentes de estas partículas exóticas podría ser los “Defectos Topológicos” que serían vestigios de transiciones de fase en el universo temprano con rupturas espontáneas de simetría, los cuales podrían existir en el presente. Los modelos *Top-Down* predicen rayos cósmicos dominados fundamentalmente por neutrinos, nucleones y rayos γ . La teoría dice que estos mecanismos no aportan a la creación de núcleos pesados con alta energía sino solo protones y partículas sin masa, además se prevé que contribuyan significativamente con partículas del orden de 10^{20} eV y no aporten de manera importante en el rango de energía de 10^{14} a 10^{19} eV. Estos modelos muestran algunas incompatibilidades con los resultados obtenidos experimentalmente haciendo que los modelos *Bottom-Up* sean más aceptados actualmente, sin embargo los modelos *Top-Down* no se han descartado totalmente [3].

Desde que los rayos cósmicos escapan de la fuente en donde fueron acelerados hasta que llegan a la Tierra se ven influenciados tanto por la radiación de fondo cósmica con la que interaccionan produciendo partículas secundarias de menor energía, como por los campos magnéticos galácticos y extragalácticos, que los desvían de su trayectoria de propagación. En la radiación de fondo cósmica con la que interactúan los UHECR en el medio intergaláctico, se encuentra la radiación de fondo de microondas (CMB) y la luz de fondo extragaláctica (EBL) produciendo fotodesintegración, creación de pares $e^{\pm}$ y producción de piones. Greisen y Zatsepin y Kuzmin predijeron la existencia de un corte en el espectro de protones, debido a la radiación de fondo de microondas, alrededor de 5×10^{19} eV principalmente debido a la producción de fotopiones en el fondo de microondas, plantean que los protones con energías mayores a 6×10^{19} eV tienen una alta probabilidad de interactuar con los fotones del fondo de radiación cósmico. Este límite de espectro recibió el nombre de corte GZK en honor a sus descubridores. Este último proceso mencionado es importante debido a que explica la supresión GZK del flujo de los rayos cósmicos, además de predecir la existencia de fotones y neutrinos ultra energéticos provenientes del decaimiento de los piones producidos [10].

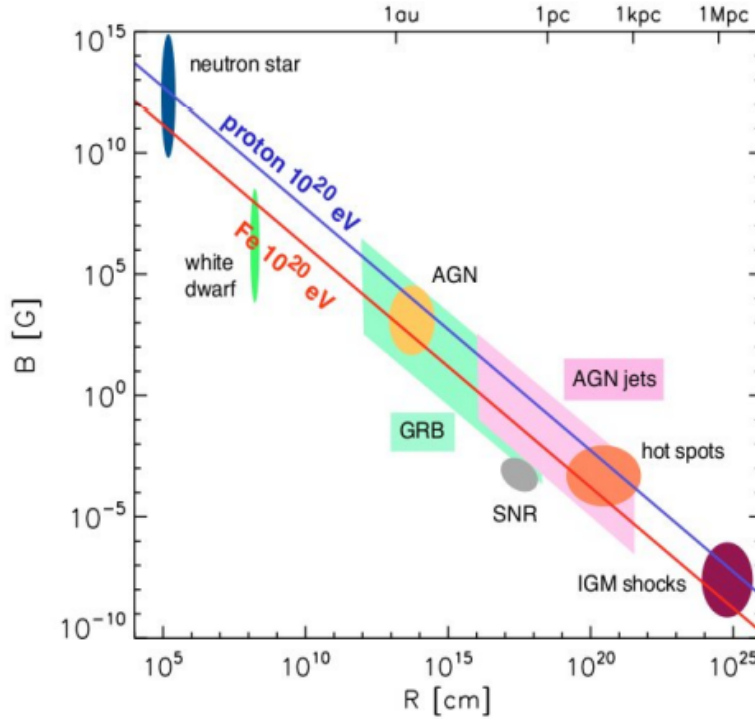


Figura 1.3: Diagrama de Hillas actualizado, donde se muestran los tamaños y magnitudes de campos magnéticos de posibles objetos astronómicos de aceleración de partículas. Imagen tomada de [3].

1.3. Procesos de interacción

Cuando un rayo cósmico alcanza la atmósfera terrestre se producen una serie de interacciones entre este y la materia presente (moléculas de nitrógeno, oxígeno, argón etc.) lo que resulta en que se crean nuevas partículas conocidas como partículas secundarias. Al interaccionar con la materia de esta manera se presentan colisiones que derivan en la pérdida de energía de la partícula primaria y son estos procesos de los cuales se producen las partículas secundarias. Describiremos brevemente las interacciones que conducen a lo anterior descrito.

1.3.1. Ionización y excitación

Una partícula cargada pierde energía debido a que colisiona con átomos y dependiendo de la energía cedida por la partícula incidente el átomo puede ionizarse o excitarse.

Si una partícula colisiona contra los electrones en un átomo y cede la suficiente energía para expulsar a un electrón del estado en el que se encuentra, es decir, la energía perdida excede la energía de enlace del electrón colisionado este es arrancado del átomo por lo que se tiene un electrón libre y en consecuencia se crea un ion con carga positiva, tenemos así una ionización ocasionada por la transferencia de energía de una partícula cargada a los electrones de un átomo, esto es conocido como ionización primaria. Existe otro tipo de ionización, conocido como ionización secundaria, la cual ocurre cuando los electrones libres producidos en la ionización primaria tienen

energía suficiente para producir nuevas ionizaciones.

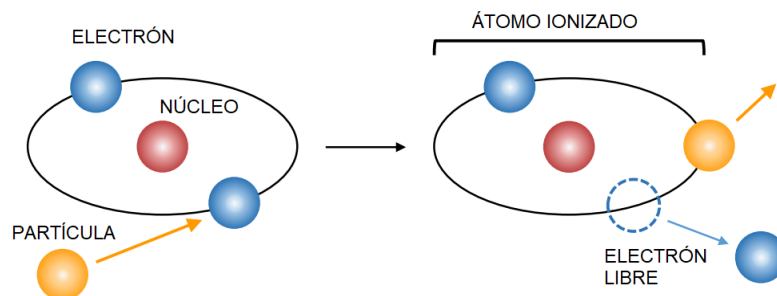


Figura 1.4: Representación del proceso de ionización; una partícula cargada colisiona con un electrón y lo arranca del átomo, lo cual resulta en un electrón libre y un átomo ionizado [1].

Por otro lado, si la energía que cede la partícula cargada que colisiona con el electrón no es suficiente para ionizar el átomo, aun transfiere energía a este y es promovido a un estado de mayor energía y como resultado tenemos un átomo excitado, eventualmente el electrón volverá a su estado anterior y la energía sobrante será emitida en forma de radiación electromagnética.

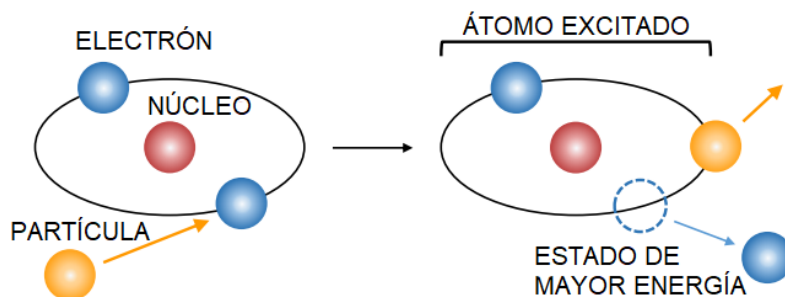


Figura 1.5: Representación del proceso de excitación; una partícula cargada colisiona con un electrón transfiriéndole energía que lo lleva a un estado de mayor energía. [1].

1.3.2. Radiación de Bremsstrahlung

La radiación de Bremsstrahlung o radiación de frenado es un tipo de radiación electromagnética ocasionada cuando una partícula incidente con masa y carga eléctrica interacciona con el campo eléctrico de un núcleo atómico por lo que la partícula es desacelerada o desviada, proceso por el cual pierde energía, esta energía perdida es emitida como radiación electromagnética en forma de fotones de bremsstrahlung.

1.3.3. Creación de pares

Cuando un fotón de alta energía interacciona con un tercer cuerpo, tal como un átomo (es necesario este tercer cuerpo para que se conserve la energía), puede transferir suficiente energía para la creación de un par positrón-electrón. Este nuevo par de partículas se crean debido a un proceso conocido como creación de pares donde un fotón de alta energía interactúa con un campo electromagnético y se transforma en un nuevo par de partículas de igual masa, pero con cargas opuestas. Como la energía se conserva tenemos que la energía del fotón que desaparece debe ser

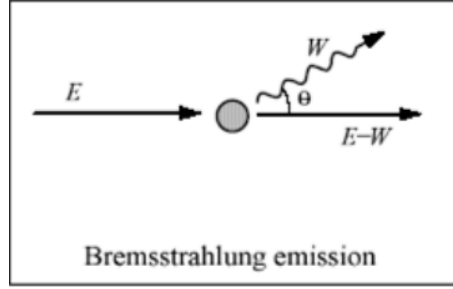


Figura 1.6: Representación de la radiación de bremsstrahlung; una partícula cargada interacciona con el campo eléctrico de un núcleo atómico y pierde energía emitiendo un fotón.

$E = h\nu \geq 2m_e c^2 = 1.022 \text{ MeV}$, la cual es la energía mínima requerida para que pueda ocurrir este proceso.

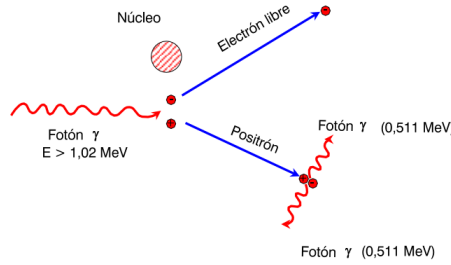


Figura 1.7: Representación del proceso de creación de pares [58].

1.4. Cascadas atmosféricas extensas (CAE)

Para estudiar a los rayos cósmicos a las más altas energías se aprovechan los efectos ocasionados por la interacción de estos con la atmósfera terrestre, como ya mencionamos se producen partículas secundarias, lo que ocasiona un fenómeno que se denomina cascadas atmosféricas extensas (CAE) o cascada de partículas o lluvia de partículas, el cual consiste en que cuando la partícula interacciona con otras partículas se crean nuevas partículas y estas a su vez avanzan y crean nuevas partículas, proceso que se repite durante todo el avance de las partículas hasta que la energía disponible no permite la creación de más partículas secundarias y se producen fenómenos de absorción y pérdida de energía. Las partículas secundarias producidas en estas cascadas contienen información acerca de la partícula primaria. Aunque la reconstrucción de las propiedades de la partícula primaria es “posible”, existen dificultades en este proceso. Las simulaciones de Monte-Carlo, son una herramienta computacional que nos ayuda a conectar las observables de la lluvia con las características del primario.

Para darnos una idea, la energía de las partículas primarias es tal que permiten la creación del orden de 10^{11} partículas secundarias y un gran porcentaje de ellas logra alcanzar el nivel de la superficie terrestre.

La atmósfera terrestre presenta, entre otras, las siguientes características [10]:

- Una baja densidad atómica, lo que permite una dispersión que cubre un área extensa de las partículas secundarias generadas por la partícula primaria.

- El espesor vertical, que es adecuado para que las cascadas alcancen su desarrollo máximo de partículas, esto permite relacionar proporcionalmente el número total de partículas con la energía total.
- La inhomogeneidad del perfil de densidad, gracias a esto es posible establecer diferentes zonas de desarrollo de las cascadas donde se producen los procesos de multiplicación y absorción de partículas.

Las mediciones se realizan al nivel del suelo lo cual tiene la desventaja de que no son mediciones directas de la partícula primaria ni de sus primeras interacciones. Sin embargo, a partir del estudio de las partículas secundarias se puede inferir la naturaleza de la partícula primaria y su comportamiento en las primeras interacciones.

La cascada generada tiene una forma aproximada a un disco cuyo centro es el eje de la lluvia que aumenta su diámetro a medida que avanza. Este disco se mueve casi a la velocidad de la luz y adquiere una máxima extensión cuando coexisten el mayor número de partículas secundarias, esto se conoce como el máximo de la lluvia. La profundidad atmosférica correspondiente a este máximo se denomina X_{max} . Tras alcanzar este máximo desarrollo de partículas se reduce el diámetro del disco debido a que la energía ya no es suficiente para producir nuevas partículas. Las unidades utilizadas son g/cm^2 , donde $0 g/cm^2$ corresponde a la parte superior de la atmósfera y $1000 g/cm^2$ al nivel del mar.

La primera interacción suele ocurrir aproximadamente entre los 15 y 35 km de altitud. Las partículas secundarias se clasifican en tres componentes principales: la componente hadrónica, la componente muónica y la componente electromagnética.

La componente electromagnética está dominada por la creación de pares de Bremsstrahlung. Se producen partículas secundarias hasta que la energía cae por debajo de un valor de energía crítico ϵ_0 . De esta manera se observan tres fases en el desarrollo de la lluvia: la fase de crecimiento en la cual todas las partículas tienen una energía mayor que ϵ_0 , la fase del máximo desarrollo X_{max} y la fase donde las partículas son absorbidas o se desintegran sin la creación de nuevas partículas secundarias.

El número de partículas secundarias en la componente electromagnética de la cascada es de varios órdenes de magnitud, por ejemplo, para una lluvia originada por un protón de 10^{11} GeV se producen alrededor de 10^{11} partículas secundarias de las cuales el 99 % son fotones, electrones y positrones y transportan alrededor del 85 % de la energía total que logra alcanzar el nivel del suelo [10].

1.5. Modelización de la cascada

Cuando ingresa a la atmósfera la partícula primaria atraviesa una cantidad de materia que está dada por la longitud de radiación X_0 cuyo valor aproximado es [10]:

$$X_0 = \frac{716.4 A}{Z(Z+1) \ln \frac{287}{Z}} \quad (1.2)$$

En donde Z es el número atómico y A es el número másico.

Cuando el primario interactúa con los núcleos presentes en la atmósfera se genera un nuevo par de partículas, estas nuevas partículas atraviesan una nueva X_0 , generando un nuevo par de manera que se va duplicando el número de partículas por cada secundario producido, así, transcurridas n_{max} generaciones de partículas la lluvia alcanza su máximo desarrollo, produciéndose $N_{max} = 2^{n_{max}}$ secundarios a una profundidad atmosférica que está dada por $X_{max} = X_0 \cdot n_{max}$, a su vez cada partícula posee una energía E_0/E_c , donde E_0 es la energía de la partícula primaria y E_c es la energía mínima necesaria para producir un nuevo par de partículas [10].

Se puede emplear el número de partículas en cada generación para determinar el máximo desarrollo de la lluvia

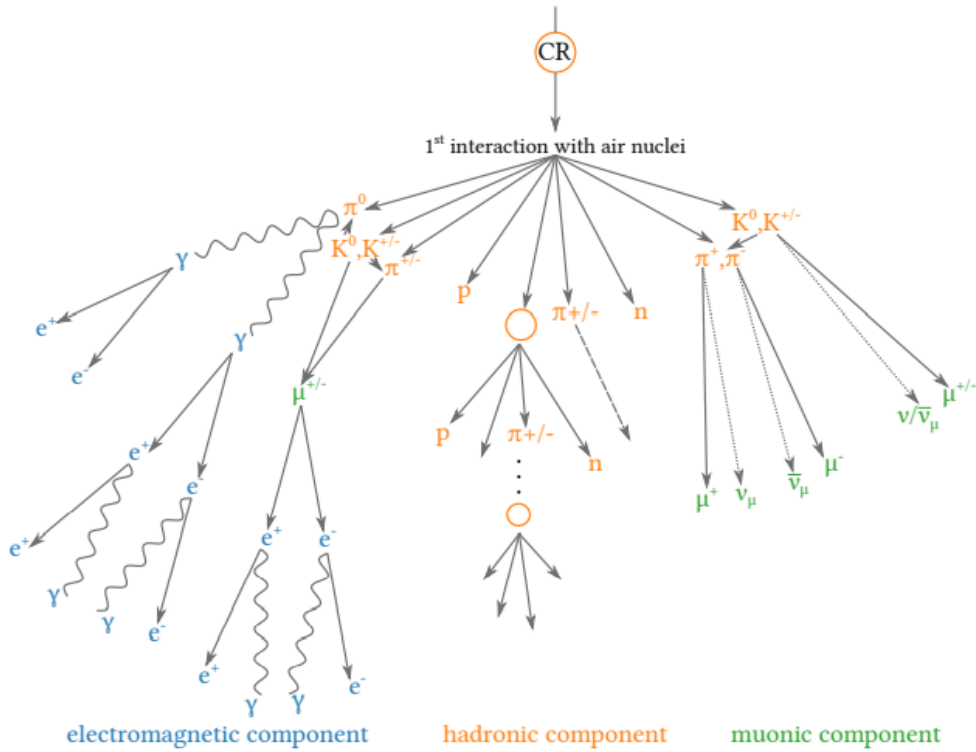


Figura 1.8: Desarrollo de una CAE inducida por un rayo cósmico. Se muestran las diferentes componentes de la cascada: electromagnética, hadrónica y muónica [5].

$$X_{max} = X_0 \frac{\ln(E_0/E_c)}{\ln 2} \quad (1.3)$$

donde X_0 es la longitud de radiación, aproximadamente 37 g/cm^2 en el aire.

Notemos que el máximo de la lluvia depende del logaritmo de la energía del primario.

Esto nos brinda una descripción precisa de la componente electromagnética, sin embargo, no se puede utilizar regularmente para modelar cascadas de partículas, en general debido a que la componente hadrónica evoluciona de forma diferente.

1.6. El eje y la dirección de la lluvia

La dirección de la cascada es la misma que la que lleva el rayo cósmico que la produce. Se puede deducir esta dirección haciendo uso de los datos obtenidos por el detector de superficie gracias a la señal captada por al menos tres detectores no alineados.

Como se mencionó anteriormente, las partículas se dispersan en un disco que aumenta su radio hasta alcanzar el máximo desarrollo de partículas. Si el arreglo de detectores se encuentra a una profundidad adecuada podrá detectar distintas señales en una amplia zona. Por esta razón los arreglos de superficie diseñados para detectar rayos cósmicos de energía mayor a 10^{19} eV deben situarse en regiones de altura comprendida en el equivalente a 750 g/cm^2 y 120 g/cm^2 ya que el promedio de la profundidad máxima de lluvias provocado por primarios con dichas energías ocurre alrededor de 750 gr/cm^2 y para obtener valores precisos en la determinación de los parámetros conviene estudiar la lluvias después del valor de X_{max} .

Capítulo 2

Observatorio Pierre Auger

El observatorio Pierre Auger, nombrado así en honor a Pierre Víctor Auger (1899 - 1993), a quien se puede considerar el descubridor de las cascadas atmosféricas extensas generadas por la interacción de rayos cósmicos de muy alta energía con la atmósfera terrestre, es el observatorio de rayos cósmicos más grande del mundo que cubre un área de $3,000 \text{ km}^2$, está ubicado en el oeste de Argentina en la Pampa Amarilla, cerca de la ciudad de Malargüe, construido desde el año 2002 al año 2008 con el propósito de operar en el rango de energías mayores a 10^{18} eV con el fin de obtener grandes cantidades de datos de las cascadas. Está situado en una región con un cielo claro y poca nubosidad la mayor parte del tiempo, a una altitud de aproximadamente 1400 m sobre el nivel del mar que corresponde a 875 g/cm^2 , lo cual hace posible observar la lluvia después de su máximo desarrollo. Para obtener mediciones precisas de las lluvias se emplea una técnica de **detección híbrida** que consiste en 1600 detectores de superficie (SD por sus siglas en inglés) y 4 detectores de fluorescencia (FD por sus siglas en inglés). El diseño del observatorio se asemeja a un polígono en el que se encuentran ubicados los detectores de superficie con una separación 1500 m entre cada detector hasta cubrir el área total, sobre pequeñas elevaciones alrededor del perímetro del polígono, y aproximadamente espaciados a la misma distancia entre sí, se encuentran cuatro detectores de fluorescencia que contienen seis cámaras cada uno, esta distribución permite que la mayor parte de las cascadas que detectan también sean detectadas por los SD de esta manera se pueden combinar los datos recopilados por cada método para obtener mediciones más precisas. Adicional a esto se han instalado componentes adicionales que permiten el estudio de rayos cósmicos con mayor detalle, uno de ellos es AMIGA (Auger Muon and Infill Ground Array) el cual combina las mediciones realizadas por los detectores de superficie con centelladores subterráneos para medir los muones provocados por una CAE, otra importante actualización que se ha implementado es conocido como AERA (Auger Engineering Radio Array), consiste en 150 estaciones de antenas de radio, cubriendo un área de 17 km^2 , estas antenas de radio permiten medir la emisión de radio producida por una cascada atmosférica.

Ventajas de la Detección Híbrida

El FD solo puede operar en noches claras con menos del 60 % de la luz reflejada en noches de luna llena haciendo que su ciclo útil sea del 15 % pero sus datos son usados para **calibrar las mediciones del SD**, esta es una de las principales características y ventajas de combinar dos técnicas de detección.

2.0.1. Consideraciones del Observatorio Pierre Auger

El Observatorio está diseñado para soportar las condiciones ambientales de la Pampa Amarilla, en esta zona la temperatura oscila entre los 15°C a los 50°C con una gran variación durante el

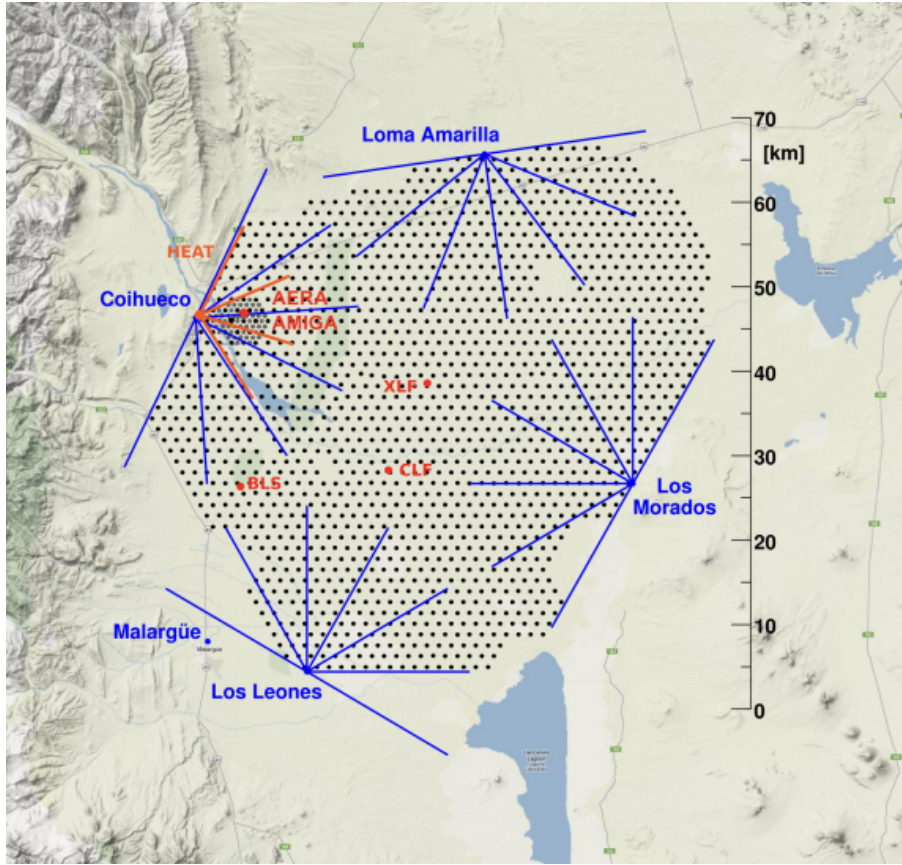


Figura 2.1: Representación de los detectores del Observatorio Pierre Auger. Los detectores de superficie se representan como puntos negros, las líneas azules muestran el rango de visión de los seis detectores de fluorescencia en cada una de las cuatro estaciones: Los Morados, Los Leones, Coihueco y Loma Amarilla. Se muestran las extensiones HEAT, AERA y AMIGA.

día. Debido a que los detectores se encuentran al aire libre están expuestos a una intensa radiación ultravioleta y al viento que puede alcanzar velocidades de hasta 160 km/h. Los detectores son resistentes a inundaciones, lluvia, nieve, polvo, arena y granizo. Los tanques de los detectores son capaces de soportar a una persona pesada encima y resisten daños que pueden ser ocasionados por insectos, roedores y animales que pastan. Así mismo los detectores están libres de vegetación densa a su alrededor para evitar daños que pueden ser ocasionados por incendios forestales.

2.1. Detector de superficie

El detector de superficie del Observatorio Pierre Auger mide la densidad de las partículas secundarias generadas por la interacción de la partícula primaria con los átomos presentes en la atmósfera. Abarcan un área de $3,000 \text{ km}^2$ y poseen un ciclo de trabajo de casi el 100 % gracias a lo cual tiene la capacidad para medir rayos cósmicos ultra energéticos de hasta 10^{20} eV con estadísticas adecuadas.

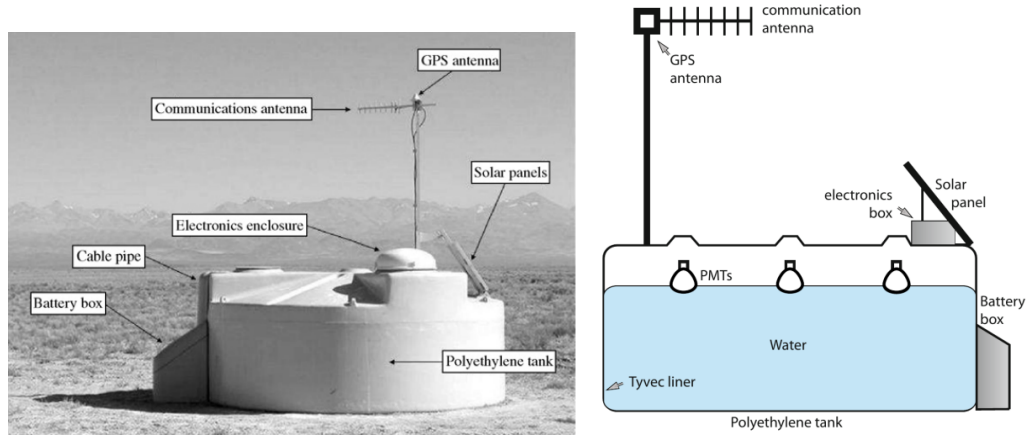


Figura 2.2: Fotografía de una estación de superficie del Observatorio Pierre Auger [17] y bosquejo esquemático del mismo, ambos mostrando sus principales componentes [7].

2.1.1. Detector Cherenkov

Un detector de agua Cherenkov es aquel que usa agua como medio de interacción para producir lo que se conoce como radiación Cherenkov la cual se produce cuando una partícula cargada se mueve en un medio transparente con velocidad mayor al que la luz tiene en dicho medio produciendo una perturbación electromagnética que origina una emisión de luz. Un tanque de agua hermético y oscuro resulta ser un buen detector del rastro de la partícula si este posee fotomultiplicadores, esto es justo lo que son los detectores de superficie del Observatorio Pierre Auger. Una estación detectora de superficie o detector de agua Cherenkov (WCD: Water Cherenkov detector) está fabricado en polietileno, tiene un diámetro de 3.6 m y una altura de 1.2 m, contiene un revestimiento sellado con una superficie interior reflectante que se llena con 12,000 litros de agua ultrapura. Tres tubos fotomultiplicadores (PMT: Photomultiplier tubes) de 9 pulgadas de diámetro están distribuidos simétricamente en la superficie del revestimiento a una distancia de 1.20 m del eje central del tanque los cuales captan la luz de Cherenkov producida por el paso de partículas cargadas relativistas a través del agua [16].

Los detectores de agua Cherenkov están colocados en forma hexagonal, cada detector está separado de sus seis vecinos más cercanos por una distancia de 1500 metros, el Observatorio Pierre Auger cuenta con 1600 detectores de este tipo y cubren un área de 3000 km^2 . Cabe destacar que dentro de este arreglo existen dos regiones más densas, una de las cuales cubre un área de 28 km^2 , contiene 71 SD separados por 750 metros entre sí y la segunda región cubre un área de 2 km^2 con un espaciamiento de 433 metros entre cada detector, estas partes más densas permiten obtener datos de las cascadas con energías de alrededor de 10^{17} eV .

Cada estación SD cuenta con un panel solar que le otorga la energía necesaria para el funcionamiento de los PTM y su paquete electrónico que consta de un procesador, un receptor GPS, un transceptor de radio y un controlador de radio. Para su funcionamiento durante la noche cada estación cuenta con baterías de plomo-ácido de 12 V [7].

2.2. Detector de fluorescencia

El detector de fluorescencia del Observatorio Pierre Auger consta de 24 telescopios FD agrupados en unidades de 6, situados en 4 estaciones o edificios que reciben el nombre de una pequeña loma sobre la que se encuentran: Los Leones, Los Morados, Loma Amarilla y Coihueco, de manera que es posible observar distintas regiones de la atmósfera. En conjunto los 6 telescopios logran

cubrir un rango de visión de 180° en el azimut.

Estos telescopios captan la luz de fluorescencia que se produce durante el desarrollo de una cascada atmosférica. Son capaces de observar directamente el desarrollo longitudinal de la lluvia con lo cual se puede determinar la energía colorimétrica de la cascada y medir directamente X_{max} .



Figura 2.3: Edificio FD en la estación Los Leones, detrás del edificio hay una torre de comunicación. La fotografía fue tomada cuando se abrieron las persianas por mantenimiento. Imagen tomada de [16].

2.2.1. Luz de fluorescencia

Durante el desarrollo de la cascada las partículas secundarias interaccionan con las moléculas presentes en la atmósfera depositando energía en esta mediante colisiones inelásticas, parte de esta energía es emitida en forma de luz fluorescente.

El número de fotones N_γ por profundidad atmosférica X que puede detectar un telescopio de fluorescencia está dado por

$$\frac{dN_\gamma}{dX} = \frac{dE_{tot.}}{dX} \cdot \int f(\lambda, p, T) \tau(\lambda, X) \epsilon_{det}(\lambda, X) d\lambda \quad (2.1)$$

Donde ϵ_{det} es la eficiencia del detector τ la transmitancia óptica de la atmósfera y f el rendimiento de fluorescencia, λ , p y T denotan las dependencias de la longitud de onda, la presión y la temperatura, podemos notar que la intensidad de la luz emitida es proporcional a la energía disipada por la lluvia [7].

2.3. Sistema óptico

Los telescopios de fluorescencia hacen uso del sistema óptico de *Schmidt* el cual consiste en un diafragma circular de 1.1 m de radio colocado en el centro de la curvatura de un espejo esférico en la apertura del sistema óptico, que es por donde ingresa la luz de fluorescencia, cuenta con un filtro UV que suprime la luz de fondo. Cuando la luz de fluorescencia atraviesa la apertura es captada

por un espejo esférico de 3.5 m x 3.5 m que concentra la luz hacia la cámara, está dividido en segmentos sobre soportes ajustables que permiten alinearlo. En Los Leones y Los Morados cada espejo se compone de 36 segmentos cuadrados de aluminio, mientras que en Coihueco y Loma Amarilla se componen por 60 segmentos hexagonales de vidrio de alta calidad, en ambos casos la reflectividad es casi del 90 %. Finalmente la cámara consta de un arreglo de 440 píxeles hexagonales que cuentan con un tubo fotomultiplicador Photonis XP3062 ordenados en 20 filas y 20 columnas [7].

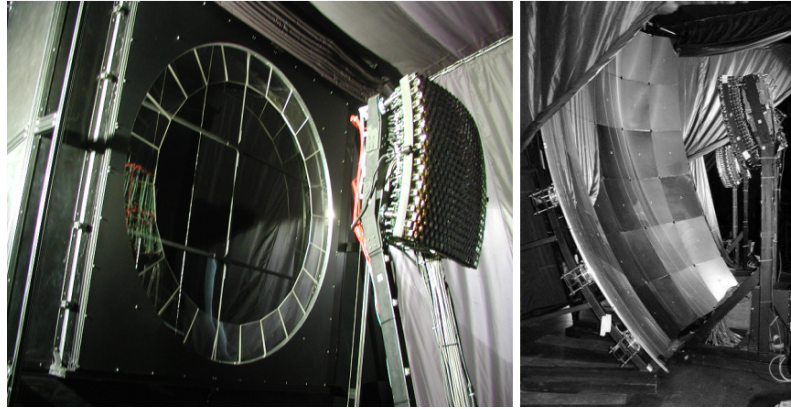


Figura 2.4: Fotografías de la cámara, el espejo y la apertura de un telescopio de fluorescencia. Imagen tomada de [7].

La razón por la que el FD solo tiene un ciclo útil de trabajo del 15 % es que la intensidad de la luz de fluorescencia es muy débil y solo se puede garantizar la calidad del muestreo en las condiciones de noches claras y sin luna.

2.4. AERA

Auger Engineering Radio Array o AERA es una mejora del Observatorio Pierre Auger que consta de más de 150 estaciones de antenas de radio puestas en forma de cuadrícula con diferentes espaciamientos entre sí. Las antenas de radio permiten medir la emisión de radio producida por una cascada atmosférica que se origina por la deflexión geomagnética de los electrones y positrones de la cascada y son sensibles a la componente electromagnética de la lluvia, lo cual las hace útiles para el estudio de la energía y composición de la partícula primaria [56].

Las señales de radio recolectadas por estas antenas otorgan una alta precisión sobre mediciones de la dirección y energía de las lluvias de partículas. Además, dado que estas antenas son sensibles a la componente electromagnética de las cascadas podemos obtener información sobre la profundidad máxima de la lluvia y de la masa de la partícula primaria.

2.5. Extensiones al Observatorio Pierre Auger

El Observatorio Pierre Auger fue inicialmente construido para estudiar partículas de muy alta energía con más de 3×10^{18} eV, sin embargo se han implementado algunas regiones dentro del observatorio que permiten extender el rango de energía para el estudio de rayos cósmicos a energías del orden correspondiente a la zona de la segunda rodilla. El edificio HEAT (High Elevation Auger Telescopes) consiste en tres telescopios de fluorescencia con un rango de visión de 30° en elevación de manera muy similar y con el mismo funcionamiento que las otras estaciones de FD, la diferencia

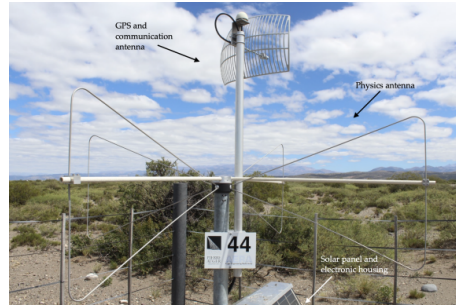


Figura 2.5: Fotografía de una estación de radio que forma parte de AERA. Imagen tomada de [59].

radica en que el edificio cuenta con un sistema de elevación que permite inclinar a los telescopios hasta un ángulo de 29° sobre la horizontal con lo cual se extiende el rango de visión y esto permite observar el desarrollo de cascadas cuya energía corresponde al rango ya mencionado.

Infill es la parte del arreglo de detectores de superficie cuya separación es de 750 metros entre sí y es parte del proyecto AMIGA que abarca a Infill junto con una segunda región donde los detectores de superficie tienen una separación de 433 metros en un área de 2 km^2 . Esta región, al igual que HEAT, permite el estudio de rayos cósmicos a una energía menor que la que estudia el resto del área del observatorio. Infill se encuentra sobre el área de visión de HEAT con el propósito de utilizar los datos recopilados por ambos sistemas y lograr una reconstrucción más precisa. Además AMIGA cuenta con detectores de muones, cada detector consta de tres módulos de 10 m^2 segmentados en 64 tiras de plástico centellador, con los cuales es posible identificar la masa de los rayos cósmicos en la región de la segunda rodilla.

AugerPrime (Primary cosmic Ray Identification with Muons and Electrons) es otra actualización del Observatorio Pierre Auger que está en desarrollo, la cual busca mejorar las mediciones de la composición de masa de los rayos cósmicos a energías mayores a $10^{19.5} \text{ eV}$, con el fin de proporcionar más información del desarrollo de las cascadas atmosféricas extensas a estas energías. Entre otras cosas esto ayudará a mejorar nuestro conocimiento de las interacciones hadrónicas aumentando la precisión de los modelos existentes.

AugerPrime contempla actualizar los detectores de superficie colocando detectores de centelleo de superficie y un detector de radio, además de proporcionarle una nueva electrónica y se busca aumentar el ciclo de trabajo de los detectores de fluorescencia. Finalmente, AugerPrime se complementará con un detector subterráneo de muones.



Figura 2.6: Detector de superficie actualizado, posee un detector de centelleo y una antena de radio en la parte superior [57].

Capítulo 3

Simulaciones de cascadas de partículas

Con el fin de entrenar un modelo de aprendizaje automático es necesario contar con una gran cantidad de datos que contengan una muestra representativa del comportamiento de las cascadas, esta colección de datos debe ser suficientemente grande para entrenar a la red y posteriormente probar su rendimiento.

Con esto en mente utilizamos el software CONEX y el clúster de computadoras "*Cuetlaxcoapan*" del *Laboratorio Nacional de Supercómputo del sureste de México* para realizar simulaciones de Monte-Carlo del desarrollo de las cascadas atmosféricas extensas con tres modelos hadrónicos y dos primarios distintos.

Posteriormente hicimos uso de Offline que es el software principal utilizado para la reconstrucción de los datos obtenidos en el Observatorio Pierre Auger, estas nuevas simulaciones se conocen con el nombre de Real Monte-Carlo.

3.1. Simulaciones de Monte-Carlo

Si quisiéramos modelar el desarrollo de una cascada de partículas necesitamos tener en cuenta los fenómenos que ocurren durante su desarrollo como las interacciones hadrónicas y electromagnéticas entre las partículas de la cascada y las moléculas presentes en la atmósfera, el decaimiento de las partículas inestables, la energía perdida, deflexión, etcétera. Si conociéramos estas propiedades en cualquier rango de energía podríamos predecir el comportamiento de una CAE a través de simulaciones numéricas, sin embargo, en la práctica resulta difícil conocerlas con detalle. Por suerte existe lo que se conoce como *simulaciones de Monte-Carlo (MC)* las cuales están relacionadas con experimentos aleatorios donde los resultados no se conocen de antemano. Se basan en muestreos aleatorios repetidos y análisis estadísticos para calcular resultados, los números aleatorios son utilizados para modelar reacciones que pueden tener varios resultados. Las simulaciones de Monte-Carlo conducen a fluctuaciones estadísticas de manera similar a los resultados que se obtienen en los experimentos por lo que gracias a estas simulaciones es posible obtener modelos realistas de las cascadas provocadas por un rayo cósmico.

3.1.1. Modelos de cascadas de partículas y CONEX

Si deseamos construir modelos que se ajusten a los datos observados necesitamos conocer diferentes aspectos del desarrollo de las cascadas como la composición del medio de interacción (la atmósfera terrestre), que partícula primaria provoca la lluvia y que secundarios se generan, también debe tenerse en cuenta el campo magnético terrestre, la pérdida de energía, absorción, ionización y

producción de luz Cherenkov. La mayoría de estas variables son conocidas, de esta forma la técnica de Monte-Carlo nos proporciona un método capaz de simular todo el proceso de una cascada. Sin embargo, el análisis se complica debido que a altas energías las interacciones hadrónicas y nucleares presentan grandes incertidumbres. No es posible alcanzar estas energías en los laboratorios y no existe una teoría fundamental de la que se pueda extrapolar a energías más altas, lo que sí tenemos son modelos hadrónicos que se ajustan a los datos en colisionadores artificiales y a los registrados por observatorios como Pierre Auger.

CONEX es un programa informático especializado en la simulación de cascadas atmosféricas extensas iniciadas por un rayo cósmico de alta energía, en el cual es posible utilizar como primarios protones, núcleos ligeros hasta el hierro, fotones y muchas otras partículas; CONEX simula el paso de las partículas a través de la atmósfera hasta que interactúan con los núcleos de aire o, en el caso de los secundarios inestables, se descomponen. Para abordar el problema relacionado con las interacciones hadrónicas a altas energías es posible implementar varios modelos hadrónicos como VENUS, QGSJET y DPMJET, SIBYLL, neXus, EPOS, HDPM, FLUKA y UrQMD [27].

En este trabajo nos centramos en los modelos de interacción hadrónica SIBYLL, QGSJETII-04 y EPOS LHC.

3.1.2. Aspectos de la simulación de una cascada de partículas

Números aleatorios: La generación de números aleatorios son una pieza fundamental en las simulaciones de Monte-Carlo que son clave en la simulación de interacciones, descomposición y transporte de partículas.

Atmósfera: La atmósfera está compuesta por 78 % de nitrógeno, 21.5 % de oxígeno y 0.5 % de argón, los demás elementos en el aire se desprecian ya que solo forman el 1 % del total de su composición. La densidad atmosférica está dada por $\rho(h) \approx \rho(0)e^{-h/h_0}$, donde $\rho(0) = 0.000123 \text{ g/cm}^3$ y al nivel del mar $h_0 = 7 \text{ km}$. Al nivel del mar la profundidad atmosférica es de 1000 g/cm^2 .

Decaimiento e interacción: En general es difícil determinar si una partícula experimentará un decaimiento o por el contrario logra interactuar con otra; el método de Monte-Carlo calcula de forma independiente una colisión y una descomposición y selecciona la que ocurre primero.

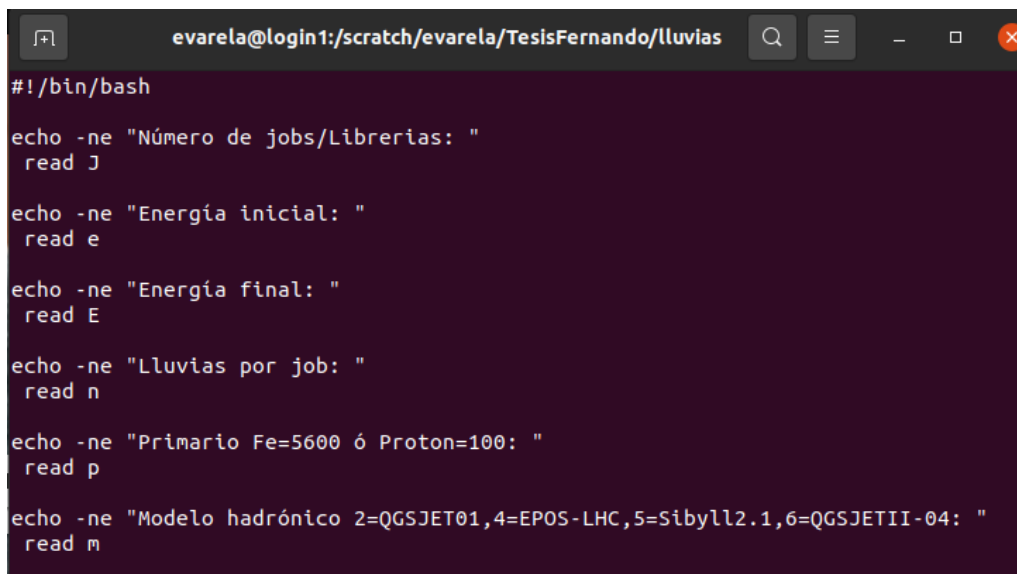
3.2. Producción de los eventos simulados

Para obtener nuestro conjunto de datos realizamos simulaciones de Monte-Carlo utilizando el software *CONEX v4r37*, como primarios utilizamos partículas de protón (p) y núcleos de hierro (Fe), como modelos de interacción de partículas (modelos hadrónicos) elegimos SIBYLL, QGSJETII-04 y EPOS LHC cubriendo un rango de energía en logaritmo en base 10 (medida en eV) desde 16.875 hasta 18.625. Las simulaciones de Monte-Carlo consumen muchos recursos computacionales y requieren de un gran tiempo de procesamiento, por esta razón necesitamos hacer uso de una computadora de alto rendimiento lo que nos condujo a utilizar el clúster de computadoras *Cuetlaxcoapan* del *Laboratorio Nacional de Supercómputo del sureste de México* para llevar a cabo dichas simulaciones, aun así ejecutar las simulaciones cubriendo todo el intervalo de energía deseado requiere mucho tiempo, de esta forma y con el fin de optimizar el tiempo de procesamiento y recursos computacionales decidimos dividirlo en siete intervalos separados por $\text{Log}[\text{Energía/eV}] = 0.25$. En los

primeros intervalos ($16.875 - 17.125$, $17.125 - 17.375$, $17.375 - 17.625$), el tiempo aproximado que tomó procesar cada uno de ellos fue de una semana mientras que para los siguientes intervalos ($17.625 - 17.875$, $17.875 - 18.125$, $18.125 - 18.375$, $18.375 - 18.625$) el tiempo aproximado fue de dos o incluso tres semanas ya que conforme la energía aumenta el tiempo de procesamiento también incrementa, notamos que en general las simulaciones que tienen como primario núcleos de hierro tardan más en ser completadas.

Creamos un script en bash que automatiza el proceso para ejecutar las simulaciones, el cual al ser ejecutado nos pregunta acerca de las características que deseamos simular como la energía inicial y final, número de lluvias, primario y modelo hadrónico y estos valores los almacena en una variable.

Realizamos un total de 100 Jobs con 10000 lluvias en cada subintervalo de energía, es decir, obtuvimos un total de 1 millón de simulaciones por cada subintervalo de energía por primario y modelo hadrónico.



```
evarela@login1:scratch/evarela/TesisFernando/lluvias

#!/bin/bash

echo -ne "Número de jobs/Librerías: "
read J

echo -ne "Energía inicial: "
read e

echo -ne "Energía final: "
read E

echo -ne "Lluvias por job: "
read n

echo -ne "Primario Fe=5600 ó Proton=100: "
read p

echo -ne "Modelo hadrónico 2=QGSJET01,4=EPOS-LHC,5=Sibyll2.1,6=QGSJETII-04: "
read m
```

Figura 3.1: Fragmento del script.

mandamos a llamar al software CONEX

```
echo 'module load applications/conex/conex2r4.37'
```

y el script lanza las lluvias con las características proporcionadas

```
echo '$CONEX_RUN' "-e $e -E -z 0 -Z 65 -n $n -p $p -x  
$e-$E-$n-m $m"
```

Una vez que tenemos las simulaciones es necesario unir las todas en un solo archivo, ya que como mencionamos separamos las lluvias en siete subintervalos cubriendo un rango de energía específico con el fin de optimizar el tiempo requerido para las simulaciones. Para lo cual hacemos uso de la función hadd:

```
module load applications/root/root-5.34.26  
hadd NuevoArchivo.root TodosLosIntervalos*.root
```

Los datos son leídos en el software **ROOT** donde es posible manipular y visualizar los datos contenidos en nuestros archivos generados por las simulaciones [48], sin embargo, a pesar de que ROOT es un software muy completo en el que es posible llevar a cabo el análisis y procesamiento de datos e incluso cuenta con su propia paquetería de aprendizaje automático TMVA, el uso de este se torna difícil por los conocimientos de la sintaxis de lenguajes de programación que hay que poseer para manipularlo pero más importante, el tiempo de cómputo es mayor comparado con otras herramientas, por lo cual optamos por extraer la información relevante utilizando la librería de Python **Uproot** que puede leer y escribir archivos en formato .root. De esta forma podemos incrementar la velocidad de procesamiento de los datos de nuestras simulaciones además de que podemos aprovechar todas las herramientas de Python para análisis de datos (como Pandas) y aprendizaje automático [41].

3.3. Offline

Offline es el software utilizado para la reconstrucción de las cascadas atmosféricas en el Observatorio Pierre Auger, permite la reconstrucción y simulación de eventos de detectores de superficie, de fluorescencia y eventos híbridos, así como la simulación de técnicas de calibración y otras tareas auxiliares.

Offline fue diseñado para adaptarse a futuras actualizaciones en la instrumentación del observatorio, emplea una serie de formatos de datos para manejar la información de monitoreo.

Es posible realizar una reconstrucción de los eventos tanto de datos reales como de aquellos realizados en simulaciones de Monte-Carlo, durante la reconstrucción sigue una jerarquía que está asociada a los detectores del observatorio.

Utilizamos este software para realizar un nuevo conjunto de simulaciones utilizando como datos de entrada para su reconstrucción las simulaciones realizadas en CONEX con el modelo QGSJETII-04 con ambos primarios: protón y núcleos de hierro. Estas nuevas simulaciones son denominadas Real Monte-Carlo porque reconstruyen los datos proporcionados como si hubiesen sido detectados por los detectores del Observatorio Pierre Auger haciendo uso de un *DataSet* que contiene información real acerca de las condiciones meteorológicas, tiempos de llegada, etcétera, con el fin de acercar lo más posible la reconstrucción de datos simulados a la realidad.

Capítulo 4

Aprendizaje automático y redes neuronales artificiales

El *aprendizaje automático* o *machine learning* (ML) es el campo de investigación dedicado al estudio de los sistemas de aprendizaje, es un subcampo de las ciencias de la computación y la rama de la inteligencia artificial que tiene como objetivo que los algoritmos desarrollen técnicas para resolver un problema sin que la máquina sea explícitamente programada para ello, es decir, en lugar de codificar algoritmos que resuelvan un problema en concreto el aprendizaje automático tiene como propósito hallar relaciones y patrones a partir de ejemplos y observaciones, así como de recompensas (aprendizaje reforzado), esto es posible gracias a algoritmos que mediante iteraciones logran aprender el comportamiento de los datos proporcionados para un problema en específico, lo cual brinda la capacidad de encontrar patrones complejos sin haberlos especificado en el programa. El aprendizaje automático es un campo interdisciplinario que hace uso de la estadística, la informática, la ingeniería, teoría de optimización y otras disciplinas de la ciencia y las matemáticas, los algoritmos basados en el aprendizaje automático han demostrado ser exitosos al momento de abordar distintos problemas como:

- Problemas donde es necesario codificar una larga secuencia de reglas y algoritmos complicados. Un algoritmo de aprendizaje automático tiene la capacidad de simplificar el código.
- Problemas complejos donde no existe una solución general con métodos tradicionales.
- Entornos fluctuantes: un sistema de aprendizaje automático es capaz de adaptarse a nuevos datos.
- Descifrar relaciones e identificar patrones en grandes cantidades de datos y problemas no triviales.
- Problemas de la visión artificial, procesamiento de lenguaje natural, máquinas autónomas, entre otros problemas de la ingeniería, así como de las ciencias fundamentales.

4.1. Tipos de aprendizaje automático

Existen tres principales tipos de aprendizaje automático clasificados de acuerdo con el tipo de supervisión que reciben durante el entrenamiento.

Para explicar estos diferentes tipos de aprendizaje automático consideremos una máquina a la que se le proporciona una secuencia de valores de entrada $x_1, x_2, \dots, x_n$, que corresponden a los datos de entrenamiento, pueden ser cualquier tipo de información como imágenes de retinas, los píxeles de una cámara, una forma de onda de sonido, la lista de los artículos para comprar en el supermercado, etcétera.

Los describimos a continuación:

■ Aprendizaje supervisado

En el aprendizaje supervisado se le proporciona a la máquina, junto con los datos ya mencionados, una secuencia de datos objetivo o valores deseados $y_1, y_2, \dots, y_n$, la meta es encontrar relaciones entre los datos de entrada que conduzcan al valor deseado, es decir, tenemos un mapeo entre un conjunto de variables de entrada (llamémosle X) y una variable de salida (Y), los datos de entrada X alimentan a la red neuronal encontrando relaciones entre estos para predecir el valor de salida Y.

■ Aprendizaje reforzado

En el aprendizaje reforzado (reinforcement learning) la máquina interactúa con el ambiente produciendo acciones $a_1, a_2, \dots$ y recibe un escalar que corresponde a una recompensa o a un castigo, $r_1, r_2, \dots$, la meta en este tipo de aprendizaje es que la máquina aprenda a maximizar las recompensas y minimizar los castigos. Pensemos en el juego de “caliente-frío”, en donde alguien esconde algún objeto en una habitación, nuestra meta es encontrarlo, cuando estamos cerca del objeto la persona que lo escondió previamente nos advierte diciendo la palabra “caliente”, cuando nos alejamos del sitio donde se encuentra el objeto la persona mencionara la palabra “frío”, de esta forma aprendemos cuales son los sitios más probables en donde se encuentra el objeto hasta lograr hallarlo.

■ Aprendizaje no supervisado

En este tipo de aprendizaje se le proporciona a la máquina los datos de entrada $x_1, x_2, \dots, x_n$ pero no tenemos valores objetivo ni recompensas del entorno. El objetivo es que la máquina logre aprender relaciones entre los datos, incluso entre lo que normalmente se consideraría como ruido.

En esta tesis nos enfocamos en el aprendizaje supervisado.

4.2. Aprendizaje automático supervisado

Recibe este nombre porque tenemos un conjunto de datos que ha sido etiquetado y clasificado, es decir, conocemos el valor de salida que le corresponde a cada uno de los ejemplos en el conjunto de entrenamiento o entrada.

El objetivo es generar un modelo capaz de predecir las salidas a partir de un mapeo entre un conjunto de entrada $\mathbf{X}$ y un conjunto de salida o valores deseados que comúnmente se les da el nombre de etiqueta $\mathbf{Y}$, durante el entrenamiento se proporcionan ambos conjuntos de esta forma el algoritmo va aprendiendo a clasificar o hallar relaciones comparando el resultado predicho con el valor real de la etiqueta $\mathbf{Y}$. Una vez finalizada la fase de entrenamiento se busca que la máquina sea capaz de predecir correctamente el valor deseado proporcionándole únicamente los datos de entrada $\mathbf{X}$.

El tipo de problemas que pueden resolverse empleando ese tipo de aprendizaje corresponde tanto a problemas de clasificación como a problemas de regresión. En este trabajo utilizamos este enfoque para resolver un problema de regresión.

4.3. Redes neuronales artificiales

Las redes neuronales artificiales (RNA), fueron creadas tomando inspiración en la arquitectura del cerebro con la intención de construir una máquina inteligente, sin embargo esto no significa que las RNA funcionen de la misma manera en la que nuestro cerebro lo hace ya que desconocemos mucho acerca del funcionamiento de este último, es por esto que algunos investigadores argumentan que deberíamos abandonar esta analogía biológica y llamar a las componentes de la red neuronal artificial unidades en lugar de neuronas.

Las RNA son potentes y capaces de adaptarse con rapidez a diversas funciones lo que las hace ideales para ser utilizadas en tareas complicadas, superando con frecuencia a otras técnicas de aprendizaje automático en este aspecto. Cabe mencionar que su éxito se debe, en gran parte, al avance tecnológico que hemos experimentado en las últimas décadas, gracias al cual es posible acceder a grandes conjuntos de datos, cosa que ha facilitado enormemente el desarrollo de las RNA, además gracias a la ley de Moore el poder computacional ha incrementado en forma exponencial y gracias a la industria de los videojuegos se han desarrollado unidades de procesamiento gráfico (GPU) que han resultado ser poderosas herramientas para el entrenamiento de redes neuronales artificiales.

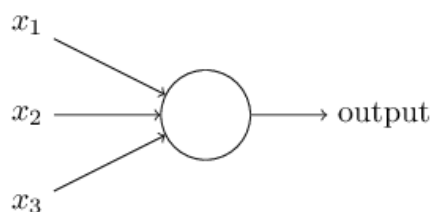
Se puede entender al proceso realizado por una RNA como la respuesta a la pregunta: ¿Cómo puedo usar $\mathbf{X}$ para predecir $\mathbf{Y}$? De esta forma las redes neuronales artificiales pueden entenderse como funciones

$$\mathbf{y} = f(\mathbf{x}) \tag{4.1}$$

Las redes neuronales también son conocidas como aproximadores de funciones universales ya que pueden calcular cualquier función que se les proporcione, esto es, cualquier proceso imaginable se puede resolver mediante el uso de redes neuronales artificiales.

4.3.1. El perceptrón

El perceptrón fue desarrollado por Frank Rosenblatt quien se inspiró en los trabajos previos de Warren McCulloch y Walter Pitts. El perceptrón es la componente principal y más simple de una red neuronal artificial, su funcionamiento está basado en lo que se conoce como unidad lógica de umbral (TLU) o unidad de umbral lineal (LTU), los cuales actúan como dispositivos a los que se le introduce un valor de entrada y devuelven uno o más valores de salidas, se conectan entre sí y cada conexión entre ellos está asociada a un peso w_i que son números reales cuya finalidad es expresar la importancia de las entradas con respecto de la salida, mientras más grande sea el peso de una entrada significa que dicho valor de entrada está más relacionado con la salida [32].



Para comprender mejor esto consideremos la situación en la que cada ejemplo de entrenamiento es de la forma $(\bar{X}, y)$ donde $\bar{X} = [x_1, \dots, x_n]$ contiene n variables como datos de entrada y $y = [y_1, \dots, y_n]$ son los valores deseados para cada ejemplo. La meta es predecir y en base a los datos de entrada.

La capa de entrada de la RNA tendrá n nodos, uno para cada característica de los datos de entrada y a cada una se le asocia un peso $\bar{W} = [w_1, \dots, w_n]$.

El perceptrón calcula una suma ponderada de las entradas $z = \sum_{i=1}^n w_i x_i = w_1 x_1 + w_2 x_2 + \dots + w_n x_n$, donde w_i corresponde a los pesos de conexión y x_i es el i -ésimo valor de entrada correspondiente a los datos de entrenamiento. Luego se aplica una *función de paso* a esa suma $h_w(x) = h_x(z)$ y el resultado final se determina considerando si el valor de esta suma es mayor o menor que un valor de umbral:

$$salida = \begin{cases} 0 & \text{si } z \leq \text{valor de umbral} \\ 1 & \text{si } z > \text{valor de umbral} \end{cases} \quad (4.2)$$

El perceptrón puede ser visto como un dispositivo que toma decisiones en base a la ponderación de los datos disponibles.

El perceptrón se compone de una sola unidad conectada a todas las entradas, generalmente se agrega una función de *sesgo* adicional ($x_0 = 1$) que se representa usando una neurona que genera 1 todo el tiempo. De esta forma es posible desarrollar perceptrones que realicen tareas de clasificación, por ejemplo, si el valor de la suma ponderada sobrepasa o no un valor de umbral el perceptrón decidirá si una imagen corresponde a un perro o un gato. Además de esto podemos hacer que un perceptrón de como resultado un número real, siguiendo un procedimiento similar, necesitamos

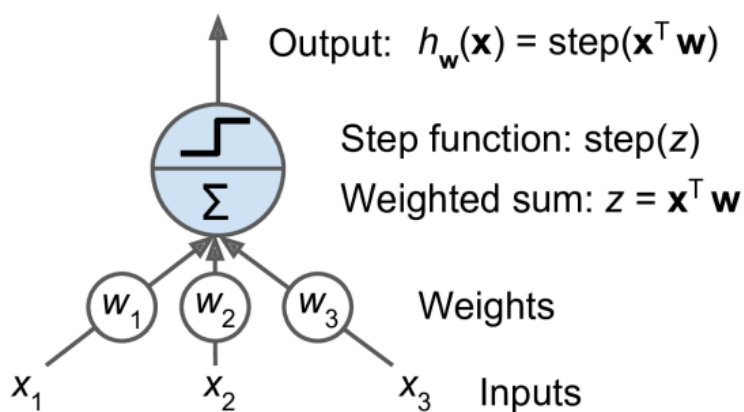


Figura 4.1: Arquitectura de un perceptrón; los datos de entrada son multiplicados por los pesos w_i , se realiza una suma ponderada y se aplica una función de paso dando lugar al valor de salida.

proporcionar datos de entrada $x_1, x_2, \dots, x_n$ y cada entrada a su vez tendrá asociado un peso $w_1, w_2, \dots, w_n$ que generalmente se escogen aleatoriamente y se agrega un valor de sesgo, de la misma manera se realiza una suma ponderada $\sum_{i=1}^n w_i x_i = w_1 x_1 + w_1 x_2 + \dots + w_n x_n$ y finalmente se suma el valor de sesgo, el resultado de estas operaciones es la salida mostrada por el perceptrón.

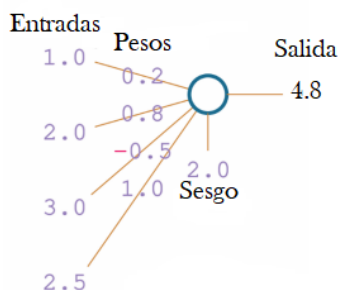


Figura 4.2: Visualización de un perceptrón con entradas (1.0, 2.0, 3.0, 2.5) asociadas con sus respectivos pesos (0.2, 0.8, -0.5, 1.0) además de un sesgo con valor (2.0). El resultado de computar la suma ponderada y añadirle el valor de sesgo es 4.8.

4.3.2. Redes neuronales multicapa.

El perceptrón posee una sola capa de entrada que es donde se proporcionan los valores de entrada y consta de una sola capa de salida. La capa de entrada transmite los datos a la capa de salida y todos los cálculos son visibles para el usuario. Sin embargo, las redes neuronales artificiales contienen múltiples capas computacionales que consisten en más de un perceptrón agrupados en columnas y conectándose con

las capas siguientes de forma paralela. Las capas intermedias adicionales (entre la entrada y la salida) se conocen como capas ocultas porque los cálculos realizados no son visibles para el usuario, en este sentido se dice que son cajas negras. Cada capa se puede representar mediante un vector, que se obtiene multiplicando la capa anterior por una matriz de peso, más un vector de sesgo y al resultado se le aplica alguna función no lineal conocida como *función de activación*.

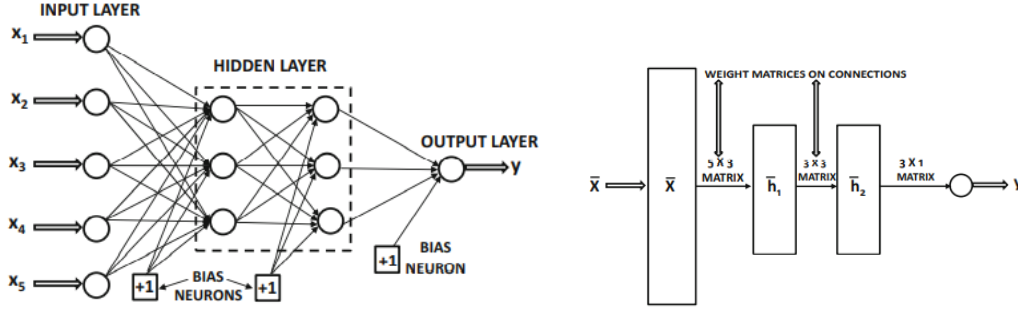


Figura 4.3: Arquitectura básica de una red neuronal multicapa. La red tiene dos capas ocultas y una de salida. Se muestra la red con los perceptrones y la representación vectorial de la misma.

En la figura 4.3 podemos notar que cada perceptrón de cada capa está conectado a cada perceptrón de la capa anterior, cuando esto ocurre se dice que es una red neuronal totalmente conectada, lo cual es muy común en las redes neuronales multicapa, sin embargo no es necesario que esto ocurra siempre.

4.4. Algoritmo de propagación hacia atrás

Durante muchos años se ignoró el potencial de las redes neuronales multicapa ya que no existía una forma adecuada de entrenarlas hasta que en 1986 David Rumelhart, Geoffrey Hinton y Ronald Williams publicaron un artículo en el que presentaban el algoritmo de entrenamiento de propagación hacia atrás [52].

Para entender cómo surge este algoritmo primero notemos que deseamos hallar los valores de los pesos y sesgos que den como resultado un valor lo más cercano posible al valor deseado; lo cual nos lleva inmediatamente a la necesidad de cuantificar que tan buenas (o malas) son las predicciones de la red, para esto se hace uso de lo que se conoce como *función de pérdida*, de la cual hablaremos más adelante, por el momento usaremos una función arbitraria y nos concentramos en el algoritmo de propagación hacia atrás.

Para comenzar necesitaremos conocer el algoritmo de descenso de gradiente. Sea una función C de m variables $v_1, \dots, v_m$, deseamos encontrar un mínimo de esta función, consideremos un cambio en C , ΔC , que es producido por pequeños cambios $\Delta v = (\Delta v_1, \dots, \Delta v_m)$ en la direcciones $v_1, \dots, v_m$, respectivamente, entonces tenemos que

$$\Delta C \approx \frac{\partial C}{\partial v_1} \Delta v_1 + \dots + \frac{\partial C}{\partial v_m} \Delta v_m \quad (4.3)$$

Definamos estos cambios en forma vectorial de la siguiente manera: sea

$$\Delta v = (\Delta v_1, \dots, \Delta v_m)^T \quad (4.4)$$

y el gradiente de C como el vector de las derivadas parciales

$$\nabla C = \left(\frac{\partial C}{\partial v_1}, \dots, \frac{\partial C}{\partial v_m} \right) \quad (4.5)$$

Entonces podemos escribir los cambios en C como

$$\Delta C \approx \nabla C \cdot \Delta v \quad (4.6)$$

En particular escogemos $\Delta v = -\eta \nabla C$, donde η se conoce como tasa de aprendizaje. Notemos que $\Delta C \approx -\eta \nabla C \cdot \nabla C = -\eta \|\nabla C\|^2$, como $\|\nabla C\|^2 \geq 0$, entonces $\Delta C < 0$, lo cual significa que siempre decrece.

Luego, $v \rightarrow v' = v - \eta \nabla C$

Aplicando esta regla una y otra vez podemos lograr alcanzar un mínimo de C .

Ahora, sea w_{jk}^l el peso de conexión del k -ésimo perceptrón en la capa $l-1$ con el j -ésimo perceptrón en la capa l , b_j^l el sesgo del j -ésimo perceptrón en la capa l y a_j^l la activación del j -ésimo perceptrón en la capa l , dicha activación es el resultado de aplicar una función de activación en la salida de cada perceptrón: $a_j^l = \sigma(\sum_k w_{jk}^l a_k^{l-1} + b_j^l)$

Donde la suma corre sobre los perceptrones k en la capa $l-1$.

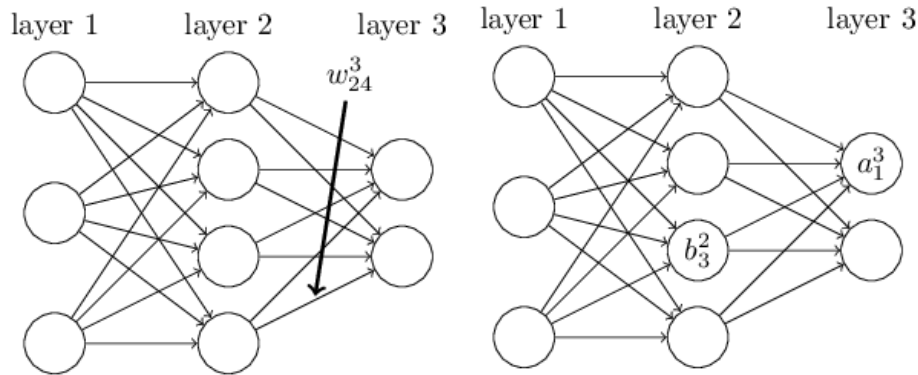


Figura 4.4: Ilustración gráfica de la notación usada para los pesos, sesgos y activaciones en un perceptrón en diferentes capas.

Sea una red neuronal con n capas y pesos definidos, supongamos que hacemos un pequeño cambio en uno de estos pesos Δw_{jk}^l en la red, dicho cambio a su vez ocasionará un cambio en la salida de activación a_j^l en el correspondiente perceptrón, esto ocasionará un cambio en todas las salidas de activación en la siguiente capa y estos a su vez producirán cambios en la siguiente capa y así hasta la salida final, es decir, ocasionará un cambio en la función C , ΔC .

El cambio en ΔC está relacionado con el cambio en el peso Δw_{jk}^l de la forma

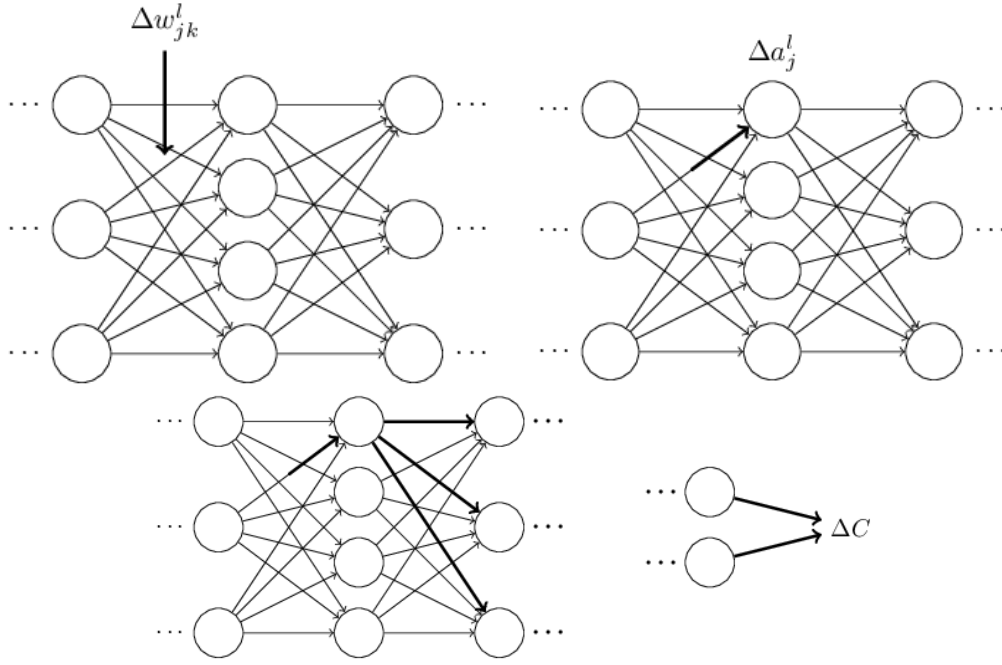


Figura 4.5: Imagen ilustrativa de cómo un cambio en un peso afecta al perceptrón al que está conectado ocasionando un cambio en su salida de activación, este a su vez afecta a todas las salidas de activación en la siguiente capa y este cambio se propaga hasta la salida ocasionando, finalmente, un cambio en C .

$$\Delta C \approx \frac{\partial C}{\partial w_{jk}^l} \Delta w_{jk}^l \quad (4.7)$$

El enfoque que adoptaremos será observar la forma en cómo el cambio en un peso, Δw_{jk}^l , se propaga a través de la red ocasionando un cambio en la función C , ΔC .

De esta forma fijemos nuestra atención en cómo el cambio Δw_{jk}^l ocasiona un cambio en la activación del j -ésimo perceptrón en la capa l , a_j^l , el cual está dado por

$$\Delta a_j^l \approx \frac{\partial a_j^l}{\partial w_{jk}^l} \Delta w_{jk}^l \quad (4.8)$$

Luego, este cambio causará otro cambio en todas las salidas de activación de la siguiente capa, $l + 1$. Concentrémonos solamente en una de estas salidas afectadas: a_q^{l+1} , la cual está relacionada con el cambio de a_j^l de la forma

$$\Delta a_q^{l+1} \approx \frac{\partial a_q^{l+1}}{\partial a_j^l} \Delta a_j^l \quad (4.9)$$

Haciendo uso de la ecuación (4.8) tenemos que

$$\Delta a_q^{l+1} \approx \frac{\partial a_q^{l+1}}{\partial a_j^l} \frac{\partial a_j^l}{\partial w_{jk}^l} \Delta w_{jk}^l \quad (4.10)$$

Como ya mencionamos esto causará un cambio en las salidas de activación de la siguiente capa y estos cambios modifican las salidas de la siguiente capa, continuando esto hasta la salida final. Siguiendo cuidadosamente las salidas $a_j^l, a_q^{l+1}, \dots, a_n^{L-1}, a_m^L$ tendremos que

$$\Delta C \approx \frac{\partial C}{\partial a_m^L} \frac{\partial a_m^L}{\partial a_n^{L-1}} \frac{\partial a_n^{L-1}}{\partial a_p^{L-2}} \cdots \frac{\partial a_q^{l+1}}{\partial a_j^l} \frac{\partial a_j^l}{\partial w_{jk}^l} \Delta w_{jk}^l \quad (4.11)$$

Esto es el cambio en C debido a los cambios del camino que hemos seguido dentro de la red, sin embargo, el cambio en el peso se propaga a través de todos los caminos posibles, de esta manera tenemos que

$$\Delta C \approx \sum_{mnp \dots q} \frac{\partial C}{\partial a_m^L} \frac{\partial a_m^L}{\partial a_n^{L-1}} \frac{\partial a_n^{L-1}}{\partial a_p^{L-2}} \cdots \frac{\partial a_q^{l+1}}{\partial a_j^l} \frac{\partial a_j^l}{\partial w_{jk}^l} \Delta w_{jk}^l \quad (4.12)$$

El siguiente paso es escribir una expresión explícita para todas las derivadas parciales individuales de la ecuación anterior, después escribir las sumas sobre los índices como multiplicaciones de matrices, hecho esto y simplificando tanto como sea posible terminamos con el algoritmo que se describe a continuación:

1. **Ingresar un conjunto de ejemplos de entrenamiento.**

2. **Para cada ejemplo de entrenamiento, x , realizamos los siguientes pasos:**

◦ **Propagación hacia adelante:** para cada $l = 2, 3, \dots, L$, donde l indica el número de capas, calculamos

$$z^{x,l} = w^l a^{x,l-1} + b^l \text{ y } a^{x,l} = \sigma(z^{x,l})$$

◦ **Calculamos el error:** Definimos el error como

$$\delta^{x,L} = \nabla_a C_x \sigma'(z^{x,L})$$

Donde $\nabla_a C_x$ es un vector cuyas componentes son las derivadas parciales $\frac{\partial C}{\partial a_j^L}$, el término $\sigma'(z^{x,L})$ mide el cambio en la función σ' en $(z^{x,L})$

◦ **Propagar hacia atrás el error:**

Para cada $l = L-1, L-2, \dots, 2$ calculamos

$$\delta^{x,l} = ((w^{l+1})^T \delta^{x,l+1}) \sigma'(z^{x,l})$$

3. **Descenso del gradiente:**

Aquí es donde la discusión del algoritmo del *descenso del gradiente* interviene.

Recordemos que una variable v en C decrece de la forma

$$v \rightarrow v' = v - \eta \nabla C$$

De esta manera basta con que C sea una función que tiene componentes w_k y b_l entonces tenemos una forma para minimizar los pesos y los valores de sesgo

$$w_k \rightarrow w'_k = w_k - \eta \nabla C \quad (4.13)$$

y

$$b_l \rightarrow b'_l = b_l - \eta \nabla C \quad (4.14)$$

Donde ∇C tiene las componentes correspondientes $\frac{\partial C}{\partial w_k}$ y $\frac{\partial C}{\partial b_l}$.

Es decir, como paso final actualizamos los pesos y los sesgos de acuerdo con las ecuaciones (4.13) y (4.14).

En resumen, para cada ejemplo de entrenamiento, el algoritmo de propagación hacia atrás primero hace una predicción (paso hacia adelante), mide el error y luego recorre cada capa en sentido inverso para medir la contribución de cada conexión al error (paso inverso) y finalmente ajusta ligeramente los pesos de conexión para reducir el error (paso de descenso de gradiente).

4.5. Funciones de activación

Las funciones de activación son clave en el funcionamiento de las RNA, sin estas la salida se comportaría como una función lineal que aunque es simple de resolver tiene una complejidad limitada lo que ocasiona que las RNA no sean capaces de reconocer relaciones no triviales dentro de los datos de entrada. Por lo tanto, es necesario aplicar una función de activación no lineal para otorgarle a la red la capacidad de hallar relaciones complejas entre los datos y así lograr predecir comportamientos no lineales entre la entrada y la salida. Esta, en resumen, es la razón por la que las funciones de activación son utilizadas en las RNA, podemos decir que las funciones de activación ayudan a aprender y dar sentido a los valores que tienen comportamientos no lineales entre las entradas y las salidas correspondientes.

Una característica importante de una función de activación es que debe ser diferenciable para que podamos implementar una estrategia de optimización de propagación hacia atrás.

Es muy común utilizar la misma función de activación en todas las capas ocultas, sin embargo, es posible utilizar una función de activación distinta en cada capa.

Existen diferentes funciones de activación, en la figura 4.6 se dan algunos ejemplos

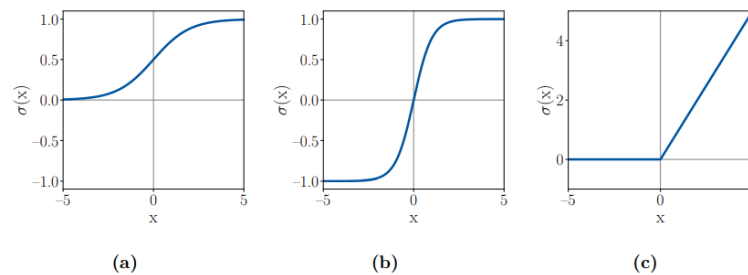


Figura 4.6: (a) Función Sigmoide. (b) Función tangente hiperbólica. (c) Función Relu o Rectified Linear Unit.

Una de las funciones de activación más usadas debido a su simplicidad es la función ReLU (Rectified Linear Unit), es una función lineal para valores mayores a cero

con lo cual logra mantener muchas de las propiedades que facilitan la optimización de los modelos lineales, sin embargo, a los valores menores o iguales a cero les asigna un cero como salida por lo que es una función no lineal que permite aprender relaciones complejas en los datos. Dado que esta es una función lineal para la mitad del dominio de entrada y no lineal para la otra mitad se le denomina función lineal por partes o función bisagra.

En este trabajo utilizamos, como función de activación, a la función ReLU, como ya lo mencionamos se compone por dos partes lineales: como resultado devuelve el mismo valor de la entrada si este es positivo o cero en otro caso. La describimos a continuación:

```
if input > 0
    return input
else
    return 0
```

O podemos describirla matemáticamente como una función g utilizando la función $\max()$ sobre el conjunto formado por el 0.0 y la entrada z , con $z \in \mathbb{R}$

$$g(z) = \max\{0, z\} \quad (4.15)$$

Esta última forma es quizás la manera más fácil de implementarla en Python.

4.6. Función de pérdida

Una duda que surge de manera natural al momento de entrenar una RNA y utilizarla para hacer predicciones es qué tan buenos (o malos) son los resultados obtenidos, de esta manera nos vemos en la necesidad de, alguna forma, medir la fiabilidad de predicción. El principal objetivo es entrenar una RNA, como ya lo hemos visto, para hacerlo modificamos los pesos y sesgos con el fin de mejorar la precisión de los resultados obtenidos. Para lograr esto es necesario calcular el error de la predicción con respecto de los valores deseados. La función de pérdida, también conocida como función de costo es la encargada de realizar esta tarea.

Una función de pérdida debe cumplir con dos condiciones:

- Que se pueda escribir como un promedio de las funciones de costo para ejemplos de entrenamiento individuales x , $C_x : C = \frac{1}{n} \sum_x C_x$

La razón de esta condición es que el algoritmo de propagación hacia atrás calcula las derivadas parciales $\frac{\partial C_x}{\partial w}$ y $\frac{\partial C_x}{\partial b}$ para ejemplos de entrenamiento individuales para posteriormente obtener $\frac{\partial C}{\partial w}$ y $\frac{\partial C}{\partial b}$ de todo el conjunto de datos promediando sobre todos los ejemplos de entrenamiento.

- La segunda condición es que sea una función que depende de las salidas de la red neuronal.

Existen diferentes funciones que son utilizadas como función de pérdida, en este trabajo nos enfocamos en la raíz cuadrada del error cuadrático medio (*RMSE*), que mide el error promedio en las predicciones de nuestra red. La fórmula matemática para calcularlo es

$$RMSE(X, h) = \sqrt{\frac{1}{m} \sum_{i=1}^m (y^i - h(x^i))^2} \quad (4.16)$$

Donde:

m es el número de ejemplos de entrenamiento.

x^i es el vector que representa los valores del i-ésimo ejemplo de entrenamiento y

y^i es la etiqueta o valor deseado para tal ejemplo.

X es una matriz que contiene los valores de los ejemplos de entrenamiento. Hay una fila por ejemplo que es igual a la transpuesta de x^i , $(x^i)^T$.

h es la función de predicción, también llamada hipótesis.

Por último, hacemos uso de la notación $RMSE(X, h)$ para referirnos a la función de costo utilizando el conjunto de datos **X** bajo la hipótesis h .

Capítulo 5

Análisis y resultados

En el siguiente capítulo introduciremos y explicaremos el algoritmo basado en técnicas de aprendizaje automático que desarrollamos y utilizamos para predecir la profundidad del máximo desarrollo de partículas X_{max} . Como ya se explicó anteriormente, usamos un conjunto de datos extraídos de simulaciones generadas en el software CONEX que genera archivos en formato .root, posteriormente procesamos este tipo de formato utilizando la librería Uproot de Python. Además de esto, utilizamos los datos generados en las simulaciones con el modelo QGSJETII-04 como datos de entrada para su reconstrucción en el software Offline y estas nuevas simulaciones sirvieron para entrenar otro modelo de red neuronal artificial. Nuestras redes neuronales artificiales fueron programadas en el lenguaje Python con el fin de aprovechar sus ventajas respecto a otros lenguajes de programación tanto en la ciencia de datos como en técnicas de aprendizaje automático.

5.1. ROOT

ROOT es un framework basado en $C++$ orientado a objetos que fue creado por la comunidad de física de altas energías (HEP: High Energy Physics) diseñado para almacenar y analizar grandes cantidades de datos de una manera eficiente.

La estructura de ROOT se divide en lo que se conoce como **TTree** que son como ramas que dividen un archivo y es donde se almacena la información. ROOT cuenta con varias herramientas para el análisis además de que cuenta con su propio paquete para implementar algoritmos de aprendizaje automático, el cual recibe el nombre de TMVA. Como ventajas extras tenemos que este framework nos permite crear gráficos, manipularlos y exportarlos a diferentes tipos de formatos.

En este trabajo utilizamos ROOT para ser capaces de acceder a los datos simulados que se explicaron en el capítulo 3 ya que el software CONEX genera archivos .root. Hacemos uso de una herramienta llamada el buscador de ROOT que, como su nombre lo indica, nos permite buscar archivos .root dentro de nuestro equipo y acceder al contenido de dicho archivo, se puede acceder a esta mediante el comando *TBrowser tb*. Con esta herramienta logramos visualizar todos los datos generados por las simulaciones accediendo al TTree y a distintas ramas del archivo, contamos

con un visualizador en el cual podemos ver los histogramas de las distintas características generadas y almacenadas y podemos trabajar directamente desde esta interfaz para analizar estos datos, para más información consulte [30].

5.1.1. Uproot

A pesar de que ROOT cuenta con una vasta cantidad de herramientas para el procesamiento y análisis de grandes cantidades de datos y su propio paquete de aprendizaje automático TMVA debemos decir que realizar lo anterior en ROOT resulta difícil por la amplia sintaxis de distintos lenguajes de programación de la que el usuario debe tener conocimiento, así como por el tiempo de procesamiento. En general, podemos obtener resultados profesionales y lograr realizar excelentes análisis y modelos de inteligencia artificial pero los recursos computacionales y tiempo de procesamiento son mayores comparados con otros lenguajes de programación como lo es Python, el cual es actualmente uno de los lenguajes de programación más utilizados para el análisis de datos, así como para el desarrollo de algoritmos de aprendizaje automático. De esta manera hemos decidido extraer los datos de importancia en nuestro trabajo para que puedan ser leídos y manipulados en Python.

Afortunadamente contamos con una librería de Python que nos permite leer y escribir archivos en formato .root, aquí quiero mencionar que contamos con dos herramientas que nos permiten hacer esto, una es **PyROOT** y la otra, que hemos decidido utilizar, **Uproot**. La diferencia radica en que la primera es solamente una interfaz en Python que ejecuta ROOT, mientras que Uproot puede ser utilizada sin requerir o ejecutar ROOT en absoluto lo cual se traduce en una mayor velocidad de procesamiento al momento de abrir o escribir archivos .root con respecto a PyROOT.

Mostramos a continuación cómo cargar un archivo .root haciendo uso de Uproot y una forma de acceder al TTree denominado *Shower* que es donde se almacenan los datos que son de nuestro interés.

```
import uproot

file = uproot.open("directorio/archivo.root:Shower")
file.show()
```

Listing 5.1: Importar Uproot y abrir un archivo .root

Gracias a este paquete podemos trabajar completamente en Python únicamente importándolo para la manipulación de los archivos .root; es necesario conocer la forma en la que está estructurada la información en el tipo de archivos que estamos tratando. Nos encontramos con funciones muy fáciles de implementar para navegar a través de los TTree y ramas que conforman el archivo .root, además podemos realizar histogramas, gráficas, análisis y ajustes con esta paquetería, sin embargo encontramos útil importar las muy bien conocidas librerías numpy y pandas, así podemos extraer las características que son de nuestro interés para convertirlos en dataframes

```
variable = pd.DataFrame(file["variable"].array(library="pd"))
```

Listing 5.2: Extracción y conversión de las características a un DataFrame

y posteriormente unir las en un archivo de formato .csv

```
bin = var1.assign(var1 = var1, var2 = var2, ...)

bin.columns = ['var1', 'var2', ...]
print(bin)

bin.to_csv('nombre.csv', sep='\t')
```

Listing 5.3: Unir las características en un archivo .csv

una vez hecho esto podemos hacer uso de todas las herramientas de análisis de datos y aprendizaje automático que nos ofrece Python.

5.2. Estrategia de entrenamiento

La red neuronal artificial fue programada usando una computadora Mac mini con memoria de 15.5 GiB, procesador Intel Core i7-4578U CPU 3.00 GHz x 4, disco duro de 1.1 terabytes. Para obtener el valor de la profundidad del máximo desarrollo de partículas de las lluvias X_{max} utilizando redes neuronales artificiales hacemos uso de una arquitectura simple de una red neuronal: creamos una red neuronal secuencial con capas totalmente conectadas utilizando las librerías *TensorFlow*, *scikit-learn* y el framework *Keras*.

Durante el entrenamiento de la red la predicción (salida) se compara con el valor deseado o valor real que nos proporcionan los datos de entrenamiento.

Hemos realizado pasos previo al entrenamiento de la red con la finalidad de lograr la generalización por parte de nuestra red neuronal de los 7×10^6 lluvias generadas por primario y por modelo hadrónico. Antes que nada es necesario una preparación cuidadosa de los datos.

5.2.1. Selección de los datos a utilizar

Cuando nos enfrentamos a conjuntos de datos muy grandes se vuelve necesario identificar los parámetros o características que son de mayor utilidad para llevar a cabo el estudio deseado y sobre todo identificar los datos más relevantes para poder entrenar una red neuronal de manera óptima.

Como primer paso accedemos al, así llamado, TTree en nuestro archivo .root y al segundo TTree *Shower* en donde se encuentran los parámetros físicos más destacados de las simulaciones de una cascada de partículas. Haciendo uso de Uproot y pandas podemos extraer estos datos y manejarlos en forma de dataframe para su posterior conversión al formato .csv, el cual nos facilita mucho a la hora de trabajar con una gran cantidad de datos (como es el caso) y visualizar la estructura de estos de una forma más sencilla. Otro aspecto que destacar es que, una vez hecho esto el manejo de datos y entrenamiento de la red neuronal ahora es completamente en Python, lo cual nos brinda aún más flexibilidad durante todo el proceso.

Podemos eliminar las características que no son relevantes como lo es la semilla que utilizó el programa de simulación o el tiempo de procesamiento, entre otros, ya que estos no representan parámetros reales en una cascada real de rayos cósmicos.

name	typename	interpretation
lgE	float	AsDtype('>f4')
zenith	float	AsDtype('>f4')
azimuth	float	AsDtype('>f4')
Seed2	int32_t	AsDtype('>i4')
Seed3	int32_t	AsDtype('>i4')
Xfirst	float	AsDtype('>f4')
Hfirst	float	AsDtype('>f4')
XfirstIn	float	AsDtype('>f4')
altitude	double	AsDtype('>f8')
X0	float	AsDtype('>f4')
Xmax	float	AsDtype('>f4')
Nmax	float	AsDtype('>f4')
p1	float	AsDtype('>f4')
p2	float	AsDtype('>f4')
p3	float	AsDtype('>f4')
chi2	float	AsDtype('>f4')
Xmx	float	AsDtype('>f4')
Nmx	float	AsDtype('>f4')
XmxdEdX	float	AsDtype('>f4')
dEdXmx	float	AsDtype('>f4')
cpuTime	float	AsDtype('>f4')
nX	int32_t	AsDtype('>i4')
X	float[]	AsJagged(AsDtype('>f4'))
N	float[]	AsJagged(AsDtype('>f4'))
H	float[]	AsJagged(AsDtype('>f4'))
D	float[]	AsJagged(AsDtype('>f4'))
dEdX	float[]	AsJagged(AsDtype('>f4'))
Mu	float[]	AsJagged(AsDtype('>f4'))

Figura 5.1: Parámetros obtenidos en las simulaciones con CONEX.

Una manera de darnos una idea del tipo de datos con los que estamos trabajando es realizar un histograma de cada atributo numérico, también podemos realizar gráficos de alguna característica en especial contra el valor que deseamos predecir X_{max} . De esta forma es posible visualizar el tipo de datos con el que tratamos y de manera intuitiva observar el comportamiento de estos.

```
%matplotlib inline
import matplotlib.pyplot as plt
file.hist(bins=50, figsize=(20,15))
plt.show()
```

Listing 5.4: Visualización de los atributos.

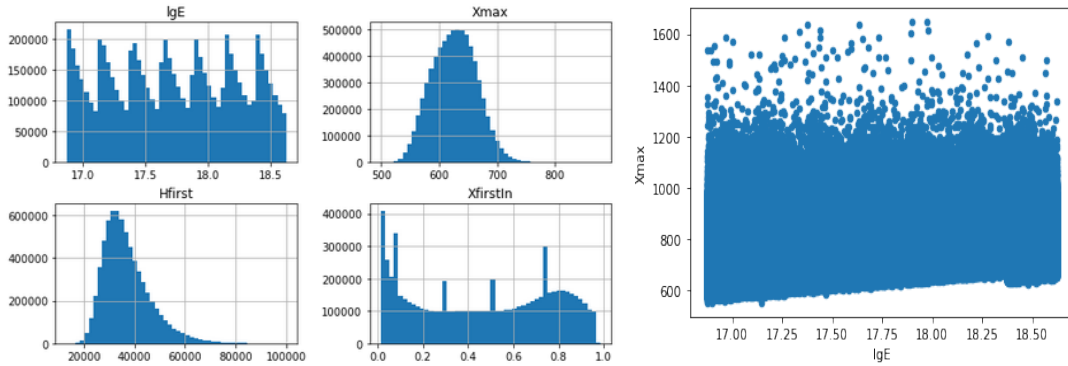


Figura 5.2: Ejemplos de algunos histogramas realizados y una gráfica de X_{max} vs lgE

La red neuronal artificial puede no dar resultados óptimos si en los datos de entre-

namiento se encuentran distintas escalas en las diferentes características, esto podría llevarnos a que la red no logre identificar relaciones en los datos y las predicciones se tornen muy malas, para evitar esto hemos normalizado el conjunto entero de datos.

```
x = (x - x.mean())/x.std()
x
```

Listing 5.5: Normalización de los datos.

5.2.2. Conjunto de validación, entrenamiento y de prueba

Hasta este punto no hemos explorado los datos con un nivel de profundidad que nos permita comprender las relaciones entre estos, seguramente podemos aprender mucho más de ellos si los analizamos con mayor detenimiento, sin embargo, si continuamos de esta forma nuestro cerebro podría observar algún patrón en el conjunto de datos e inconscientemente podríamos generar un modelo que tome en cuenta el patrón que hayamos observado, lo que nos conduciría a que nuestro modelo realice predicciones acertadas con los datos proporcionados pero no sea capaz de desempeñarse correctamente cuando se enfrente a un nuevo conjunto de datos.

Tomando lo anterior en consideración, procedemos a dividir el conjunto de datos en tres subconjuntos los cuales son:

- Conjunto de validación:

Este conjunto se utiliza para evaluar la precisión de las predicciones hechas por la RNA al mismo tiempo que es entrenada, de esta forma podemos ver que tan bien o que tan mal nuestro modelo logra generalizar y cómo van mejorando, empeorando o si no existe una variación notable en las predicciones, esto nos ayuda a escoger mejor los hiperparámetros de la red neuronal artificial. La razón de usar el conjunto de validación para esta tarea y no el conjunto de prueba es que si usamos este último lo que haremos será ajustar nuestro modelo para obtener mejores resultados con el conjunto de prueba y terminaremos con una red neuronal sobre ajustada que no podrá realizar buenas predicciones en nuevos conjuntos de datos.

- Conjunto de prueba:

Es utilizado para medir la precisión de las predicciones de la red, este conjunto solamente es utilizado hasta una vez finalizado el entrenamiento para mostrar que tan bien la red logra realizar predicciones con nuevos conjuntos de datos.

- Conjunto de entrenamiento

Como su nombre lo indica, el conjunto de entrenamiento es utilizado durante el entrenamiento para que la red ajuste sus parámetros (pesos y sesgos) con el fin de encontrar relaciones entre los datos.

Para dividir el conjunto tomamos los valores de los datos de forma aleatoria con el fin de que cada uno de los tres subconjuntos contenga muestras representativas del total de los datos. Para evitar que cada vez que ejecutemos el algoritmo se genere

un conjunto de entrenamiento diferente y que con el tiempo nuestra red termine visualizando el conjunto total (que es lo que deseamos evitar) fijamos la semilla para asegurarnos que los datos siempre serán los mismos en los tres subconjuntos.

```
from sklearn.model_selection import train_test_split

(X_train_full, X_test, y_train_full,
 y_test = train_test_split(x, y, test_size=0.2,
                           random_state=42))
(X_train, X_valid, y_train,
 y_valid = train_test_split(X_train_full, y_train_full,
                           random_state=42))
```

Listing 5.6: Código para dividir el conjunto en tres subconjuntos.

5.2.3. Correlación con X_{max}

La correlación es la relación que existe entre dos variables y cómo varía una con respecto de la otra en caso de que exista un cambio en alguna de ellas.

Para estudiar la relación lineal existente entre dos variables podemos utilizar la covarianza que mide el grado de variación de dos variables aleatorias:

$$\text{covarianza muestral} = \text{Cov}(X, Y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{N - 1} \quad (5.1)$$

En donde $\bar{x}$ y $\bar{y}$ son las medias de cada variable y x_i y y_i son el i -ésimo valor de las variables x e y respectivamente y N es el número total de variables.

Un valor positivo indica que las dos variables cambian en la misma dirección y un valor negativo indica que cambian en dirección opuesta.

El coeficiente de correlación estándar (también llamado coeficiente de Pearson) es la covarianza estandarizada:

$$\rho = \frac{\text{Cov}(X, Y)}{\sigma_x \sigma_y} \quad (5.2)$$

En la práctica, raramente tenemos acceso a toda la población, por lo que necesitamos estimar su valor a partir de una muestra mediante el coeficiente de Pearson muestral (r):

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}} \quad (5.3)$$

En este coeficiente tenemos valores entre -1 y 1. 1 significa una fuerte correlación lineal positiva perfecta y -1 una fuerte correlación lineal negativa perfecta (los valores entre las dos variables varían inversamente) mientras que cero o un valor cercano nos indica que no existe correlación o es demasiado baja. Utilizamos la función `corr()` de pandas para conocer la correlación existente entre cada atributo con la variable X_{max} y así determinar qué características son las que mejor resultados nos darán al entrenar una red neuronal.

```
corr_matrix = file.corr()
corr_matrix["Xmax"].sort_values(ascending=False)
```

Listing 5.7: Función corr() de Python.

5.2.4. Preparación de los datos

Algunos conjuntos de datos presentaban valores que se alejaban mucho de los valores esperados de la profundidad de desarrollo de una cascada de partículas

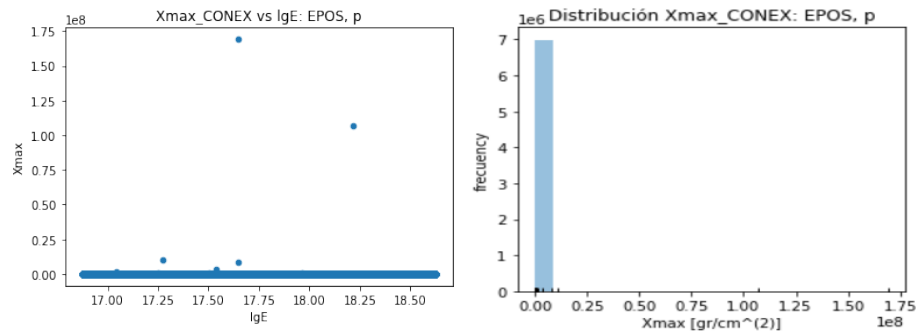


Figura 5.3: Ejemplo de la gráfica X_{max} contra $\lg E$ del modelo EPOS LHC utilizando como primario protón y la correspondiente distribución de X_{max} utilizando todos los datos de la simulación.

por lo que realizamos un corte de las profundidades atmosféricas que se alejan mucho del valor medio

```
file2 = file[file.Xmax < 1000]
```

Listing 5.8: Filtro de X_{max}

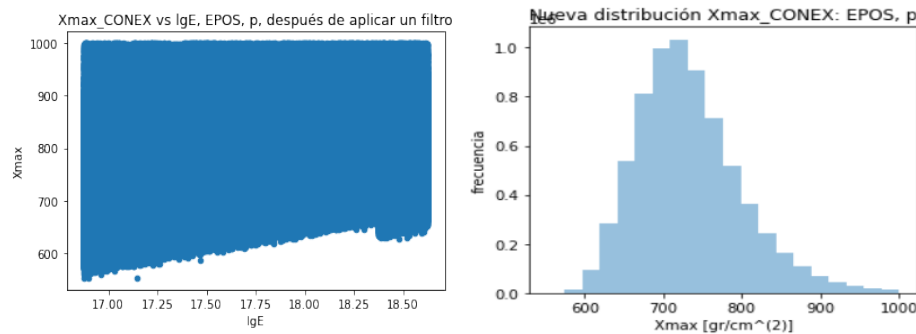


Figura 5.4: Ejemplo de la gráfica X_{max} contra $\lg E$ del modelo EPOS LHC utilizando como primario protón y la correspondiente distribución de X_{max} después de haber quitado algunos elementos.

Finalmente, los atributos seleccionados para el entrenamiento de la red neuronal son los siguientes:

- $\lg E$: el logaritmo en base 10 de la energía del primario en eV.
- X_{firts} : primer punto de interacción real en profundidad inclinada, medida en g/cm^2 .
- N_{max} : Número de partículas cargadas por encima del límite en el máximo de la cascada.
- N_{mx} : Máximo ajuste cuadrático del perfil.
- $dEdX_{mx}$: Ajuste cuadrático máximo en el perfil $dE/dX(X)$, medido en $GeV/(g/cm^2)$.

Existen otras variables que están fuertemente relacionadas con el valor de X_{max} , sin embargo, algunos algoritmos de aprendizaje automático pueden sufrir de sobreajuste (cuando el algoritmo hace predicciones muy buenas con los conjuntos ya probados pero muy malas con nuevos datos) si existen atributos altamente correlacionados por lo que decidimos no utilizarlos para entrenar la red. Por otro lado, tenemos otras características muy importantes que nos pueden dar mucha información acerca de la lluvia y su desarrollo, variables que vale la pena analizar y manipular para el entrenamiento de nuestra red neuronal artificial, pero nos vimos rebasados en la capacidad de nuestro equipo informático ya que requieren mucha memoria y recursos computacionales para ser procesados por lo que no las incluimos en este trabajo.

5.3. Arquitectura de la red neuronal artificial desarrollada

Nuestro modelo de una red neuronal artificial fue programado utilizando el lenguaje de programación Python, muy popular en el campo del aprendizaje automático, utilizando las librerías TensorFlow y Scikit-Learn en el framework Keras, estas herramientas han sido diseñadas para la creación de algoritmos de aprendizaje automático, gracias a esto mantenemos el entorno de desarrollo simple y robusto.

Utilizamos el error cuadrático medio como función de pérdida para medir el error en las predicciones de nuestro modelo el cual sufrió múltiples cambios a lo largo del desarrollo con el fin de determinar qué configuración de capas ocultas y número de perceptrones nos proporcionaba mejores resultados evitando el sobreajuste o un desajuste ya que no existe un procedimiento que nos guíe para obtener los mejores valores en estos parámetros.

Del total de los datos el 60 % forma parte del conjunto de entrenamiento, 20 % se destinó para el conjunto de validación y el restante 20 % para el conjunto de prueba.

En el primer modelo la capa de entrada consta de cinco perceptrones cada una de las cuales corresponde a las entradas de cada característica, seguido por las tres capas ocultas totalmente conectadas con 128, 512 y 256 perceptrones respectivamente. Finalmente, la capa de salida consta únicamente de un solo perceptrón ya que solamente deseamos obtener la predicción de la profundidad atmosférica del máximo desarrollo de partículas secundarias.

```

import tensorflow as tf
import keras
from keras.models import Sequential, Model
from keras.layers import *
from keras.optimizers import Adam, SGD
from tensorflow.keras.optimizers import Adam, SGD
from sklearn.datasets import make_regression

from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense

model = keras.models.Sequential([
    (keras.layers.Dense(128, activation="relu",
                        input_shape=X_train.shape[1:])),
    keras.layers.Dense(512, activation="relu"),
    keras.layers.Dense(256, activation="relu"),
    keras.layers.Dense(1)
])
(model.compile(loss="mean_squared_error",
               optimizer=keras.optimizers.Adam(learning_rate=1e-3)))
(history = model.fit(X_train, y_train, epochs=100,
                    validation_data=(X_valid, y_valid)))

```

Listing 5.9: Código del primer modelo desarrollado.

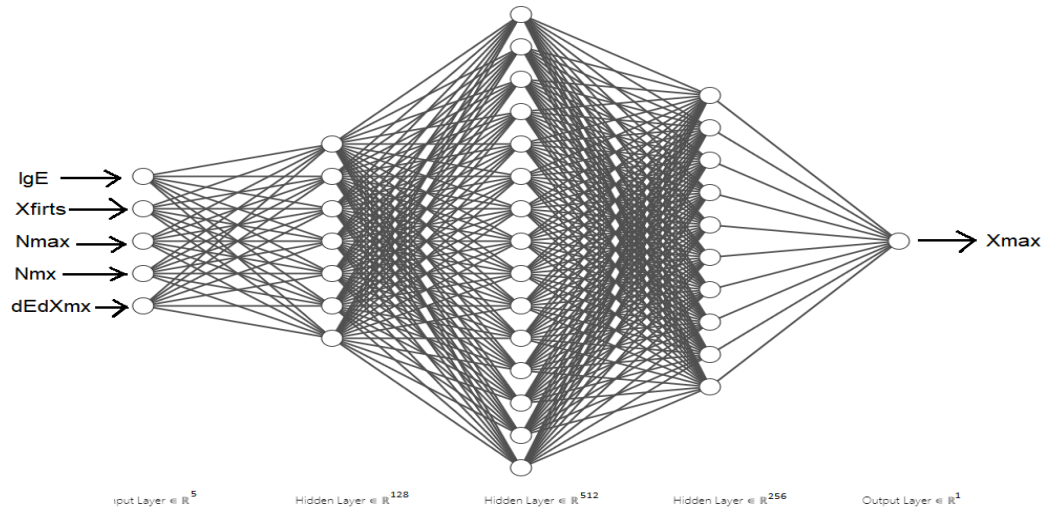


Figura 5.5: Representación de nuestro modelo de red neuronal, incluyendo la capa de entrada, salida, capas ocultas y sus perceptrones correspondientes.

El segundo modelo, en el cual utilizamos las salidas del software *Offline*, está compuesto por ocho perceptrones en la capa de entrada entre las cuales tenemos a X_{max} y lgE correspondientes a la profundidad atmosférica del máximo desarrollo de partículas y la energía en logaritmo en base 10 de la energía obtenidos en las simulaciones con el software CONEX y a $logE$ de la energía obtenida en las simulaciones de Real Monte-Carlo, además de valores que corresponden a las señales que otorgan los detectores del Observatorio Pierre Auger en las simulaciones, tenemos un solo

perceptrón en la capa de salida y tres capas ocultas totalmente conectadas con 256, 512 y 256 perceptrones respectivamente. El conjunto de entrenamiento, validación y prueba están divididos de la misma manera que en el modelo anterior.

```
import tensorflow as tf
import keras
from keras.models import Sequential, Model
from keras.layers import *
from keras.optimizers import Adam, SGD
from tensorflow.keras.optimizers import Adam, SGD
from sklearn.datasets import make_regression

def regression_model():
    model = Sequential()
    (model.add(Dense(256, activation='relu',
                    input_shape=X_train.shape[1:])))
    model.add(Dense(512, activation='relu'))
    model.add(Dense(256, activation='relu'))
    model.add(Dense(1))

    (model.compile(optimizer
                   =keras.optimizers.Adam(learning_rate=0.0001),
                   loss='mean_squared_error'))

    return model

# build the model
model = regression_model()

# fit the model
(history
 = model.fit(X_train, y_train, batch_size= 32, epochs=100,
             validation_data=(X_valid, y_valid), verbose=1))
```

Listing 5.10: Código del segundo modelo desarrollado.

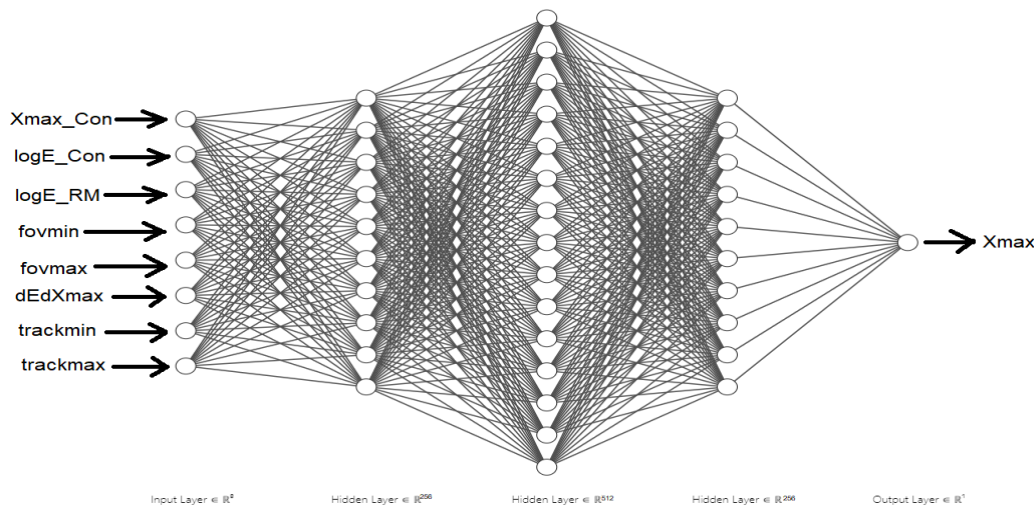


Figura 5.6: Representación del segundo modelo de red neuronal artificial, incluyendo la capa de entrada, salida, capas ocultas y sus perceptrones correspondientes.

En todos nuestros modelos utilizamos la función de activación ReLU entre los

perceptrones de cada capa de nuestra red neuronal artificial.

5.4. Resultados

El primer modelo de red neuronal artificial ha sido entrenado tanto para núcleos de hierro y protón utilizando tres modelos hadrónicos distintos: EPOS LHC, QGSJETII-04 y SIBYLL, es decir, concluimos con seis modelos entrenados. La red ha sido entrenada para predecir la profundidad del máximo desarrollo de partículas X_{max} de cada modelo con cada uno de los primarios que utilizamos para este trabajo.

Dividimos los datos de la siguiente manera:

Total de datos utilizados en cada subconjunto			
Modelo	Conjunto de entrenamiento	Conjunto de validación	Conjunto de prueba
EPOS LHC:			
Fe	4,018,023	1,339,342	1,339,342
p	4,181,211	1,393,737	1,393,737
QGSJETII-04			
Fe	4,159,677	1,386,559	1,386,560
p	4,181,211	1,393,737	1,393,737
SIBYLL			
Fe	4,246,527	1,415,509	1,415,509
p	4,177,476	1,392,492	1,392,493

Tabla 5.1: Número de datos en cada subconjunto utilizado para el entrenamiento y prueba de rendimiento de la red neuronal artificial en cada modelo y con cada primario utilizado.

La razón de que algunos modelos tengan más datos que otros es que algunos datos fueron eliminados dependiendo de que tanto se alejaran de los valores esperados, esto varía con cada modelo y primario.

Para evitar que tengamos que entrenar la red cada vez que deseamos hacer alguna predicción guardamos los modelos entrenados para ocuparlos cuando sea necesario

```
model.save('modelo.h5')
```

Listing 5.11: Guardar los modelos entrenados.

Y lo cargamos para hacer predicciones con el conjunto de prueba

```
from keras.models import load_model
modelo = load_model('modelo.h5')
valores_predichos = model.predict(X_test)
valores_predichos
```

Listing 5.12: Cargar modelo y hacer predicciones.

Posteriormente hacemos uso de estas predicciones para compararlas con los valores que obtuvimos en las simulaciones, se muestran a continuación los resultados.

Conocer la energía del primario es de mucha importancia en el estudio de los rayos cósmicos, por ello graficamos el valor de X_{max} contra el logaritmo de la energía del primario (ya descrito) utilizando el conjunto de prueba; graficamos los datos de la simulación y los comparamos con los datos que nuestra red neuronal artificial predice en una gráfica superpuesta para una visualización directa, además de realizar un ajuste lineal para los valores provenientes de las simulaciones y aquellos predichos por la RNA.

```
%matplotlib inline
import pylab as pl
from scipy import interpolate
from scipy.interpolate import interp1d

lgE = X_test['lgE']

pl.plot(lgE,y_test,'bo',label="Xmax_CONEX")
pl.plot(lgE,valores_predichos,'g+',color='orange',label="Xmax_ANN")
pl.title('Cotejo de Xmax_CONEX vs Xmax_ANN')
```

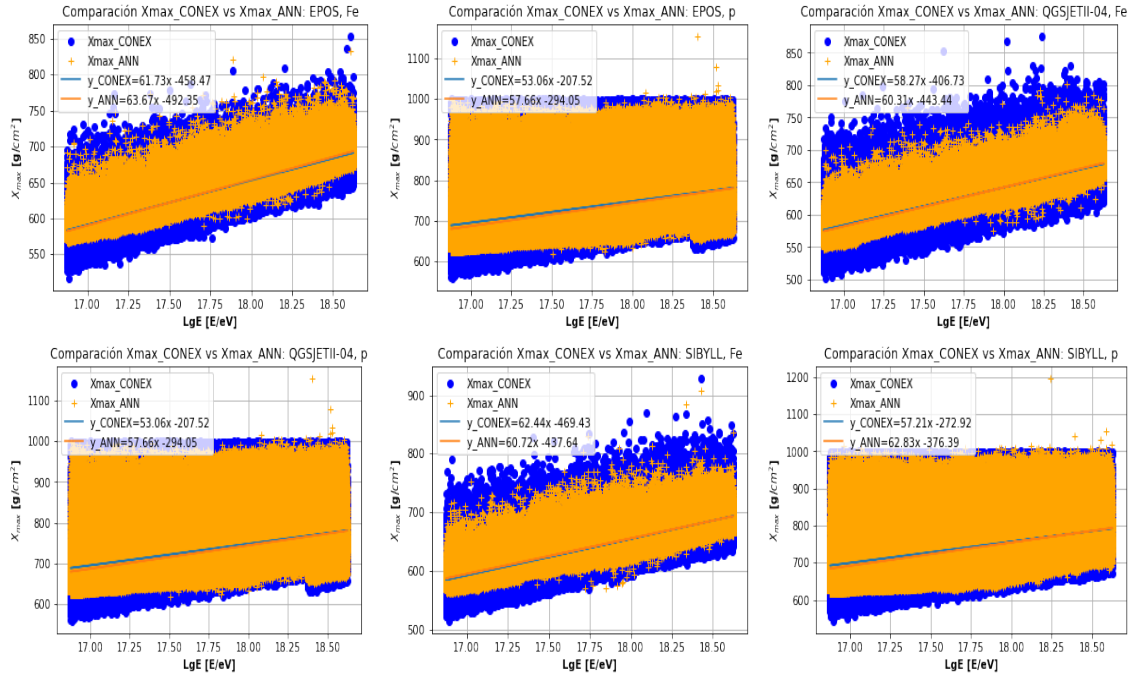


Figura 5.7: X_{max} contra lgE obtenidas para protón y núcleos de hierro en cada uno de los modelos hadrónicos utilizados.

En la siguiente tabla mostramos los ajustes lineales realizados para cada primario en cada uno de los modelos hadrónicos utilizados, mostramos además la diferencia entre cada ajuste lineal:

AJUSTES LINEALES			
Modelo	Ajuste simulaciones	Ajuste red neuronal	Variación
EPOS LHC:			
Fe	$y = 61.73x - 458.47$	$y = 63.67x - 492.35$	$\Delta y = 1.94x - 33.88$
p	$y = 53.06x - 207.52$	$y = 57.66x - 294.05$	$\Delta y = 4.6x - 86.53$
QGSJETII-04			
Fe	$y = 58.27x - 406.73$	$y = 60.31x - 443.44$	$\Delta y = 2.04x - 36.71$
p	$y = 53.06x - 207.52$	$y = 57.66x - 294.05$	$\Delta y = 4.6x - 86.53$
SIBYLL			
Fe	$y = 62.44x - 469.43$	$y = 60.72x - 437.64$	$\Delta y = 1.72x - 31.79$
p	$y = 57.21x - 272.92$	$y = 62.83x - 376.39$	$\Delta y = 5.62x - 103.47$

Tabla 5.2: Comparación de ajustes lineales de los valores de las simulaciones y los predichos por la red neuronal artificial en cada modelo y primario utilizado.

Posteriormente elaboramos un histograma del valor X_{max} utilizando el conjunto de prueba y otro con los valores predichos por nuestra red neuronal con el fin de comparar la distribución de este valor en ambos casos. Mostramos la media y desviación estándar en cada histograma.

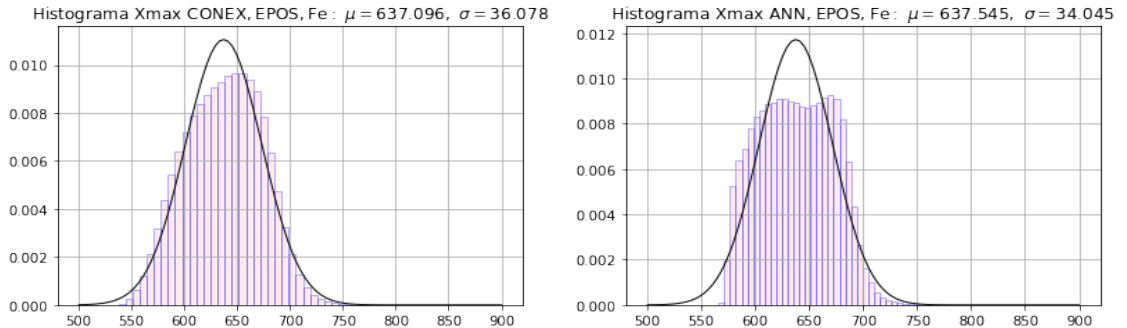
```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))

axs[0].hist(x=y_test, bins=20, color="#3182bd", alpha=0.5)
(axs[0].plot(y_test, np.full_like(y_test, -0.01),
             'k', markeredgewidth=0))
axs[0].set_title('Histograma de Xmax_CONEX')
axs[0].set_xlabel('Xmax [gr/cm(2)]')
axs[0].set_ylabel('frecuency')

axs[1].hist(x=y_prue, bins=20, color="#3182bd", alpha=0.5)
(axs[1].plot(prediccion, np.full_like(y_prue, -0.01),
             'k', markeredgewidth=0))
axs[1].set_title('Histograma de Xmax_ANN')
axs[1].set_xlabel('Xmax [gr/cm(2)]')
axs[1].set_ylabel('frecuency')

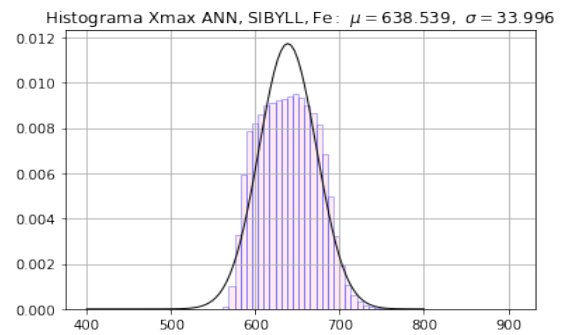
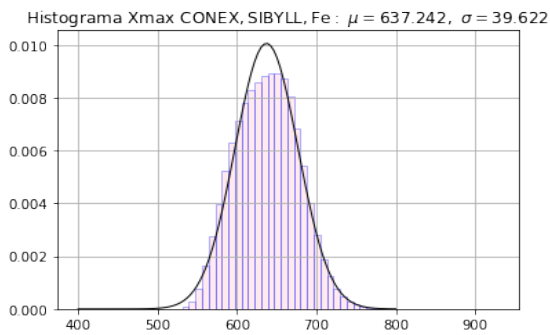
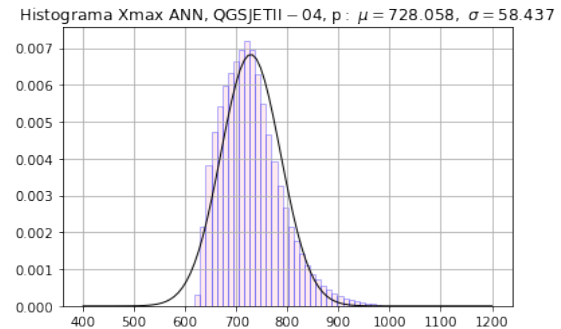
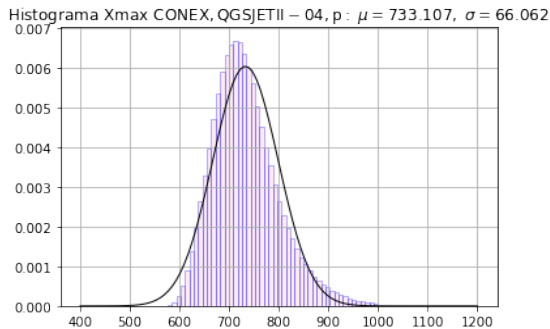
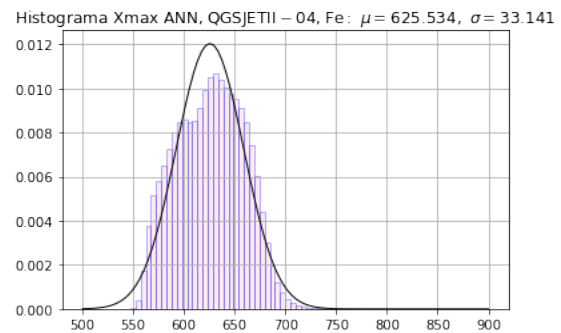
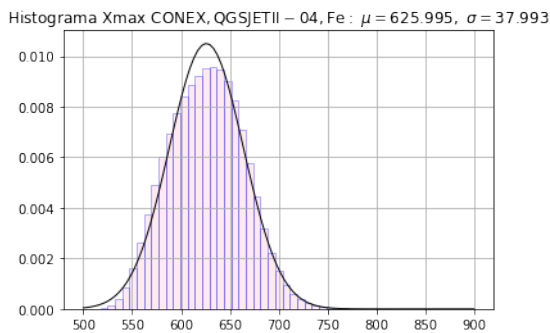
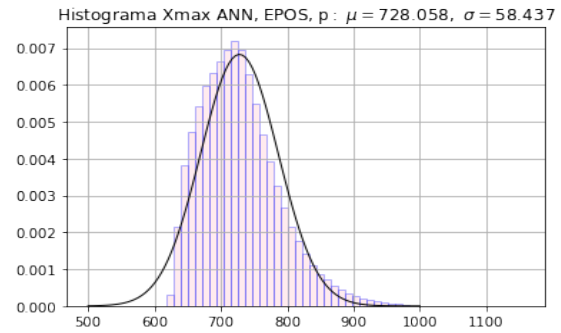
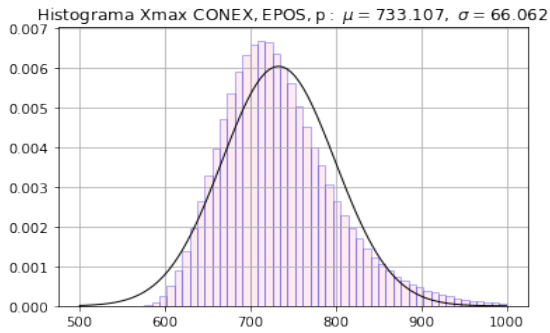
plt.tight_layout();
```

Listing 5.13: Realizar los histogramas de X_{max} contra lgE



Análisis y resultados

5.4 Resultados



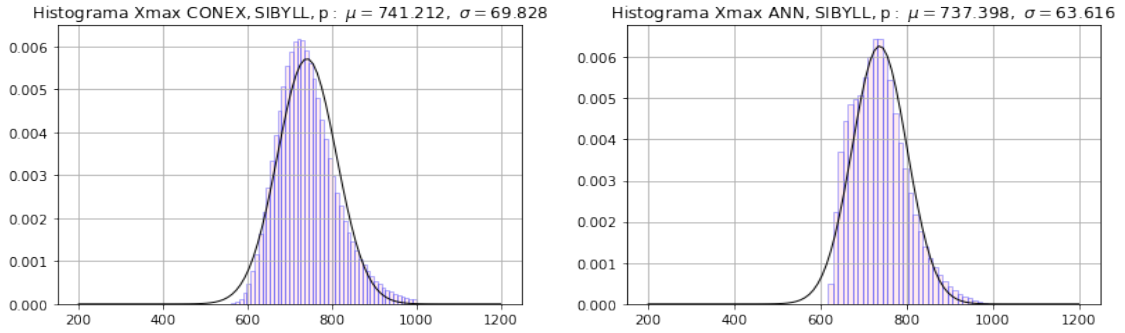


Figura 5.8: Comparación de la distribución de X_{max} del conjunto de prueba con los valores predichos por la red.

En la siguiente tabla comparamos los valores promedio correspondientes al valor X_{max} obtenidos en las simulaciones y el predicho por la red neuronal artificial.

DISTRIBUCIÓN DE X_{max}					
Modelo	μ simulaciones	σ simulaciones	μ ANN	σ ANN	$\Delta\mu$
EPOS LHC:					
Fe	637.096 g/cm ²	36.078	637.545 g/cm ²	34.045	0.449 g/cm ²
p	733.107 g/cm ²	66.062	728.058 g/cm ²	58.437	5.049 g/cm ²
QGSJETII-04					
Fe	625.995 g/cm ²	37.993	625.534 g/cm ²	33.141	0.461 g/cm ²
p	733.107 g/cm ²	66.062	728.058 g/cm ²	58.437	5.049 g/cm ²
SIBYLL					
Fe	637.242 g/cm ²	39.622	638.539 g/cm ²	33.996	1.297 g/cm ²
p	741.212 g/cm ²	69.828	737.398 g/cm ²	63.616	3.814 g/cm ²

Tabla 5.3: Media y desviación estándar de las distribuciones obtenidas.

Finalmente medimos la correlación existente entre el valor de X_{max} del conjunto de prueba con el predicho por la red, esta cantidad nos proporciona información de que tan bien o mal nuestro modelo logra predecir el valor que deseamos.

```
fig, axes = plt.subplots()
axes.set_title('Correlacion Xmax_ANN contra Xmax_CONEX')
axes.scatter(Ydnn, Ytrue, edgecolors=(0, 0, 0))
axes.set_xlabel(r'$X_{max,\backslash;ANN}$ $[g/cm^2]$',)
axes.set_ylabel(r'$X_{max,\backslash;CONEX}$ $[g/cm^2]$',)
left, bottom, width, height = [0.23, 0.1, 0.3, 0.4]
axes.text(0.98,0.25, \
         r"$\mu$=%.3f" % np.mean(reco) + \
         r"$[g/cm^2]$" + \
         "\n" + "$\sigma$=%.2f" % np.std(reco) \
         + r"$[g/cm^2]$" + \
         "\n" + "corr = %.3f " \
         % np.corrcoef(Ydnn.T,Ytrue.T)[0,1] \
         + r"",
         verticalalignment = 'top',
         horizontalalignment = \
         'right', transform=axes.transAxes,
```

```
backgroundcolor='w')
(axes.plot([Ytrue.min(), Ytrue.max()],
           [Ytrue.min(), Ytrue.max()], "r--", lw=4))
```

Listing 5.14: Correlaciones.

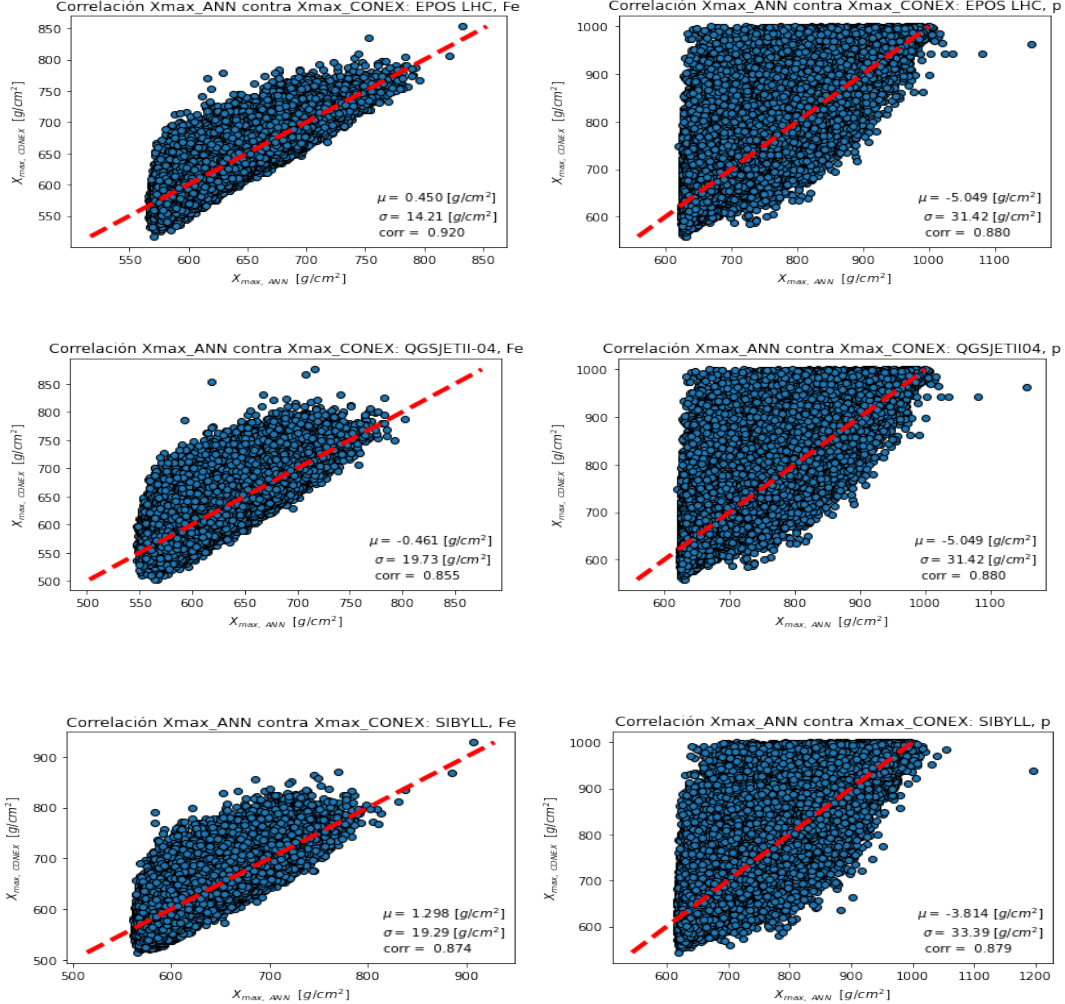


Figura 5.9: Se muestran las correlaciones halladas entre los valores de la simulación contra los predichos para los primarios y modelos hadrónicos empleados en este trabajo.

Se ha encontrado una correlación por encima de 0.85 para la mayoría de los modelos hadrónicos, siendo el protón el que ha requerido una mayor limpieza de datos y en el cual la red ha tenido mayor dificultad para aprender el comportamiento de los mismos.

Resultados de la red alimentada con simulaciones de Offline

Para la reconstrucción en el software Offline utilizamos solamente las simulaciones realizadas en CONEX con el modelo hadrónico QGSJETII-04 tanto para protón y núcleos de hierro. Nuestro propósito nuevamente es lograr predecir la profundidad del máximo desarrollo de partículas X_{max} , para el entrenamiento utilizamos los datos

generados en las simulaciones de Real Monte-Carlo además de los valores de energía y X_{max} obtenidos en las simulaciones de CONEX.

El total de datos en cada subconjunto se muestra a continuación:

Total de datos utilizados en cada subconjunto			
Modelo	Conjunto de entrenamiento	Conjunto de validación	Conjunto de prueba
Fe	168,054	56,018	56,018
p	183,531	61,177	61,178

Tabla 5.4: Número de datos en cada subconjunto.

Mostramos los resultados obtenidos con esta red:

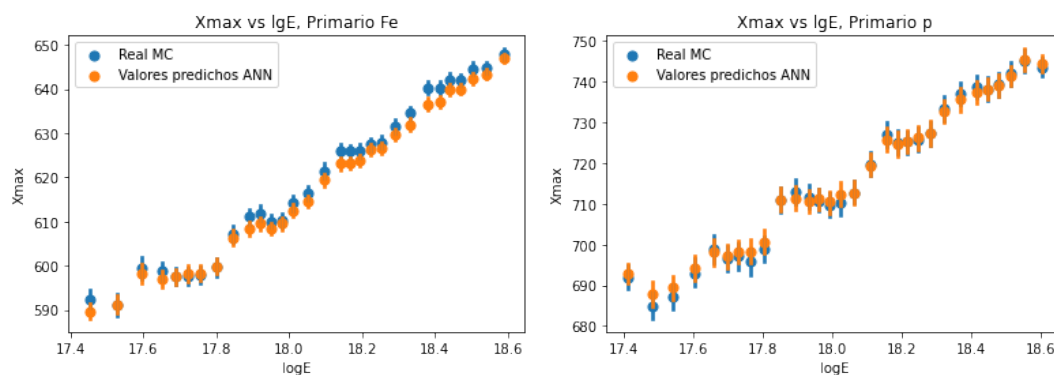
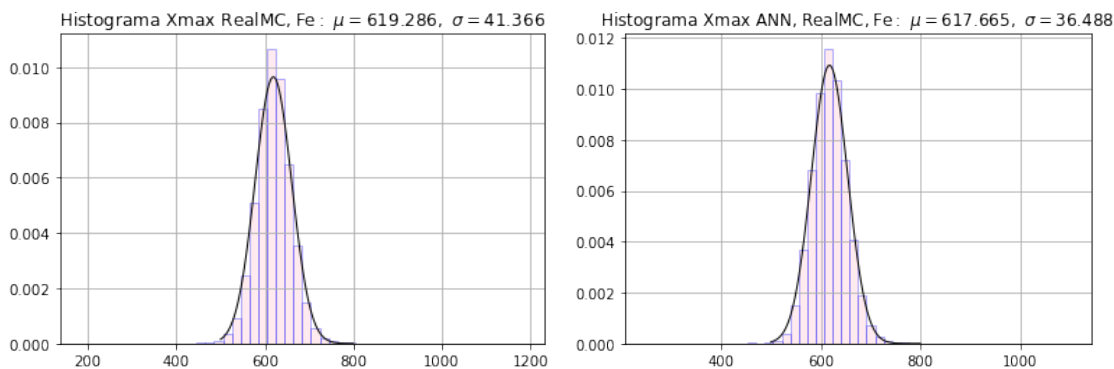


Figura 5.10: X_{max} contra $\log E$ para núcleos de hierro y protón en las simulaciones de Real Monte-Carlo, se puede apreciar que la red es capaz de reproducir los quiebres.

Las distribuciones del valor X_{max} se muestran a continuación:



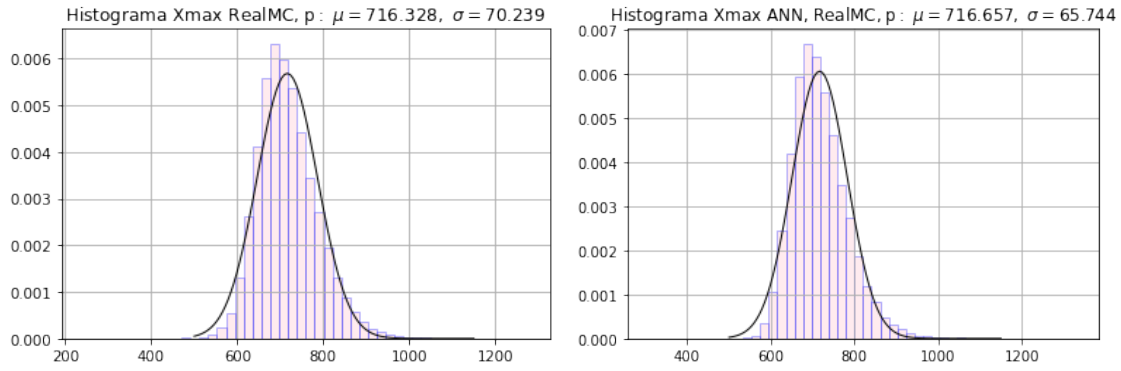


Figura 5.11: Comparación de la distribución de X_{max} del conjunto de prueba generado con Offline contra los valores predichos por la red para el modelo hadrónico QGSJETII-04 con núcleos de hierro y protón.

La siguiente tabla muestra la comparación del valor medio de X_{max} encontrado en las simulaciones y el predicho por la red neuronal artificial.

DISTRIBUCIÓN DE X_{max}					
Primario	μ simulaciones	σ simulaciones	μ ANN	σ ANN	$\Delta\mu$
Fe	619.286 g/cm ²	41.366	617.665 g/cm ²	36.488	1.621 g/cm ²
p	716.328 g/cm ²	70.239	716.657 g/cm ²	65.744	0.329 g/cm ²

Tabla 5.5: Media y desviación estándar de las distribuciones obtenidas en las simulaciones de Real Monte-Carlo.

Finalmente mostramos las correlaciones halladas:

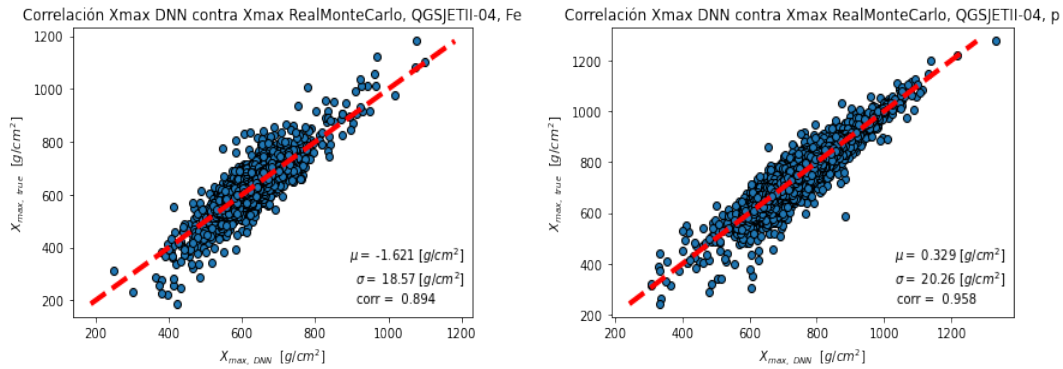


Figura 5.12: Se muestran las correlaciones halladas entre los valores de las simulaciones de Offline contra los predichos.

Conclusiones y trabajo a futuro

El uso de algoritmos de aprendizaje automático en el campo de la ciencia ha ido ganando popularidad en los últimos años, ofrece ventajas de predicción que con métodos tradicionales son difíciles de lograr, además de que una vez entrenada una red neuronal artificial el costo computacional para hacer nuevas predicciones es bajo con respecto a otros métodos computacionales. Las redes neuronales artificiales han mostrado ser una gran herramienta en la investigación de múltiples campos de la física, las matemáticas y la ciencia en general, motivo por el cual tratamos de hacer uso de estas herramientas para realizar predicciones y en trabajos posteriores ahondar aún más con el objetivo de aportar descubrimientos en áreas donde existan investigaciones abiertas.

Podemos concluir lo siguiente:

- Hemos mostrado que el uso de redes neuronales artificiales ayuda a encontrar patrones en las lluvias de partículas y es una herramienta que nos ayuda a hallar el valor de X_{max} en simulaciones de cascadas de Monte-Carlo.
- Podemos predecir el comportamiento de los datos fuera del conjunto con el que ha sido entrenado el algoritmo siempre que logremos hallar los hiperparámetros que logren hallar las relaciones presentes en los datos.
- A pesar de que el modelo que presentamos es simple y contiene pocos hiperparámetros éste puede adaptarse a una amplia variedad de primarios y modelos hadrónicos.

5.4.1. Trabajo a futuro

La importancia del trabajo realizado y la elaboración de una red neuronal más compleja utilizando nuevos métodos de aprendizaje automático es el primer paso para lograr predecir con mayor precisión no solo X_{max} , sino también otras variables para lograr una reconstrucción completa de las lluvias y descubrir relaciones entre las variables que hasta el momento pueden pasar desapercibidas.

Como siguiente paso deseamos probar la capacidad de predicción con datos reales registrados por observatorios en tierra como es el caso del Observatorio Pierre Auger. Buscamos que una vez entrenada y probada con datos reales, nuestra red neuronal use únicamente datos recolectados por los detectores de superficie o datos híbridos, en donde si tenemos suficientes eventos y por ende una buena estadística entonces que la red sea capaz de estimar X_{max} de manera que logremos una reconstrucción y un estudio más amplio tanto para energías bajas desde $10^{16.5}$ eV que cubran la segunda rodilla hasta energías de 10^{19} eV.

Apéndice A

Instalación de ROOT

ROOT es un framework para el procesamiento de datos creado en el CERN. Las aplicaciones ROOT son utilizadas para analizar datos y realizar simulaciones, es posible utilizar esta herramienta para, entre otras cosas, guardar datos (y cualquier objeto en C++) en forma binaria comprimido en un archivo .root. ROOT proporciona una estructura de datos, el TTree, que es extremadamente potente para acceder rápidamente a grandes cantidades de datos, con órdenes de magnitud más rápido que acceder a un archivo normal, podemos utilizarlo para acceder a los datos guardados en uno o varios archivos ROOT desde la PC, la web y sistemas de entrega de archivos a gran escala utilizados, los árboles ROOT distribuidos en varios archivos se pueden encadenar y acceder como un objeto único, lo que permite bucles sobre grandes cantidades de datos, ROOT proporciona poderosas herramientas matemáticas y estadísticas para operar con sus datos, los datos también pueden generarse siguiendo cualquier distribución estadística y modelarse, lo que permite simular sistemas complejos, los resultados se pueden mostrar con histogramas, diagramas de dispersión, funciones de ajuste. Los gráficos ROOT se pueden ajustar en tiempo real, estos resultados pueden ser utilizados para su publicación.

La página oficial de ROOT es la siguiente: <https://root.cern.ch/> en donde podemos encontrar más información y todo lo necesario para su descarga e instalación.

Instalación de librerías

Antes de comenzar la descarga de ROOT es necesario instalar los prerequisites de ROOT. En el siguiente enlace se encuentran las instrucciones y librerías necesarias según la distribución de Linux que utilices: <https://root.cern/install/dependencies/>

Para instalar puedes usar

apt-get install <nombre_de_la_librería>

Descargando un binario pre combinado

En la página https://root.cern/install/all_releases/ nos encontramos con las distintas versiones de ROOT empezando por la más reciente, seleccionamos la versión

deseada, esto nos llevará a una nueva página en donde nos dirigimos a la sección "Binary distributions" y descargamos el archivo según nuestra distribución de Linux.

- Instalación

- a)* Selecciona un directorio en donde te gustaría instalar ROOT, llámémoslo \$Directorio.
mkdir \$Directorio
cd \$Directorio
Movemos el archivo binario de ROOT al directorio donde lo instalaremos
mv /Descargas/nombre_del_archivo
- b)* Descomprimos el archivo
tar -xzf nombre_del_archivo
- c)* Entramos a las carpetas
cd ROOT
cd bin
- d)* Tecleamos source thisroot.sh
y posteriormente
\$ root

Con esto tenemos ROOT instalado, con .q salimos de la aplicación.

Apéndice B

Instalación de CONEX

Instalación de librerías

Antes de descargar el software de CONEX es necesario asegurarnos de que tenemos las librerías requeridas, de lo contrario no será posible hacer una instalación exitosa.

Los paquetes necesario pueden ser encontrados en la misma página de las librerías de ROOT tratadas en el apéndice A, en la página <https://root.cern/install/dependencies/> y corresponden a **Most common optional packages**.

Otra librería importante que instalar es subversion:

ej.

```
sudo apt-get install subversion
```

Descargando el software

Para obtener el software de CONEX nos dirigimos a la página oficial de Corsika a sección de descargas o al siguiente enlace: <https://web.iikp.kit.edu/corsika/download/>, en el cual requerimos un usuario y contraseña los cuales pueden ser obtenidos mediante una solicitud, para esto consulte la página oficial <https://www.iap.kit.edu/corsika/> o el enlace <https://www.iap.kit.edu/corsika/79.php> donde encontrara las instrucciones para solicitar un usuario y contraseña.

■ Instalación

- a) Una vez que hemos descargado el software lo movemos a la carpeta donde lo deseamos instalar, utilicemos el mismo directorio que ROOT en el apéndice A

```
mv /Descargas/nombre_del_archivo
```
- b) Descomprimos el archivo

```
tar -xvzf nombre_del_archivo
```
- c) Entramos a las carpetas

cd ROOT

cd bin

nos aseguramos de que esta carpeta contenga el archivo thisroot.sh (solamente usamos ls y vemos que este contenido)

- d) Tecleamos en nuestra terminal "source thisroot.sh", damos enter, root, enter. Se nos abre root, nos salimos con .q
Tecleamos pwd y copiamos el directorio y nos dirigimos a home.
- e) Editamos el archivo de configuración de bash:
\$ gedit .bashrc, veremos el archivo de configuración de bash:
Añadimos una línea en la que escribiremos "source /el/directorio/copiado/thisroot.sh"
Guardamos los cambios y cerramos.
Una vez hecho esto escribimos root -l. luego presionamos enter.
A partir de esto siempre que abramos una terminal se cargará por default ROOT.
- f) Nos dirigimos a la carpeta donde se encuentra el archivo descomprimido de CONEX
cd /Directorio
- g) Entramos a la carpeta descomprimida de CONEX, vemos que tenemos un archivo de nombre *README* y procedemos a leerlo

nano README

Aquí encontraremos información del software así como las instrucciones para instalarlo, las cuales son:

1. Asegurarnos que el directorio de ROOT esté apropiadamente escrito, esto será así porque modificamos el archivo de configuración de bash.
2. Crear las librerías de los modelos hadrónicos.
3. Divertirnos.

- h) Procedemos a seguir los pasos:
 1. Tecleamos echo \$ROOTSYS/bin y nos aseguraremos de que la ruta de ROOT está escrita apropiadamente.
 2. Para crear las librerías de los modelos hadrónicos utilizamos el comando make nombre_del_modelo_hadrónico o make all para crear todos los modelos hadrónicos disponibles.
 3. Divertirnos.

Apéndice C

Instalación de APE

Para más información visite <https://gitlab.com/lukasnellen/ape>

■ Instalación

- a)* Seleccionar un directorio donde instalaremos el software
Llamemoslo \$Directorio y creamos una carpeta llamada ape
mkdir ape
e ingresamos
cd ape
- b)* Descargamos la fuente:
Descargamos la última versión usando
git clone <https://gitlab.com/lukasnellen/ape.git>
- c)* Movemos la carpeta ape a una nueva llamada externals y entramos en esta:
mv ape externals
cd externals
- d)* Creamos un archivo `~/.aperc`
cp ape.rc.example ape.rc
- e)* Seleccionamos un directorio en donde nos gustaría compilar el software, llamémoslo \$ build
mkdir build
y uno más para instalarlo, llamémoslo \$install
mkdir install
mkdir distfiles
- f)* Modificamos el archivo `~/.aperc`
El contenido de este archivo debe tener el siguiente aspecto:
base = %(home) /Directorio/ape/externals/install
build = %(home) /Directorio/ape/externals/build
mirrors = mx us
jobs = 1
distfiles = %(home) /Directorio/ape/externals/distfiles
- g)* Instalamos

`./ape -j4 install externals --rc=ape.rc`

- h)* Usa `eval '$Directorio/ape sh externals -rc=$Directorio/ape.rc'` para levantar las variables.

Apéndice D

Instalación de Offline

Instalación de librerías

Antes de descargar el software de Offline será necesario instalar las librerías requeridas, de lo contrario no será posible realizar una instalación exitosa.

Los siguientes paquetes serán necesarios:

- `sudo apt-get install build-essential`
- `sudo apt-get install dpkg-dev cmake g++ gcc binutils libx11-dev libxpm-dev libxft-dev libxext-dev python libssl-dev`
- `sudo apt-get install gfortran libpcre3-dev xlibmesa-glu-dev libglew1.5-dev libftgl-dev libmysqlclient-dev libfftw3-dev libcfitsio-dev graphviz-dev libavahi-compat-libdnssd-dev libldap2-dev python-dev libxml2-dev libkrb5-dev libgsl0-dev`
- `sudo apt-get install libboost-all-dev libboost1.67-dev libboost-thread1.67-dev libboost-program-options1.67-dev libboost-system1.67-dev libboost-math1.67-dev libboost-test1.67-dev libboost-python1.67-dev`
- `sudo apt-get install binutils dpkg-dev g++ gcc gfortran giflib-tools graphviz-dev libavahi-compat-libdnssd-dev libbz2-dev libfftw3-3 libfftw3-dev libftgl-dev libglew-dev libglu1-mesa-dev libgsl-dev libkrb5-dev libldap2-dev libmysqlclient-dev libpcre3-dev libpng-dev libssl-dev libtiff-dev libx11-dev libxext-dev libxft-dev libxi-dev libxml2-dev libxmu-dev libxmuu-dev libxpm-dev make python-dev subversion`
- `sudo apt-get install libmysqlclient-dev libbz2-dev liblzma-dev dh-exec libxi-dev`
- `sudo apt-get install libssl-dev libncurses-dev libncurses5-dev doxygen`
- `sudo apt-get install graphviz latexdraw latex209-base latex209-bin latex209-src svngit svn2git`
- `sudo apt-get install libbz2-dev zlib1g-dev libx11-dev libxft-dev x11proto-xext-dev libxext-dev libmysqlclient-dev`
- `sudo apt install cmake-curses-gui`

- `sudo apt-get install libhdf5-dev`

Descargando el software

El repositorio de Auger contiene las versiones recientes de todos los programas de Auger. En particular tiene la última versión de Offline ubicada en `auger/Offline/trunk`.

Para copiar este directorio a su computadora use
`svn checkout https://augsvn.iap.kit.edu/svn/auger/Offline/trunk`

■ Instalación

- Seleccionemos un directorio donde instalar Offline, usaremos el mismo que ape en el apéndice C
`$Directorio/ape`
`mkdir offline`
`cd offline`
- Copiamos el directorio trunk
`svn checkout https://augsvn.iap.kit.edu/svn/auger/Offline/trunk`
- Seleccionamos una ruta para la instalación y un directorio para la compilación, vamos a llamarlos `install_dir` y `build_dir`, respectivamente
`mkdir install_dir`
`mkdir build_dir`
y entramos a la ruta de compilación
`cd build_dir`
- Usamos el siguiente comando para la compilación:
`ccmake ../trunk/ -Dprefix=../install`
\$ trunk es el directorio de los archivos fuente.
- Presionamos `c` para configurar y presionamos nuevamente `c`
`make -j4`
`make -j4 install`
- levantamos las variables
`eval `/(home)/Directorio/ape/offline/install/auger-offline-config --env-sh``

Bibliografía

- [1] KUKEC MEZEK G., *Mass composition of ultra-high energy cosmic rays at the Pierre Auger Observatory*, Disertacion, Univerza V Novi Gorici Fakulteta Za Podiplomski Študij). Repositorio Observatorio Pierre Auger, 2019.
- [2] D. YOUNG R. *Cosmic-ray Energy Spectrum*, <https://masterclass.icecube.wisc.edu/en/analyses/cosmic-ray-energy-spectrum>, IceCube MasterClass an authentic astrophysics research experience.
- [3] VERGARA QUISPE INDIRA D., *Composición primaria de rayos cósmicos de ultra alta energía y estudios de modelos hadrónicos en el desarrollo de cascadas en el Observatorio Pierre Auger*, Tesis Doctoral, Universidad Nacional de la Plata, Facultad de Ciencias Exactas, Departamento de Física. Repositorio institucional de la UNPL, 2021.
- [4] ALFARO SOLIS J., *Los efectos gravitatorios cuánticos en los rayos cósmicos de energías ultra altas*, Facultad de Física, Pontificia Universidad Católica de Chile.
- [5] M.SC. SONJA MAYOTTE, *Study of the cosmic ray composition sensitivity of AugerPrime*, Disertación, der Fakultät Mathematik und Naturwissenschaften Fachgruppe Physik der Bergischen Universität Wuppertal, Repositorio Observatorio Pierre Auger, 2021.
- [6] DE ANGELIS ALESSANDRO, PIMENTA MÁRIO, *Introduction to particle and astroparticle physics*, Padua, Italia, Lisbon, Portugal, Springer.
- [7] GLOMBITZA J., *Deep-Learning based measurement of the mass composition of ultra-high energy cosmic rays using the surface detector of Pierre Auger Observatory*, Disertación, Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur, Repositorio Observatorio Pierre Auger 2021.
- [8] JAVIER A. OTAOLA, VALDES-GARCIA J. F., *Los rayos cósmicos: Mensajeros de las estrellas*, México D.F. Fondo de Cultura Económica, 1995.

- [9] A. M. HILLAS, *The origin of ultra-high-energy cosmic rays*, Ann. Rev.Astron. Astrophys, 1984.
- [10] CRUZ MORENO J., *Estudio de rayos cósmicos ultra energéticos en el Observatorio Pierre Auger: Nuevo modelo de profundidad atmosférica y su influencia en observables*, Tesis Doctoral, Universidad Nacional de la Plata, Facultad de Ciencias Exactas, Departamento de Física, 2012.
- [11] BONNEAU ARBELETCHÉ L., *Simulation of Extensive Air Showers and its Detection*, Tesis Doctoral, INSTITUTO DE FÍSICA DE SÃO CARLOS de la UNIVERSIDADE DE SÃO PAULO, repositorio de la Universidad de São Paulo (USP), 2021.
- [12] TUEROS MATIAS J., *Física de astropartículas de alta energía; refinamiento de técnicas de simulación y reconstrucción de rayos cósmicos en el Observatorio Pierre Auger*, Tesis Doctoral, Universidad Nacional de la Plata, Facultad de Ciencias Exactas, Departamento de Física, repositorio institucional de la UNPL, 2010.
- [13] E. VARELA , *The low-energy extensions of the Pierre Auger Observatory*, The Pierre Auger Collaboration, J. Phys.:Conf. Ser. 468 (2013) 012013.
- [14] GONZALO ASOREY H., *Reconstrucción de eventos con los detectores de superficie del Observatorio Pierre Auger de rayos cósmicos*, Tesis de Maestría, 2005.
- [15] A. AAB ET AL., **Reconstruction of Events Recorded with the Surface Detector of the Pierre Auger Observatory**, A. Aab et al 2020 JINST 15 P10021
- [16] THE PIERRE AUGER COLLABORATION., *The Pierre Auger Cosmic Ray Observatory*, In: Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 798 (Oct. 21, 2015). issn: 0168-9002. doi: 10.1016/j.nima.2015.06.058.
- [17] I. ALLEKOTTE ET AL., *The surface detector system of the Pierre Auger Observatory*, Nuclear Instruments and Methods in Physics Research Section A 586, 2008, DOI: <http://dx.doi.org/10.1016/j.nima.2007.12.016>(arXiv:0712.2832).
- [18] M. AGLIETTA ET AL., *Response of the Pierre Auger Observatory Water Cherenkov Detectors to Muons*, [Pierre Auger Collaboration] 2005 Proc. 29th ICRC (Pune), 7, p. 83.
- [19] X. BERTOU ET AL., *Calibration of the surface array of the Pierre Auger Observatory*, [Pierre Auger Collaboration,] 2006 Nucl. Instrum. Meth., A568, p. 839-841.
- [20] VARELA CARLOS E., *Estudio de la composición de rayos cósmicos en el rango de energía de $10^{17} - 10^{18.5}$ eV con extensiones del Observatorio Pierre Auger*, Tesis Doctoral, Facultad de Ciencias

Físico Matemáticas, Benemérita Universidad Autónoma de Puebla, Marzo 2015.

- [21] A. OBERMEIER, *The Fluorescence Yield of Air Excited by Electron Measured with the AIRFLY Experiment*, Thesis FZKA-7284, Karlsruher Institut für Technologie, 2007.
- [22] M. RISSE AND D. HECK, *Energy Release in Air Showers*, In: Astroparticle Physics 20.6 (Mar. 1, 2004), pp. 661–667. issn: 0927-6505.
- [23] VARELA, E., GABRIEL, I., QUIROZ, A., BÁEZ, L.A., SALAZAR, H., VILLASEÑOR, L, *Artificial Neural Network for the Study of Cosmic Rays*, In: Ruiz, P., Agredo-Delgado, V. (eds) Human-Computer Interaction. HCI-COLLAB 2019. Communications in Computer and Information Science, vol 1114. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-37386-3_9
- [24] H. M. J. BARBOSA ET AL., *Determination of the Calorimetric Energy in Extensive Air Showers*, In: Astroparticle Physics 22.2 (Nov. 1, 2004), pp. 159–166. issn: 0927-6505.
- [25] J. ABRAHAM ET AL., *Trigger and aperture of the surface detector array of the Pierre Auger Observatory*, [Pierre Auger Collaboration], 2010 Nucl. Instrum. Meth., A613, p. 29.
- [26] D. ALLARD ET AL., *The trigger system of the Pierre Auger Surface Detector: operation, efficiency and stability*, [Pierre Auger Collaboration], 2005 Proc. 29th ICRC (Pune), 7, p. 287.
- [27] CORSIKA – AN AIR SHOWER SIMULATION PROGRAM, Karlsruhe Institute of Technology Institute for Astroparticle Physics Campus North, <https://www.iap.kit.edu/corsika/79.php>
- [28] MARTIN ERDMANN, JONAS GLOMBITZA, *Deep Neural Network for the Reconstruction of the Shower Maximum X_{max} using the Water-Cherenkov Detectors of the Pierre Auger Observatory*, RWTH Aachen University, III. Physikalisches Institut A, Otto-Blumenthal-Str., 52056 Aachen, Germany.
- [29] GUARDINCERRI Y., LEONI OLIVERA M., *Detector de fluorescencia en el Observatorio Pierre Auger*, Fac. de Cs. Exactas y Naturales, Universidad de Buenos Aires, 2006.
- [30] SELIGMAN W., *Basic Data Analysis Using ROOT*, Nevis Laboratories, Columbia University, 2022.
- [31] JANIESCH, C., ZSCHECH, P., HEINRICH, K., *Machine learning and deep learning*, Electron Markets 31, 685–695, 2021. DOI: <https://doi.org/10.1007/s12525-021-00475-2>
- [32] AURÉLIEN GÉRON, *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*, Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, 2019.

- [33] DAMIEN ROLON-MÉRETTE, MATT ROSS, THADDÉ ROLON-MÉRETTE, KINSEY CHURCH, *Introduction to Anaconda and Python: Installation and setup*, University of Ottawa, 2020.
- [34] ANACONDA DOCUMENTATION, <https://docs.anaconda.com/anaconda/install/linux/>, 10 de Noviembre de 2022.
- [35] CÓMO INSTALAR ANACONDA EN UBUNTU 20.04, <https://ourcodeworld.co/articulos/leer/1609/como-instalar-anaconda-en-ubuntu-2004>, 2021.
- [36] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A. AND MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M. AND PRETENTHOFER, P., WEISS, R. AND DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M., DUCHESNAY, E., *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, volume 12, pages 2825–2830, 2011.
- [37] PANG B., NIJKAMP E., NIAN WU Y., *Deep Learning With TensorFlow: A Review*, University of California, Los Ángeles, Journal of Educational and Behavioral Statistics, 2020, Vol. 45, No. 2, pp. 227–248. DOI: 10.3102/1076998619872761
- [38] INSTALA TENSORFLOW CON PIP, <https://www.tensorflow.org/install/pip>
- [39] A. AAB ET AL., *The Pierre Auger Observatory Upgrade “Auger-Prime”*, The Pierre Auger Collaboration, Preliminary Design Report. arXiv:1604.03637v1 [astro-ph.IM], 2015.
- [40] S. ARGIRÒ ET AL., *The offline software framework of the Pierre Auger Observatory*, Nuclear Instruments and Methods in Physics Research A 580 1485–1496, 2007.
- [41] UPROOT DOCUMENTATION, <https://uproot.readthedocs.io/en/latest/>
- [42] BROWNLEE J., *Better Deep Learning. Train Faster, Reduce Overfitting, and Make Better Predictions*, Edition: v1.3, 2019.
- [43] ZOUBIN GHAHRAMANI, *Unsupervised Learning*, Gatsby Computational Neuroscience Unit, University College London, UK, 2004.
- [44] LUNA GONZALES J., <https://medium.com/soldai/tipos-de-aprendizaje-automático-6413e3c615e2>, 08 de Febrero de 2018
- [45] A. AAB ET AL., *Data-Driven Estimation of the Invisible Energy of Cosmic Ray Showers with the Pierre Auger Observatory*, [Pierre Auger Collaboration] In: Phys. Rev. D 100.8 (Oct. 25, 2019), p. 082003. doi: 10.1103/PhysRevD.100.082003.
- [46] ALANIA M. ET AL., *Air Shower Simulations*, Centro de Tecnologías de Información y Comunicaciones, Universidad Nacional de Ingeniería, Lima, Perú, Departamento de Astronomía y Astrofísica, Pontificia Universidad Católica de Chile en colaboración con

Facultad de Ciencias Físico Matemáticas , Benemérita Universidad Autónoma de Puebla, Puebla, México, 25 de Mayo de 2009.

- [47] RAYCHAUDHURI SAMIK, *INTRODUCTION TO MONTE CARLO SIMULATION*, Proceedings of the 2008 Winter Simulation Conference, Broomfield, C.O. 80021, U.S.A.
- [48] I. ANTICHEVA ET AL., *ROOT — A C++ framework for petabyte data storage, statistical analysis and visualization*, Elsevier BV, Computer Physics Communications, DOI: 10.1016/j.cpc.2009.08.005
- [49] AMAT RODRIGO J., <https://www.cienciadedatos.net/documentos/pystats05-correlacion-lineal-python.html>, marzo 2020.
- [50] SHARMA SIDDHARTH, SHARMA SIMONE, ATHAIYA A., *ACTIVATION FUNCTIONS IN NEURAL NETWORKS*, International Journal of Engineering Applied Sciences and Technology, Vol. 4, Issue 12, ISSN No. 2455-2143, Pages 310-316, 2020.
- [51] TENSORFLOW, *INSTALAR TENSORFLOW CON PIP*, <https://www.tensorflow.org/install/pip>
- [52] NIELSEN M, *Neural Networks and Deep Learning*, Determination Press, 2015.
- [53] GLOMBITZA J., *A Deep Learning-Based reconstruction of Air Showers at the Pierre Auger Observatory*, Tesis de maestría , Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur, 2017.
- [54] ROOT: ANALYZING PETABYTES OF DATA, SCIENTIFICALLY, <https://root.cern/>
- [55] THE PIERRE AUGER COLLABORATION ET AL., *Deep-learning based reconstruction of the shower maximum Xmax using the water-Cherenkov detectors of the Pierre Auger Observatory*, Journal of Instrumentation, 2021
- [56] MARTINA BOTTI A., *Determination of the chemical composition of cosmic rays in the energy region of 5 EeV with the AMIGA upgrade of the Pierre Auger Observatory*, Tesis doctoral, Departamento de Física del KIT Instituto de Tecnología de Karlsruhe (KIT) y Universidad Nacional de San Martín (UNSAM), 8 de noviembre del 2019.
- [57] JAROS LAW STASIELAK FOR THE PIERRE AUGER COLLABORATION, *AugerPrime - The upgrade of the Pierre Auger Observatory*, Institute of Nuclear Physics Polish Academy of Sciences, Radzikowskiego Kraków, Poland, 18 de Octubre de 2021.
- [58] CURSO DE SUPERVISORES DE INSTALACIONES RADIACTIVAS (IR) MÓDULO BÁSICO, INTERACCIÓN DE LA RADIACIÓN CON LA MATERIA,

https://csn.ciemat.es/MDCSN/recursos/ficheros_md/764096047_1572009112411.pdf

- [59] JELENA PETEREIT, *Optimization of Radio Reconstruction for Inclined Air Showers with AERA at the Pierre Auger Observatory*, Fakultät Mathematik und Naturwissenschaften, Fachgruppe Physik der Bergischen Universität Wuppertal, Alemania, Marzo de 2022.