INSTITUTO POLITÉCNICO NACIONAL

CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN
LABORATORIO DE CIENCIA DE DATOS

GENERACIÓN DE CASCADAS ATMOSFÉRICAS USANDO REDES GENERATIVAS ADVERSARIAS

T E S I S

PARA OBTENER EL TÍTULO DE:

MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:

JOSÉ DE JESÚS DANIEL AGUIRRE ARZATE

TUTORES:

DR. RICARDO MENCHACA MÉNDEZ

DR. LUKAS NELLEN FILLA

México, CDMX
1 de junio de 2022

Declaración de Autoría

I, José de Jesús Daniel AGUIRRE ARZATE, declare that this thesis titled, «Generación de cascadas atmosféricas usando redes generativas adversarias» and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

«*If I have seen further it is by standing on the shoulders of Giants.*»

Issac Newton

INSTITUTO POLITÉCNICO NACIONAL

Resumen

CIC

Centro de Investigación en Computación

Maestría en Ciencias de la Computación

Generación de cascadas atmosféricas usando redes generativas adversarias

por José de Jesús Daniel AGUIRRE ARZATE

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too…

Agradecimientos

The acknowledgments and the people to thank go here, don't forget to include your project advisor…

Índice general

Índice de figuras

List of Algorithms

Índice de cuadros

Índice de Abreviaturas

HEP	**H**igh **E**nergy **P**hysics
GEANT	**GE**oemtry **AN**d **T**racking
CORSIKA	**CO**smic **R**ay **SI**mulations for **KA**scade
GAN	**G**enerative **A**dversarial **N**etworks
DESY	**D**eutsches **E**lektronen-**SY**nchrotron
HAWC	**H**igh-**A**ltitude **W**ater **C**herenkov
SNOLAB	**S**udbury **N**eutrino **O**bservatory **LAB**oratory
CERN	**C**onseil **E**uropéen pour la **R**echerche **N**ucléaire
HL-LHC	**H**igh **L**uminosity **L**arge **H**adron **C**ollider
LAr	**L**iquid-**Ar**gon
EAS	**E**xtensive **A**ir **S**hower
ASICO	**A**ir shower **SI**mulation and **CO**rrelation

Constantes Físicas

Speed of Light $\quad c_0 = 2.997\,924\,58 \times 10^8\,\mathrm{m\,s^{-1}}$ (exact)

Índice de Símbolos

eV	energy	J
a	distance	m
P	power	W ($\mathrm{J\,s^{-1}}$)
ω	angular frequency	rad

For/Dedicated to/To my...

Capítulo 1

Introduccion

En este capítulo se muestra la principal motivación para el desarrollo de esta tesis, así como una breve perspectiva de la problemática general. Como capítulo introductorio este contiene la introducción, el planteamiento del problema, el objetivo, las fronteras del estudio y la estructura del escrito.

1.1. Introducción

En el área de la física de altas energías (HEP) las técnicas de aprendizaje máquina siempre han estado presentes. Debido a la sorprendente efectividad de técnicas modernas del aprendizaje profundo, se comenzó a adaptar y desarrollar estos métodos en todos los rubros del campo. Algunas de las aplicaciones van desde los enfoques que se tienen en la parte experimental, la fenomenología o en el análisis teórico de los eventos.

En los experimentos más importantes del campo, el tratamiento y análisis de datos es una tarea fundamental. Técnicas como, árboles de decisión, máquinas de soporte vectorial, algoritmos genéticos, entre otras, fallan cuando la dimensionalidad de los datos aumenta. Como referencia de la alta dimensionalidad que hay, en el gran colisionador de hadrones (LHC), las colisiones ocurren con una frecuencia aproximada de 40Mhz, además cada colisión genera un gran número de partículas. Por ejemplo, en el caso particular del LHC, se tiene alrededor de $O(10^8)$ sensores para la detección de partículas.

Debido a que las observaciones son fundamentalmente probabilísticas se tiene un modelo estadístico que describe la probabilidad de observar un evento dado los parámetros de una teoría. Dado que el modelo de los datos experimentales no se conoce explícitamente, la alta dimensionalidad y a los grandes volúmenes de datos generan un problema. Sin embargo, se tiene acceso a muestras de datos generados por simuladores estocásticos que modelan la física de las interacciones.

Herramientas como PYTHIA, HERWING, GEANT, CORSIKA, se les denomina como simuladores Monte Carlo, los cuales cumplen con dos necesidades básicas. La primera, es aproximar el modelo estadístico que genera cada evento, la segunda, es generar una base de datos de simulaciones realistas.

Debido a que los simuladores describen las interacciones de las partículas con la materia, son computacionalmente demandantes y se llevan gran parte del presupuesto computacional de las colaboraciones experimentales [3], de modo que el desarrollo de simuladores rápidos es esencial.

Simuladores como GEANT y CORSIKA generan una excelente descripción de interacciones hadrónicas, sin embargo, son lentos para simular eventos de altas energías. De ahí la razón por la cual en los últimos años ha surgido un gran interés por usar modelos generativos para acelerar simuladores y tal vez llegar a usar estos métodos directamente en datos

generados por colisiones reales y hacer ajustes al momento.

El presente trabajo está fundamentado en el desarrollo de algoritmos de aprendizaje máquina profundo, específicamente el uso de modelos generativos implícitos como arquitecturas adversarias (GAN) para la generación de cascadas atmosféricas, debido a la necesidad de generar simulaciones precisas de una manera más rápida, a consecuencia de futuros requerimentos computacionales en las colaboraciones experimentales, por ejemplo, la actualización de alta luminosidad del LHC (HL-LHC)[3].

1.2. Planteamiento del problema

El planteamiento se fundamenta en lo siguiente:

Será posible diseñar un método que utilice redes neuronales generativas que logre simular cascadas atmosféricas precisas y así reducir el tiempo computacional de la generación de simulaciones mediante métodos tradicionales.

1.3. Objetivo de la tesis

Objetivo principal:

- Diseñar y implementar una red generativa adversaria para generar cascadas atmosféricas acordes a simulaciones de detectores de rayos cósmicos.

Objetivos particulares:

I. Obtener datos de simulaciones acordes a cascadas atmosféricas.

II. Diseñar y entrenar una red generativa adversaria para la simulación de cascadas atmosféricas.

III. Ajustar la dimensión del vector latente del modelo y comparar resultados.

IV. Obtener vectores latentes promedio y registrar como afecta a la cascada saliente.

1.4. Delimitación del tema

El presente trabajo se limita a utilizar al menos dos arquitecturas generativas en forma funcional para así lograr generar respuestas de detectores a una cascada atmosférica extensa. Enfatizando la rapidez de generación de nuevas muestras.

Cabe destacar que el trabajo se enfoca en la parte electromagnética de una cascada atmosférica, pero no se limita al análisis de otro tipo de cascadas de partículas o en su defecto, simulaciones de otro tipo de fenómenos físicos que involucran una multitud de procesos probabilísticos. Aunque no se pueda asegurar que el método presentado sea el mejor para la acelerar simulaciones, próximos estudios deben de intentar desenredar los parámetros latentes y probar la efectividad de otro tipo de arquitecturas.

Es necesario subrayar que este trabajo sólo utilizará arquitecturas adversarias (GAN) para acelerar simulaciones.

1.5. Organización de la tesis

El **Capítulo 1** presenta una breve y concisa introducción a la problemática principal, el planteamiento del problema, el objetivo principal y objetivos particulares del trabajo.

El **Capítulo 2** muestra el estado del arte, así como una breve explicación fenomenológica de las cascadas atmosféricas y la correspondiente operación básica de simuladores modernos. También se introducen las problemáticas principales relacionadas a los simuladores Monte Carlo.

En el **Capítulo 3** se introduce el fundamento teórico de los modelos generativos así como conceptos fundamentales del aprendizaje no supervisado. Asimismo se hace especial énfasis en las arquitecturas generativas y la teoría que las respalda.

El **Capítulo 4** presenta la metodología experimental que este trabajo sigue para lograr los objetivos propuestos.

En el **Capítulo 5 y 6** presentan los resultados obtenidos junto con una discusión de ellos y las conclusiones a las que este trabajo llegó.

Capítulo 2

Estado del arte

El presente capítulo proporciona el estado del arte mediante la revisión de conceptos y trabajos referentes a cascadas atmosféricas, detectores de partículas, simuladores y modelos generativos para establecer el fundamento del desarrollo de este trabajo.

2.1. Antecedentes

El problema de reducir el costo computacional que colaboraciones experimentales dedican a simulaciones en la física de altas energías (HEP), ha tenido mucha atención en los últimos años. Por esa razón, estudiar técnicas de aprendizaje máquina aplicadas a procesos experimentales, se incluyeron como un área estratégica de inversiones iniciales [3], para enfrentar futuros desafíos que actualizaciones como HL-LHC presentarán en los próximos años.

El aprendizaje máquina siempre ha estado presente en los flujos de trabajo de experimentos en HEP, sin embargo, técnicas modernas de aprendizaje profundo han comenzado a introducirse a procesos de análisis. En específico, se ha visto que los modelos generativos permiten acelerar los tiempos de generación de simulaciones, debido a lo anterior, es de suma importancia estudiar estas técnicas para así poder agregarlas a la próxima generación de simuladores, GEANTV, CORSIKA8, etc.

Algunos investigadores que han comenzado a usar modelos generativos para acelerar simulaciones son, Paganini [12, 13], Erdmann [6, 7], Carminati [1, 2], Glombitza [9, 5]. Paganini *et al*, desarrollan una arquitectura llamada CaloGAN basada en redes generativas adversarias para acelerar simulaciones de cascadas de partículas en calorímetros LAr y así lograr generar cascadas electromagnéticas tridimensionales con una reducción de tiempo computacional cinco órdenes de magnitud menor de lo que le toma a GEANT4.

Erdmann *et al*, usan una arquitectura WGAN para mejorar la estabilidad del entrenamiento y así poder reconstruir propiedades de la cascada simulada. El modelo se condiciona a un parámetro físico y logran generar simulaciones de un arreglo de calorímetros.

Carminati *et al*, presentan una red generativa adversaria convolucional tridimensional para generar la deposición energética de partículas en calorímetros de alta granularidad. Además, este trabajo es parte del proyecto GEANTV.

Glombitza usa redes convolucionales para reconstruir el máximo y la energía de una cascada atmosférica usando simulaciones de un arreglo de detectores, generadas con CORSIKA. Cabe señalar que el trabajo es parte de la colaboración Pierre Auger.

En ambos casos Paganini y Erdmann, proporcionan un modelo generativo capaz de generar simulaciones de cascadas de partículas acordes a algunos simuladores Monte Carlo

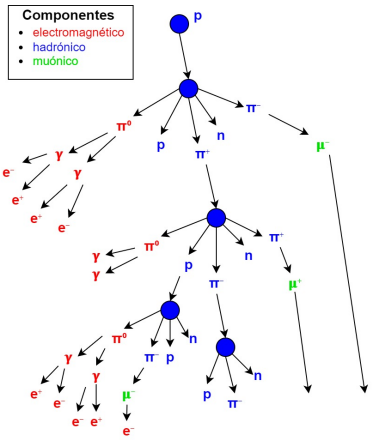

FIGURA 2.1: Mecanismo de la interacción de una partícula primaria con un componente atmosférico.

existentes. Consecuentemente, estos resultados muestran la posible efectividad de los modelos generativos para reducir los costos computacionales en los actuales flujos de trabajo experimentales en HEP.

2.2. Cascadas Atmosféricas

Problemas como conocer la naturaleza de la masa, la naturaleza de la antimateria, la dimensionalidad del espacio o lograr la unificación de las fuerzas fundamentales; se tienen que abordar tanto de los enfoques teórico como experimental, para así intentar responder a preguntas fundamentales como, ¿Cuántas partículas hay?, ¿Cuáles son sus propiedades y cómo interactúan?.

Estudiar la fenomenología de los rayos cósmicos es de vital importancia para poder responder las preguntas anteriores. En consecuencia laboratorios como DESY, SNOLAB, CERN, en conjunto con experimentos como el observatorio Pier Auger o el observatorio HAWC, entre otros; representan el actual estado del arte del campo de la física de altas energías. Por ejemplo, las partículas generadas en las colisiones del LHC viven por fracciones de segundo, lo cual hace que encontrar indicadores de que una partícula fue creada a pesar de nunca ser detectada, sea una tarea artesanal.

Los **rayos cósmicos** son partículas aceleradas en la galaxia o en objetos astrofísicos extragalácticos que al impactar con la atmósfera terrestre generan **cascadas de partículas**. El consenso general es que rayos cósmicos debajo de energías de $3 \cdot 10^6$ *GeV* son acelerados en los remanentes de supernovas galácticas y para energías mayores no se tiene una clara idea de que es lo que acelera a estas partículas, lo único que se tiene claro es de que dichas partículas tienen un orígen extragaláctico. A modo de comparación, estas fuentes aceleran partículas en tres órdenes de magnitud mayor que el equivalente energético del LHC.

Cuando la energía de los rayos cósmicos sobrepasa significativamente 1000 *GeV*, estos tienen que ser estudiados por las cascadas de partículas que generan en la atmósfera. Dado que la mayoría de las cascadas son iniciadas por hadrones con energías que van desde 1000 *GeV* hasta energías mayores a 10 *TeV*, que al entrar isotrópicamente a la atmósfera producen un gran número de productos secundarios en una serie de colisiones sucesivas con los núcleos de los constituyentes atmosféricos ie. N_2, O_2, Ar (Figura 2.1). Estos productos secundarios resultantes se comportan de una manera similar, mientras se van propagando a través de la atmósfera.

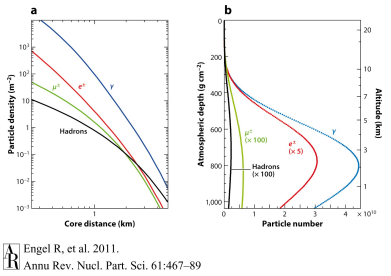

FIGURA 2.2: Distribución logitudinal promedio de las cascadas inducidas por un protón a 10^{19} GeV. Figura obtenida de [4].

Las cascadas de partículas descritas anteriormente son conocidas como **cascadas atmosféricas extensas** *EAS* [10]. Estas cascadas atmosféricas se propagan longitudinalmente a lo largo de la dirección de incidencia de la partícula primaria, debido al momento transversal de los productos secundarios la cascada también se extiende lateralmente. Como referencia de la complejidad que se tiene, una partícula primaria altamente energética puede crear una cascada gigante de partículas que se propaga esencialmente a la velocidad de la luz a través de la atmósfera y puede alcanzar el nivel del mar si el evento es lo suficientemente energético, del mismo modo, el número de productos secundarios es del orden $O(10^10)$ (Figura 2.2).

2.2.1. Propiedades de las cascadas atmosféricas

Algunas propiedades que caracterizan a las cascadas atmosféricas son:

- $\boldsymbol{E_0}$ $[eV]$: energía de la partícula primaria.
- **N** (*shower size*): número total de partículas producidas en un nivel en particular de la atmósfera. Es una función que depende de la energía E_0 el ángulo de incidencia cenit y a altura de la primera interacción del evento primario en la atmósfera h_1.
- **Xmax** $[gcm^-2]$: profundidad de máximo desarrollo, medida desde la parte más alta de la atmósfera. Se desfasa a profundidades mayores conforme la energía del primario se incrementa.
- **Shower Axis**: extensión del vector de momento del primario incidente en la dirección de propagación de la cascada.
- **Dirección de arribo**: dirección de incidencia de la partícula primaria determinada por sus ángulos acimut y cenit.

2.2.2. Métodos de detección

Los principales detectores de cascadas atmosféricas son arreglos de detectores espaciados uno del otro en distancias que dependen de la energía de los rayos cósmicos que se quieren observar. Para energías de 10^6 GeV, la distancia entre los detectores deben de ser del orden de decenas de metros y para energías que sobrepasan 10^9 GeV, la distancia es del orden de miles de metros, por ejemplo en el observatorio Pierre Auger la distancia entre detectores es de 1500 m. Diferentes métodos observacionales como detectores de aire cherenkov o detectores de fluorescencia son combinados con estos arreglos como en el caso del observatorio Pierre Auger (Figura 2.3).

Sin importar el tipo de sistema de detección que se use, los datos adquiridos representan a la cascada en una etapa en particular de su desarrollo, así como una foto instantánea de la

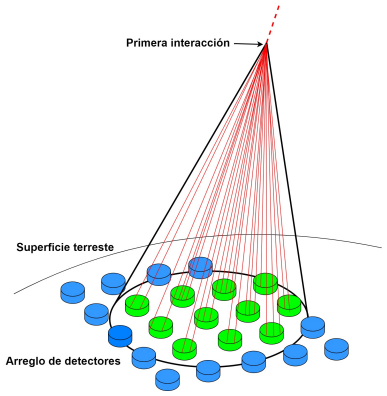

FIGURA 2.3: Esquema de una cascada atmosférica extensa. El rayo cósmico primario (línea punteada) interactúa con constituyentes atmosféricos (a una altura típica de $\sim$ 20 km) y produce una cascada de parítculas elementales (líneas rojas solidas). Un arreglo de detectores (que cubren áreas alrededor $\sim km^2$) detectan el arribo de las partículas.

cascada, en el plano de observación. Con estos datos se puede conocer información básica que caracteriza a la cascada atmosférica, así como los tiempos de arribo de las partículas cargadas, fotónes asociados no ópticos, las distribuciones laterales de partículas y fotónes en el plano de observación a una profundidad atmosférica específica. Las propiedades anteriores son inmediatamente accesibles con un arreglo de detectores simple.

La reconstrucción de la partícula primaria depende del modelo hadrónico de interacciones que use el simulador Monte Carlo.

2.3. Análisis de las cascadas atmosféricas

Las cascadas atmosféricas están caracterizadas por un delgado disco de partículas radialmente extenso que se propaga a la velocidad de la luz a través del eje de la cascada. El patrón de las cascadas es circular para cascadas verticalmente incidentes, mientras que la extensión longitudinal y lateral dependen principalmente de la energía de la partícula primaria. El esparcimiento lateral de las partículas en regiones bajas de la atmósfera, llega a cubrir áreas de varios kilómetros cuadrados, agregado a lo anterior, la mayoría de las partículas arriban en intervalos estrechos de tiempo que van desde unos cuantos nanosegundos, en la vecindad del eje de la cascada, hasta unos 10 *ns* a distancias mayores del núcleo de la cascada.

Eventos de baja energía alcanzan su máximo desarrollo en zonas altas de la atmósfera y se mitigan lentamente a mayor profundidad; los componentes que alcanzan a llegar a la superficie son los muones y neutrinos. Para eventos extremadamente energéticos las cascadas logran alcanzar su máximo desarrollo a nivel del mar mientras que sus componentes hadrónicos y electromagnéticos sobrevivientes, son absorbidos en la superficie terrestre y los muones resultantes altamente energéticos continúan propagándose bajo tierra.

Como regla de dedo se puede decir que en promedio una cascada atmosférica está constituida al 1 % por hadrones, alrededor del 10 % son muones y el 90 % o más son electrones o positrones. También para las primeras estimaciones energéticas de la primaria, se tiene que cascadas verticales a una altitud de 5 *km* tienen una energía 1 *GeV*, 3 *GeV* para alturas entre 2,5 *km* y 3 *km* 10 *GeV* a nivel del mar.

Para poder visualizar la ocurrencia de estos eventos en distintos regímenes energéticos se puede estudiar el número de partículas que cruzan un área en un tiempo dado, conocido como flujo cósmico. Este flujo sigue una ley de potencias con la forma $\frac{1}{E^3}$, en Figura 2.4 se

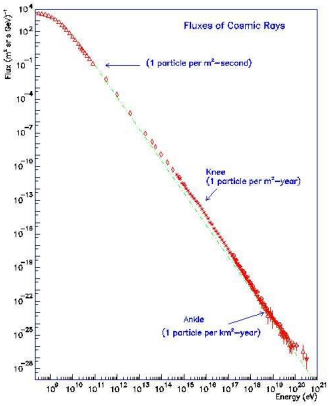

FIGURA 2.4: Espectro de energía de los rayos cósmicos.
Figura obtenida de [14].

puede ver energías alrededor de $10^12\ eV$ el flujo es de 10 partículas primarias por minuto y m^2, para energías entre $10^18\ eV$ y $10^19\ eV$ se tiene 1 partícula primaria por año y km^2, en estos regímenes energéticos las estadísticas son pobres y las incertidumbres altas.

Los observables importantes que se deben de obtener para poder reconstruir una cascada son, el tiempo de arribo t_i al detector i respecto al tiempo de referencia t_0, la densidad de partículas ρ_i y la posición del detector con respecto al sistema de referencia (x_i, y_i). Con la distribución lateral de las partículas su puede adquirir la localización del eje de la cascada y el tamaño de la cascada para así obtener un estimado de la energía. Los parámetros anteriores son clasificados como accesibles directamente ya que no necesitan un análisis complejo para su adquisición.

Por último, parámetros indirectamente accesibles que se relacionan con la naturaleza de la partícula primaria, como el tipo de partícula, masa y carga, no se pueden extraer inmediatamente y requieren métodos sofisticados de análisis.

2.4. Simulaciones

Los simuladores de cascadas atmosféricas son de vital importancia para la evaluación e interpretación de datos experimentales. Las técnicas se reducen a crear e insertar un modelo de cascadas que corresponda a nuestro mejor entendimiento de la realidad, simular cascadas, comparar resultados con los datos experimentales, modificar el modelo o sus parámetros y intentar de nuevo; hacer ajustes pequeños al modelo y repetir hasta que se obtenga un consenso entre la predicción y el experimento.

Las cascadas iniciadas por primarios hadrónicos consisten en la superposición de dos tipos de cascadas, una hadrónica y otra electromagnética. La cascada electromagnética se entiende bien y solo posee problemas prácticos asociados al gran número de partículas participantes en el orden de $\sim 10^{10}$. Los programas computacionales que simulan cascadas hadrónicas o electromagnéticas de alta energía o cascadas atmosféricas completas son altamente complejos.

Para tomar en cuenta la complejidad computacional que se tiene, una simulación completa de una cascada debe incluir los componentes electromagnéticos y hadrónicos. Además se debe de tomar en cuenta todos los procesos relevantes, donde la mayoría son de naturaleza estocástica y muchos están en competencia entre sí. También se debe incluir los parámetros que especifican el estado de cada partícula como su masa, carga, energía, momento,

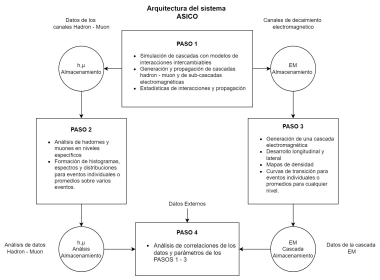

FIGURA 2.5: Diagrama de la arquitectura ASICO.

ubicación de su creación en el espacio tiempo x, y, z, t, la orientación angular respecto al marco de referencia y parámetros genéticos que revelan la altura de la interacción donde cada partícula fue creada. Estos observables son cruciales para análisis subsecuentes y para la comparación con datos experimentales.

2.4.1. Estrategia para simulaciones de EAS

La arquitectura del sistema ASICO sirve como base para entender el proceso de un simulador completo. ASICO fue el primer simulador que generaba cascadas detalladas al usar 12 parámetros que definen a cada partícula y este fue la base para el desarrollo de CORSIKA. Para simular una cascada completa se comienza con la simulación de la cascada hadrónica y se le llama *PASO 1*, este paso da lugar a datos como la elasticidad, distribuciones de interacciones hadrónicas en diferentes rangos de energías, entre otros; para la parte electromagnética de la cascada se le llama *PASO 2* y la combinación de las dos simulaciones para formar la cascada completa se le conoce como el *PASO 3* [10].

2.4.2. Problemática de las simulaciones

Los parámetros que determinan a cada partícula se deben de asignar en su punto de creación y requieren actualizarse después de cada proceso al cual está sujeta. Esto es al final de cada trayectoria particular, después de propagarse al siguiente punto de interacción o decaimiento y cuando pasan a un nuevo nivel de observación. En cada actualización la partícula y sus parámetros son guardados para su subsecuente análisis y evaluación de los datos de la cascada simulada.

Consecuentemente, la ejecución de programas que simulan cascadas requieren mucho tiempo computacional particularmente para cascadas energéticas donde el número de partículas involucradas se vuelve muy grande. La gran cantidad de datos producidos por estas simulaciones requieren también una gran capacidad de almacenamiento. La complejidad aumenta si los componentes atmosféricos cherenkov o de fluorescencia se incluyen, en este caso los datos corren el riesgo de divergir y métodos computacionales más sofisticados deben de ser usados para su análisis.

En general el desarrollo y la propagación de una cascada a partir del punto de iniciación (la primera interacción) hasta el nivel de observación consume más tiempo que el análisis subsecuente de los datos producidos.

- **Memoria:** Almacenamiento confiable de los datos crudos.
- **Rastreabilidad:** Rastreo de los parámetros que determinan a cada partícula.
- **Fenomenología:** Propagación de las partículas en la atmósfera tomando en cuenta todos los procesos a los que están sujetas.

- **Configuración inicial:** Correlación confiable entre los parámetros iniciales (*i.e.* modelos de interacción, propiedades del detector, propiedades de la primaria) con la cascada final
- **Tiempo:** Existe una relación lineal entre la energía de la partícula y el tiempo que lleva simular la cascada que genera.

2.5. Modelos generativos

Los modelos generativos son un tipo de aprendizaje no supervisado, que describen cómo se genera un conjunto de datos en términos de un modelo probabilístico. Al muestrear de dicho modelo se es capaz de generar datos no observados previamente. Típicamente el marco de trabajo de los modelos generativos involucra las siguientes partes [8].

- **Los datos:** Conjunto de observaciones, que se asumen ser generadas de acuerdo a una distribución de probabilidad desconocida.
- **El modelo:** Un modelo generativo que intenta imitar lo mejor posible, a la distribución que genera las observaciones. Este modelo es capaz de generar datos no observados que parecen haber sido generados con la distribución desconocida y no debe de generar datos conocidos.

2.5.1. Aplicación de modelos generativos

Algunas de las tareas modernas de los modelos generativos son:

- **Generación de datos novedosos:** Se generan datos nunca antes vistos que pueden ser utilizados para imitar fenómenos o para ayudar a flujos en modelos discriminativos con un pre entrenamiento autosupervisado.
- **Compresión de datos:** El modelo es capaz de aprender las características más importantes que determinan a la observación y así logra reducir la dimensionalidad del espacio de características donde vive la observación original.
- **Tecnologías de síntesis condicional:** Proporciona un método capaz de generar información novedosa condicionada a un dominio específico. Lo anterior permite una suerte de transformación de datos de un dominio a otro.

Las tareas anteriores han logrado avances en:

- Generacion de rostros humanos
- Transformación de imágenes
- Transferencia de estilos
- Texto a imagen
- Texto a voz
- Edicion de imagenes
- Super resolución
- Generación de objetos 3D
- Predicción de fotogramas en videos

Capítulo 3

Modelos generativos

Este capítulo proporciona una breve y concreta introducción a los modelos generativos, su fundamento teórico y algunas de las principales áreas de investigación. Además de mostrar las principales aplicaciones en HEP.

3.1. Introducción

Así como los modelos discriminativos han sido el centro del progreso en metodologías del aprendizaje máquina en los últimos años ya que es más sencillo monitorear su desempeño y así poder elegir la mejor metodología que se ajuste a la tarea. Los modelos generativos suelen ser más difíciles de evaluar, lo cual hace que encontrar aplicaciones industriales sea más complicado.

Los modelos generativos han probado su efectividad para generar muestras que son capaces de imitar a observaciones reales así como rostros humanos con StyleGAN de NVIDIA o GPT3 de openAI para generar texto. Los modelos anteriores han impulsado el interés para expandir el campo del aprendizaje de máquina a través de modelos que aprenden a generar muestras indistinguibles de observaciones reales. Los avances en el campo podrían ser fundamentales para el desarrollo de una máquina que haya adquirido una inteligencia comparable a la de los humanos.

El campo de los modelos generativos es diverso y la definición de los problemas toman una gran variedad de formas. Sin embargo, para cada tarea, los desafíos que se tienen son los mismos. Entender cómo es que el modelo maneja un alto grado de dependencias condicionales entre las características de la observación y como es que logra encontrar una observación viable, en un espacio de alta dimensionalidad, que concuerda con los datos observados a partir de un conjunto pequeño de observaciones, es de vital importancia para desarrollar metodologías más robustas.

3.1.1. Fundamento teórico

Como punto de partida se debe de reconocer la diferencia clave entre los modelos discriminativos y los modelos generativos.

Los modelos discriminativos estiman $p(y|x)$ - la probabilidad del observable y dada la observación x.

Los modelos generativos estiman $p(x)$ - la probabilidad de observar x.

En otras palabras, los modelos discriminativos intentan estimar la probabilidad de que una observación x pertenezca a la categoría y, y los modelos generativos intentan estimar la probabilidad de ver la observación x. Por lo tanto el modelo debe de ser probabilístico en vez de ser determinista y debe incluir un elemento estocástico que influencia las observaciones individuales generadas por el modelo.

3.2. Líneas de investigación

Detrás del creciente interés en la academia por los modelos generativos se encuentran dos razones con una gran importancia teórica, que se describen a continuación:

Se debe de buscar un entendimiento completo de cómo se generan las observaciones para así poder formar inteligencias artificiales más sofisticadas que van más allá de lo que pueden lograr los modelos discriminativos.

Es altamente probable que los modelos generativos sean centrales para futuros desarrollos en otros campos del aprendizaje máquina.

3.3. Modelos generativos

Un modelo generativo describe cómo se genera un conjunto de observaciones en términos de un modelo probabilístico. Al generar muestras de este modelo, se es capaz de generar observaciones nunca antes vistas.

Al tener un conjunto de observaciones que representen la entidad que se quiere generar. El objetivo del modelo generativo es generar nuevas observaciones que sigan las mismas reglas con las cuales las observaciones originales fueron generadas. Lo anterior es posible debido a que se asume que existe alguna distribución de probabilidad que explica, porqué ciertas observaciones son más probables de encontrarse en un conjunto y otras no.

El trabajo del modelo es imitar una distribución desconocida lo más cercano posible, para luego muestrear de ella y generar nuevas observaciones, distintas de las conocidas, que además parezca que son parte del conjunto de entrenamiento.

El marco de trabajo de los modelos generativos es el siguiente:

- Se tiene un conjunto de datos de observaciones X.
- Se asume que las observaciones X se generaron de acuerdo a una distribución de probabilidad desconocida p_{datos}.
- Se diseña un modelo generativo p_{modelo} que intenta imitar a p_{datos}.
- Se muestrea de p_{modelo} para generar observaciones que parecen ser obtenidas de p_{datos}.

Consideramos que p_{modelo} hace un buen trabajo si:

Puede generar observaciones que parecen ser obtenidas de p_{datos}.

Puede generar observaciones que son sustancialmente diferentes a las observadas en X. En otras palabras, el modelo no debería de reproducir cosas que ya conoce.

3.3.1. Conceptos básicos

Espacio Muestral
El espacio muestral es el conjunto de todos los posibles valores que una observación x puede tomar.

Función de densidad de probabilidad
Una función de densidad de probabilidad, $p(x)$, es una función que mapea un punto x del espacio muestral a un número entre 0 y 1. La suma de la función de densidad sobre todos los puntos del espacio muestral es igual a 1 para que sea una distribución bien definida.
Por definición tenemos que solo existe una p_{datos} pero existen infinitas distribuciones p_{modelo} que pueden estimar p_{datos}. Para encontrar una distribución adecuada se tiene que usar un modelo paramétrico.

Modelo paramétrico
Un modelo paramétrico, $p_\theta(x)$, es una familia de funciones de densidad que pueden ser descritas por medio de un número finito de parámetros, θ.

Verosimilitud
La verosimilitud $L(\theta|x)$ de un conjunto de parámetros θ es una función que mide la plausibilidad de θ, dado una observación x.
Se define como sigue:

$$L(\theta|x) = p_\theta(x)$$

Esto es, la verosimilitud de θ dada una observación x está definida como el valor de la función de densidad de probabilidad parametrizada por θ, en el punto x.
Para un conjunto de observaciones se tiene:

$$L(\theta|X) = \prod_{x \in X} p_\theta(x)$$

La expresión anterior resulta ser complicada de trabajar computacionalmente hablando, entonces se usa la verosimilitud logarítmica l.

$$l(\theta|x) = \sum_{x \in X} log p_\theta(x)$$

En otras palabras, la verosimilitud de un conjunto de parámetros θ es igual a la probabilidad de observar los datos bajo el modelo parametrizado por θ.

3.3.2. Modelos basados en la verosimilitud

Dado un conjunto de observaciones finito D, muestreado de un distribución p_{datos}, el objetivo de un modelo generativo es aproximar p_{datos} a través de una aproximación paramétrica p_θ, de tal manera que, se puede pensar que el modelo aprende los parámetros que minimicen alguna suerte de distancia entre p_{datos} y p_θ (). Matemáticamente se tiene el siguiente problema de optimización.

$$\min_{\theta \in \mathcal{M}} d(p_{datos}, p_\theta) \tag{3.1}$$

La verosimilitud es la primera métrica que se considera para medir la distancia entre el modelo y p_{datos}. A esta técnica se le conoce como estimación de la verosimilitud máxima (MLE).

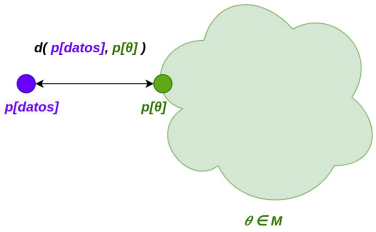

FIGURA 3.1: Esquema general del poblema de optimización (3.1).

MLE
Esta técnica permite estimar el conjunto de parámetros $\hat{\theta}$ del modelo $p_\theta(x)$, los cuales son los más probables de explicar los datos observados X.
Formalmente:

$$\begin{aligned}\hat{\theta} &= \arg\max_\theta l(\theta|X) \\ &= \arg\max_\theta \sum_{x\in X} log\, p_\theta(x)\end{aligned} \tag{3.2}$$

lo cual es equivalente a minimizar la divergencia KL.

$$\hat{\theta} = \arg\min_\theta D_{KL}(P_{datos}||P_\theta) \tag{3.3}$$

Para los modelos basados en la verosimilitud, se busca que se ajusten lo mejor que se pueda a los datos de entrenamiento o que idealmente, logren representar idénticamente a la distribución real p_{datos} y para nuevas observaciones x, el modelo debe de ser capaz de evaluar $p_\theta(x)$ de igual forma se debe de poder muestrear de la distribución $p_\theta(x)$. Por último el modelo debe obtener una representación latente significativa.

Los objetivos principales de los modelos basados en la verosimilitud son:

- Muestreo rápido
- Inferencia rápida
- Entrenamiento rápido
- Buena calidad de las muestras
- Buena compresión

Algunas de las clases más grandes de estos modelos

- Modelos de flujo
  - Flujos autorregresivos rápido para evaluar x arbitrarias, lento para muestrear
  - Flujos inversos autorregresivos lento para evaluar x arbitrarias, rápido para muestrear
- Modelos de variable latente
  - VAE lento para muestrear

A grandes rasgos los pasos que estos modelos siguen, son muestrear, evaluar la verosimilitud, entrenar y obtener una representación latente, pero si solo importara obtener muestras confiables se debe introducir la clase de modelos generativos implícitos.

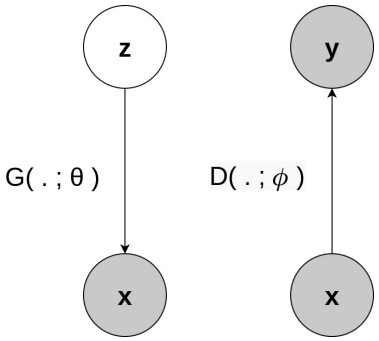

FIGURA 3.2: Diagrama simplificado de los modelos G_ϕ y D_ϕ. **x** son muestras de p_{datos} o de p_θ, **z** denota al vector de ruido aleatorio y **y** es la predicción del discriminador D aplicado a **x**.

3.3.3. Modelos implícitos

Los modelos implícitos no estiman una densidad de probabilidad explícitamente, solo se enfocan en la calidad de muestras generadas. De acuerdo al marco de trabajo de los modelos generativos, se comienza por tener muestras de una distribución $p_{datos} : x_1, x_2, \ldots, x_n$ luego se define un modelo paramétrico $q_\phi(z) = DNN(z\phi)$ que muestrea a partir de un vector latente muestreado de una fuente fijo de ruido (uniforme o gaussiana) $z \sim p(z)$.

Lo anterior quiere decir que $q_\phi(z)$ induce un modelo de densidad p_{modelo}, sin embargo, no se tiene una forma explícita de p_{datos} o p_{modelo}, solo se tiene acceso a las observaciones que generan. Lo anterior obliga a buscar una medida de distancia entre p_{datos} y p_{modelo} que no necesite la expresión explícita de p_{modelo} y que se comporte diferente a la estimación de verosimilitud máxima. Algunos ejemplos de dichas métricas son la divergencia de Jensen Shannon JSD o la métrica Earth Mover's Distance EMD.

GANS

Los modelos GAN son el primer ejemplo de un modelo que no se basa en MLE por consiguiente se estima el modelo generativo mediante un proceso adversario, el cual debe entrenar dos modelos simultáneamente

a) Un modelo generativo de variable latente G_θ que captura la distribución de los datos y genera muestras x a partir de z deterministicamente.

b) Un modelo discriminativo D_ϕ que estima la probabilidad de que una haya sido generada por p_{datos} o por G_θ.

Formalmente el objetivo de la GAN se escribe como sigue.

$$\min_\theta \max_\phi \quad \mathbb{E}_{x\sim p_{datos}}\left[logD_\phi(x)\right] + \mathbb{E}_{z\sim p(z)}\left[log(1 - D_\phi(G_\theta(z)))\right] \tag{3.4}$$

La función objetivo describe un juego minimax entre el generador G y el discriminador D, donde G intenta minimizar la probabilidad de que sus muestras sean clasificadas como falsas por D ($p_{datos} = p_\theta$) y D intenta maximizar la probabilidad para el problema de clasificación binaria ($p_{data} \neq p_\theta$).

La noción de distancia que se tiene con el objetivo está relacionada con JSD al considerar al discriminador D óptimo se tiene que el generador intenta optimizar la siguiente expresión.

$$V(G, D^*) = -log4 + 2D_{JSD}(p_{datos}||p_G) \tag{3.5}$$

Nos dice que para esta métrica, el generador óptimo para la función objetivo de la GAN se vuelve $p_G = p_{datos}$. Por último, para evitar la saturación de D se reformula las funciones de pérdida como sigue:

$$L^{(D)} = -\mathbb{E}_{x\sim p_{datos}}\left[logD_\phi(x)\right] + \mathbb{E}_{z\sim p(z)}\left[log(1 - D_\phi(G_\theta(z)))\right] \tag{3.6}$$

$$L^{(G)} = -\mathbb{E}_{z\sim p(z)}\left[logD_\phi(G_\theta(z))\right] \tag{3.7}$$

La evaluación de estos modelos aún es un problema abierto, a diferencia de otros modelos no se puede reportar estimados de la verosimilitud explícitos en conjuntos de evaluación, sin embargo, estos modelos son rápidos para muestrear, generan muestras sorprendentemente similares y interpolan correctamente entre las muestras.

WGAN

La arquitectura WGAN tiene como objetivo minimizar EMD y tiene la bondad de mejorar la calidad de las muestras generadas. Su correspondiente función objetivo es.

$$\min_G \max_{D\in D} \quad \mathbb{E}_{x\sim p_{datos}}\left[D(x)\right] + \mathbb{E}_{\tilde{x}\sim P_G}\left[D(\tilde{x})\right] \tag{3.8}$$

Este nuevo objetivo usa una nueva métrica para optimizar el generador que ataca las inestabilidades que surgen al usar JSD, de igual manera introduce la noción de continuidad de Lipschitz para estabilizar el entrenamiento de la GAN. En su versión donde la continuidad de LIpschitz se asegura mediante un término regularizador el objetivo cambia a

$$\min_G \max_{D\in D} \quad \mathbb{E}_{x\sim p_{datos}}\left[D(x)\right] + \mathbb{E}_{\tilde{x}\sim P_G}\left[D(\tilde{x})\right] + \lambda\mathbb{E}_{\hat{x}\sim P_{\hat{x}}}\left[\left(||\nabla_{\hat{x}}D(\hat{x})||_2 - 1\right)^2\right] \tag{3.9}$$

donde:

$$\hat{x} \leftarrow \epsilon x + (1-\epsilon)\tilde{x}$$

Conditional GAN

$$\min_G \max_D \mathbb{E}_{x,y}\left[logD(x, G(x)) + log(1 - D(x, y))\right] \tag{3.10}$$

3.3.4. Aprendizaje de representación

La idea principal detrás de aprendizaje representation es que en vez de tratar de modelar directamente el espacio muestral en altas dimensiones, se debe de intentar describir cada observación en el conjunto de entrenamiento, usando un espacio latente de menor dimensión para luego aprender una función que mapea puntos en el espacio latente al dominio original.

Hay que notar que el aprendizaje de representación no solo asigna valores a un conjunto de características del espacio latente para alguna observación si no que, aprende qué características son más importantes para poder describir las observaciones dadas y cómo generar estas características a partir del conjunto de entrenamiento.

Algunos de los modelos que se especializan en son la InfoGAN y la BiGAN.

Capítulo 4

Método experimental

En este capítulo se detallan las técnicas utilizadas así como el método experimental para la generación de cascadas atmosféricas.

4.1. Recolección y manejo de datos

Dado que el objetivo de este trabajo es implementar un modelo generativo que sea capaz generar simulaciones de cascadas atmosféricas, se necesita un banco de datos de dichas simulaciones. Una de las primeras maneras de obtener este conjunto de datos será tomar el simulador CORSIKA y generar el número de eventos necesarios para poder entrenar a la red neuronal. Como segunda opción de recolección se buscará un banco de simulaciones disponibles al público con características fundamentales como la energía de la primaria, la altura de máximo desarrollo, etc.

Como se ha mencionado en capítulos anteriores, las tareas que cada simulador conlleva para generar una cascada, incluyen subrutinas se encargan de la parte hadrónica, la parte electromagnética, el modelo de interacción, el rastreo de parámetros y procesamientos adicionales para reducir la carga computacional; por lo cual los datos que se utilizaran no deben de concentrarse en alguna de estas subrutinas para poder acotar el dominio de la simulación.

Para generar un resultado significativo sin pérdida de generalidad, se tomarán en cuenta las siguientes consideraciones:

- **Utilizar un número relativamente "grande" de simulaciones.**
  El número de observaciones debe ser un número "grande", como referencia se intentará obtener un conjunto de datos con al menos el mismo número de observaciones que el conjunto MNIST.
- **Tratar de estandarizar el conjunto de datos.**
  Para datos como los tiempos de arribo, el dominio de cada simulación diferirá en magnitudes que deben de tomarse en cuenta, por lo tanto técnicas de estandarización podrán ser utilizadas para tener una mejor estabilidad al entrenar el modelo.

4.2. Algoritmos

A continuación se proporciona el algoritmo de entrenamiento utilizado para la generación de cascadas.

4.2.1. WGAN

Una vez que se tenga el conjunto de entrenamiento compuesto de simulaciones de cascadas atmosféricas, se resolverá el problema de optimización planteado en (3.8) mediante el algoritmo (1), obtenido de [11].

Algorithm 1 WGAN with gradient penalty.

Require: λ, the gradient penalty coefficient. n_{critic}, the number of iterations per generator iteration. m, the batch size. α, β_1, β_2, Adam optimizer hyperparameters.
Require: w_0, initial critic parameters, θ_0 initial generator parameters.
  while θ has not converged **do**
    for $t = 1, \ldots, n_{critic}$ **do**
      for $i = 1, \ldots, m$ **do**
        Sample real data $\mathbf{x} \sim \mathbb{P}_r$, latent variable $\mathbf{z} \sim p(z)$, a random number $\epsilon \sim U[0,1]$.
        $\tilde{x} \leftarrow G_\theta(z)$
        $\hat{x} \leftarrow \epsilon x + (1-\epsilon)\tilde{x}$
        $L^{(i)} \leftarrow D_w(\tilde{x}) - D_w(x) + \lambda(\|\nabla_{\hat{x}} D_w(\hat{x})\|_2 - 1)^2$
      end for
      $w \leftarrow Adam(\nabla_w \frac{1}{m} \sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2)$
    end for
    Sample a batch of latent variables $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$
    $\theta \leftarrow Adam(\nabla_w \frac{1}{m} \sum_{i=1}^{m} -D_w(G_\theta(z)), \theta, \alpha, \beta_1, \beta_2)$
  end while

4.3. Desarrollo del método

A continuación se muestra el método experimental desglosado:

a) Obtener un banco de datos de simulaciones de cascadas atmosféricas acotadas a algún proceso seleccionado.

b) Hacer el preprocesamiento de datos adecuado para normalizar la entrada de datos y así estabilizar el entrenamiento sin dejar de tomar en cuenta las consideraciones teóricas que dicho procesamiento implica.

c) Elegir una arquitectura generativa adversaria y entrenarla para intentar generar nuevas cascadas.

d) Se tomarán las nuevas muestras y se inspeccionarán visualmente comparándolas con las observaciones originales.

e) Después de haber confirmado la similitud con las observaciones originales, se hará un análisis exploratorio de la dimensionalidad del vector latente y un análisis al proceso determinista de entradas y salidas de dicha red.

f) Una vez logrado lo anterior, cambiar de arquitectura y repetir el proceso, si se puede.

4.3.1. Mapa del método experimental

Capítulo 5

Resultados

Este capítulo muestra algunos de los resultados obtenidos por el método experimental presentado en el capítulo anterior.

5.1. Discusión de resultados obtenidos

A continuación se describirán los resultados por cada etapa del método experimental.

Obtención de datos y procesamiento

Para obtener el conjunto de datos, primero se optó por intentar generar una base de datos usando el simulador CORSIKA, sin embargo, se observo que la generación de un dataset de al menos 60000 observaciones, tomaría un tiempo inviable con el equipo de cómputo con el que se disponía. Del mismo modo, la complejidad adicional que la interfaz del programa y la inexperiencia con el mismo, agregaba una dificultad extra a la hora de configurar los parámetros del evento.

Por consiguiente, se decidió buscar una base de datos de simulaciones que estuvieran disponibles al público. De modo que el banco de datos que se usó fue tomado de [5], el cual contiene 100000 simulaciones de cascadas electromagnéticas (Tabla 5.1).

Para la selección y preprocesamiento de datos se seleccionó la señal que contiene los tiempos de arribo y la energía asociada a cada cascada. Después se hizo el ajuste propuesto en [5] a los tiempos de arribo 5.1, sin embargo el ajuste no lograba un buen entrenamiento, así que se hizo un mapeo lineal al dominio $[1, 2]$.

CUADRO 5.1: Parámetros de la base de datos.

Etiqueta	**Descripción**	**Dimensión**
showermax	Coordenadas del máximo de la cascada	$(N, 3)$
time	Tiempos de arribo al detector i	$(N, 9, 9, 1)$
Xmax	Altura de desarrollo máximo	$(N,)$
signal	Contador de particulas en detector i en 80 intervalos de tiempo	$(N, 9, 9, 80)$
logE	Energía de la partícula primaria	$(N,)$
showeraxis	Coordenadas del eje de la cascada	$(N, 3)$
showercore	Coordenadas del núcleo de la cascada	$(N, 3)$
mass	Masa de la partícula primaria	$(N,)$
detector	Coordenadas del detector i	$(81, 3)$

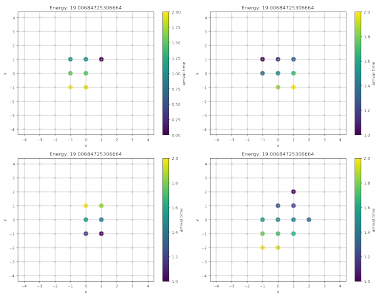

FIGURA 5.1: Gráficas de 4 eventos detectados por un arreglo de detectores con 1000 m de separación entre ellos, con su respectiva energía.

CUADRO 5.2: Parámetros de la base de datos.

Nombre	**Dimensión**	**Frecuencia**
cwagn_ld4_Ri_checkpoint.pth.tar	4	1/4
cwagn_ld8_Ri_checkpoint.pth.tar	8	1/3
cwagn_ld16_Ri_checkpoint.pth.tar	16	1/2
cwagn_ld32_Ri_checkpoint.pth.tar	32	1/2
cwagn_ld64_Ri_checkpoint.pth.tar	64	1/1
cwagn_ld128_Ri_checkpoint.pth.tar	128	1/1

$$\tilde{t}_{0,i} = \frac{t_{0,i} - \tau_{evento}}{\sigma_{t,datos}} \tag{5.1}$$

Para el escalar de la energía de cada cascada, se decidió categorizar la variable continua en 4 rangos energéticos, los cuales son $[(18{,}499, 18{,}875] < (18{,}875, 19{,}25] < (19{,}25, 19{,}625] < (19{,}625, 20{,}0]]$.

Modelo y entrenamientos

Para la arquitectura de los modelos se optó por usar capas convolucionales, con el fín de preservar la información entre detectores. Para una descripción mas detallada de cada arquitectura revisar **Anexo A**.

El único parámetro que se modificó fue la dimensión del vector latente, y se guardaron los modelos resultantes para cada dimensión, sin embargo se observó que para dimensiones bajas el modelo llegaba a un colapso de modos con más frecuencia (Table 5.2).

Generación de cascadas electromagnéticas

Cascadas electromagnéticas generadas con los modelos

Comparación con las cascadas de entrenamiento.

Comparación con 50 cascadas generadas usando CORSIKA.

Combinaciones lineales de vectores latentes

Resultados al promediar vectores latentes y hacer una combinación lineal de estos para después ser insertados en categorías energéticas diferentes.

```
from sys import argv

def reader(file):
  """
  This funciton returns a matrix M[i][j] of the input file.csv.
  """
  lines = open(file, 'r').readlines()
  a = []
  for i in range(0, len(lines)):
    a.append(lines[i].strip().split(','))
  return a

def reader_grammar(file):
  lines = open(file, 'r').readlines()
  a = []
  for i in range(0, len(lines)):
    a.append(lines[i].strip().split('->'))
  return a

def states(matr_of_file):
  """
  This function returns a list of the sates of a given transition matrix of a
    machine.
  The input must be a list of list of the form M[i][j].
  """
  a = []
  for i in range(1,len(matr_of_file)):
    a.append(matr_of_file[i][0])
  return a

def alphabet(matr_of_file):
  """
  This function returns a list of the alphabet of the machine M given by the
    input in matrix form M[i][j].
  """
  a = []
  for i in range(1, len(matr_of_file[0])-1):
    a.append(matr_of_file[0][i])
  return a

def number_list(l):
  """
  This function returns a list of number from 1 to len(l) given the list l.
  """
  a = []
  for x in range(1,len(l)+1):
    a.append(x)
  return a

def trans_func(matr_of_file, state, char):
  """
  This function returns the state that the machine goes to, given an state and
    a character of the alphabet.
  e.g. trans_funciton('state[0]', 'a') = 'state1'
  """
  return matr_of_file[dict(zip(states(matr_of_file), number_list(states(
    matr_of_file))))[state]][dict(zip(alphabet(matr_of_file), number_list(
    alphabet(matr_of_file))))[char]]

def init_state(matr_of_file):
  """
  This function returns te initial state of the machine M given by the input in
     matrix form M[i][j].
  """
  return matr_of_file[1][0]

def eof(matr_of_file):
```

```
62    """
63    This function returns a list of the acceptance states of the machine M given
        by the input in matrix form M[i][j].
64    """
65    a = []
66    for i in range(0,len(states(matr_of_file))):
67      if matr_of_file[i+1][len(matr_of_file[0])-1] == 'accept':
68        a.append(states(matr_of_file)[i])
69    return a
70
71  def isinalpha(a, alph):
72    """
73    This is a boolean function that tells if the character a is in the alphabet
        alph.
74    e.g. isinalpha('a', ['a', 'b']) = True
75    """
76    return a in alph
77
78  def machine(file, string):
79    """
80    This function implements the SFA give the input file in matrix form M[i][j]
        and given a string.
81    e.g. machine(reader('filename.csv'), 'hola') = No
82    """
83    if isinalpha(string[0], alphabet(file)):
84      s0 = trans_func(file, init_state(file), string[0])
85      i = 0
86      while i < len(string)-1:
87        n = s0
88        i += 1
89        if isinalpha(string[i], alphabet(file)):
90          try:
91            s0 = trans_func(file, n, string[i])
92          except:
93            return 'no'
94        else:
95          return 'no'
96      if (s0 in eof(file)) == True:
97        return 'yes'
98      else:
99        return 'no'
100   else:
101     return 'no'
102
103 scrpt, file, string = argv
104 print(machine(reader(file), string))
```

Capítulo 6

Conlusiones

En este capítulo se presentan las conclusiones de este trabajo.

6.1. Conclusiones puntuales obtenidas

I. Se logró implementar una arquitectura que genera cascadas electromagnéticas dado un escalar asociado a la cascada.

II. Se mostró que con la reducción de la dimensionalidad del espacio latente se siguieron obteniendo resultados confiables,a costa de obtener colapsos de modos en entrenamientos previos.

III. Por último, se mostró que el promedio de vectores latentes característicos tenía "tal" efecto en las cascadas generadas.

6.2. Trabajos futuros

Algunos de los trabajos futuros que pueden derivar directamente son:

a) Probar otros modelos generativos como InfoGAN, VAEGAN, BETA-VAE, modelos de difusión, etc. Para así hacer una comparación de cual es el modelo más efectivo para generar cascadas atmosféricas.

b) Desenredar los parámetros latentes del modelo para así obtener un control directo de parámetros asociados a la cascada y así generar observaciones confiables.

Apéndice A

Arquitecturas

A.1. Descripción de las capas

La información de las capas de generador se encuentra en la tabla A.1. Para un vector latente de dimensión 4 se tienen 1053329 parámetros de los cuales todos son parámetros de entrenamiento. Y para el modelo discriminador en la tabla A.2. Este modelo tiene un total de 1111365 parámetros.

CUADRO A.1: Capas del generador G_θ

LayerName	Size	Nparams
gen_net.0.0.weight	(8, 256, 3, 3)	18432
gen_net.0.1.weight	(256,)	256
gen_net.0.1.bias	(256,)	256
gen_net.0.1.running_mean	(256,)	256
gen_net.0.1.running_var	(256,)	256
gen_net.0.1.num_batches_tracked	()	1
gen_net.1.0.weight	(256, 256, 3, 3)	589824
gen_net.1.1.weight	(256,)	256
gen_net.1.1.bias	(256,)	256
gen_net.1.1.running_mean	(256,)	256
gen_net.1.1.running_var	(256,)	256
gen_net.1.1.num_batches_tracked	()	1
gen_net.2.0.weight	(256, 128, 3, 3)	294912
gen_net.2.1.weight	(128,)	128
gen_net.2.1.bias	(128,)	128
gen_net.2.1.running_mean	(128,)	128
gen_net.2.1.running_var	(128,)	128
gen_net.2.1.num_batches_tracked	()	1
gen_net.3.0.weight	(128, 128, 3, 3)	147456
gen_net.3.1.weight	(128,)	128
gen_net.3.1.bias	(128,)	128
gen_net.3.1.running_mean	(128,)	128
gen_net.3.1.running_var	(128,)	128
gen_net.3.1.num_batches_tracked	()	1
gen_net.4.weight	(128, 1, 3, 3)	1152
gen_net.4.bias	(1,)	1
embed.weight	(4, 4)	16

CUADRO A.2: Capas del Discriminador D_ϕ

LayerName	Size	Nparams
critic_net.0.0.weight	(64, 2, 3, 3)	1152
critic_net.0.1.weight	(64,)	64
critic_net.0.1.bias	(64,)	64
critic_net.1.0.weight	(128, 64, 3, 3)	73728
critic_net.1.1.weight	(128,)	128
critic_net.1.1.bias	(128,)	128
critic_net.2.0.weight	(128, 128, 3, 3)	147456
critic_net.2.1.weight	(128,)	128
critic_net.2.1.bias	(128,)	128
critic_net.3.0.weight	(256, 128, 3, 3)	294912
critic_net.3.1.weight	(256,)	256
critic_net.3.1.bias	(256,)	256
critic_net.4.0.weight	(256, 256, 3, 3)	589824
critic_net.4.1.weight	(256,)	256
critic_net.4.1.bias	(256,)	256
critic_net.5.weight	(1, 256, 3, 3)	2304
critic_net.5.bias	(1,)	1
embed.weight	(4, 81)	324

Bibliografía

[1] F Carminati y col. «Three dimensional Generative Adversarial Networks for fast simulation». En: *Journal of Physics: Conference Series* 1085.3 (2018), pág. 032016. ISSN: 1742-6596. DOI: 10.1088/1742-6596/1085/3/032016. URL: https://iopscience.iop.org/article/10.1088/1742-6596/1085/3/032016https://iopscience.iop.org/article/10.1088/1742-6596/1085/3/032016/meta.

[2] Federico Carminati y col. «Generative Adversarial Networks for fast simulation». En: *Journal of Physics: Conference Series* 1525.1 (2020), pág. 012064. ISSN: 1742-6596. DOI: 10.1088/1742-6596/1525/1/012064. URL: https://iopscience.iop.org/article/10.1088/1742-6596/1525/1/012064https://iopscience.iop.org/article/10.1088/1742-6596/1525/1/012064/meta.

[3] Peter Elmer, Mark Neubauer y Michael D. Sokoloff. «Strategic Plan for a Scientific Software Innovation Institute (S2I2) for High Energy Physics». En: (2017). arXiv: 1712.06592. URL: https://arxiv.org/abs/1712.06592v2.

[4] Ralph Engel, Dieter Heck y Tanguy Pierog. «Extensive Air Showers and Hadronic Interactions at High Energy». En: *http://dx.doi.org/10.1146/annurev.nucl.012809.104544* 61 (2011), págs. 467-489. ISSN: 01638998. DOI: 10.1146/ANNUREV.NUCL.012809.104544. URL: https://www.annualreviews.org/doi/abs/10.1146/annurev.nucl.012809.104544.

[5] M. Erdmann, J. Glombitza y D. Walz. «A deep learning-based reconstruction of cosmic ray-induced air showers». En: *Astroparticle Physics* 97 (2018), págs. 46-53. ISSN: 0927-6505. DOI: 10.1016/J.ASTROPARTPHYS.2017.10.006. URL: https://www.sciencedirect.com/science/article/pii/S0927650517302219.

[6] Martin Erdmann, Jonas Glombitza y Thorben Quast. «Precise Simulation of Electromagnetic Calorimeter Showers Using a Wasserstein Generative Adversarial Network». En: *Computing and Software for Big Science 2019 3:1* 3.1 (2019), págs. 1-13. ISSN: 2510-2044. DOI: 10.1007/S41781-018-0019-7. URL: https://link.springer.com/article/10.1007/s41781-018-0019-7.

[7] Martin Erdmann y col. «Generating and refining particle detector simulations using the Wasserstein distance in adversarial networks». En: *Computing and Software for Big Science* 2.1 (2018). arXiv: 1802.03325. URL: https://arxiv.org/abs/1802.03325v1.

[8] David Foster. *Generative deep learning. Teaching machines to paint, write, compose, and play*. O'Reilly Media, Inc., 2019., 2019. ISBN: 9781492041948.

[9] Jonas Glombitza. «Air-Shower Reconstruction at the Pierre Auger Observatory based on Deep Learning». En: *PoS* ICRC2019 (2020), pág. 270. DOI: 10.22323/1.358.0270. URL: http://www.auger.org/archive/authors_icrc_2019.htmlhttps://inspirehep.net/literature/1819455.

[10] Peter K.F. Grieder. *Extensive Air Showers. High Energy Phenomena and Astrophysical Aspects - A Tutorial Reference Manual and Data Book*. Springer Berlin, Heidelberg, 2010. ISBN: 978-3-540-76941-5. DOI: \url{https://doi.org/10.1007/978-3-540-76941-5}.

[11] Ishaan Gulrajani y col. «Improved Training of Wasserstein GANs». En: (2017). DOI: 10.48550/ARXIV.1704.00028. URL: https://arxiv.org/abs/1704.00028.

[12] Michela Paganini, Luke de Oliveira y Benjamin Nachman. «Accelerating Science with Generative Adversarial Networks: An Application to 3D Particle Showers in Multi-Layer Calorimeters». En: *Physical Review Letters* 120.4 (2017). DOI: 10.1103/PhysRevLett.120.042003. arXiv: 1705.02355v2. URL: http://arxiv.org/abs/1705.02355http://dx.doi.org/10.1103/PhysRevLett.120.042003.

[13] Michela Paganini, Luke de Oliveira y Benjamin Nachman. «CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks». En: *Physical Review D* 97.1 (2017). DOI: 10.1103/physrevd.97.014021. arXiv: 1712.10321. URL: https://arxiv.org/abs/1712.10321v1.

[14] Piergiorgio Picozza y V. Malvezzi. *Cosmic Rays under the Knee*. 2008. URL: https://www.researchgate.net/publication/265671393_Cosmic_Rays_under_the_Knee (visitado 12-05-2022).