UNIVERSIDAD DE MENDOZA

FACULTAD DE INGENIERÍA

INGENIERÍA EN INFORMÁTICA

Trabajo Final de Grado

HailCast: Deep Learning aplicado a la predicción de tormentas en imágenes de Radar

Caballero Federico Raúl

Asesor Especialista: Ing. Gabriel Federico Pulido

Profesor a cargo: Mgter. María Marta Arrieta Guevara

2025

Dedicatoria

Esta tesis va dedicada a mi familia, mi mujer Roxana y mi hija Olivia, que han sido la principal fuente de mi motivación e impulso para seguir adelante con esta carrera. Siempre me brindaron su apoyo, felicidad y paciencia para que esto fuera posible.

A mis padres, Raúl y Patricia, que desde pequeño me inculcaron la lectura, el estudio, el pensamiento crítico, la curiosidad y la creatividad, así como todo su apoyo en cada proyecto que emprendí a lo largo de mi vida.

A mis hermanos, Eliana y Gabriel, que también han sido fuente de inspiración, apoyo, motivación y alegría cuando más se necesitaron.

Agradecimientos

Deseo expresar mi más profundo agradecimiento al Dr. Ing. Horacio Gabriel Pessano, quien me facilitó desinteresadamente los datos históricos de radar, un insumo vital para el desarrollo de este trabajo. Agradezco también su guía, su apoyo y sus oportunos consejos en los momentos de incertidumbre.

A mis compañeros y amigos, Jesús, Facundo y Pablo. Gracias por el apoyo incondicional, la ayuda mutua y por todas las vivencias compartidas: desde las horas de estudio y los nervios de los exámenes, hasta los proyectos, las alegrías y los sueños que hemos construido juntos.

Un agradecimiento especial a la Lic. Mariela Asensio por su gestión como coordinadora, brindando siempre soluciones y disposición ante cualquier inquietud que surgió durante el cursado. Asimismo, a los profesores de la carrera, quienes con su experiencia han sido modelos a seguir para mi desarrollo, tanto en lo académico como en lo profesional.

Finalmente, a mi familia. Gracias a ellos tengo la motivación y la fuerza necesarias para progresar y alcanzar mis metas en la vida.

Resumen

La provincia de Mendoza enfrenta un riesgo crítico constante debido a tormentas convectivas severas que impactan su matriz productiva y seguridad civil. Históricamente, la mitigación y alerta temprana han dependido de sistemas de extrapolación lagrangiana (como TITAN), los cuales, al asumir linealidad y persistencia en el movimiento de las tormentas, resultan ineficaces para predecir procesos termodinámicos no lineales como la génesis explosiva o el decaimiento rápido de celdas.

Este Trabajo Final de Grado presenta **HailCast**, un sistema integral de predicción inmediata (*nowcasting*) basado en Inteligencia Artificial. La investigación propone un cambio de paradigma, transitando de la simulación física tradicional a un enfoque basado en datos (*Data-Driven*). Se desarrolló y entrenó una arquitectura de red neuronal profunda **ConvLSTM3D Enhanced**, diseñada para procesar secuencias volumétricas de radar (tensores 5D) y aprender la dinámica espaciotemporal de la atmósfera, superando las limitaciones de los métodos ópticos y de los modelos 2D convencionales.

Metodológicamente, el proyecto abordó desafíos complejos de ingeniería de datos e interoperabilidad, utilizando la suite científica **LROSE** para estandarizar archivos históricos propietarios (MDV) a formatos científicos modernos (NetCDF), garantizando la integridad física de las variables de reflectividad. A nivel operativo, se implementó una infraestructura **MLOps** robusta basada en contenedores (Docker) y microservicios, desacoplando el proceso de inferencia intensiva de la visualización. El resultado final se materializa en una interfaz web geoespacial de alto rendimiento (basada en React y MapLibre con aceleración WebGL), que permite a los operadores visualizar la evolución predicha de las tormentas con fluidez temporal.

Los resultados validan que el modelo es capaz de anticipar cambios morfológicos y estructurales en las tormentas que los sistemas actuales ignoran, demostrando que la integración de *Deep Learning* con una

ingeniería de software moderna constituye una evolución necesaria y viable para los sistemas de alerta temprana en la región.

Palabras clave: Nowcasting, Deep Learning, Radar Meteorológico, ConvLSTM, LROSE, Ingeniería de Software, MLOps, Mendoza.

Abstract

The province of Mendoza faces a critical and constant risk from severe convective storms that threaten both its productive matrix and civil safety. Historically, mitigation and early warning efforts have relied on Lagrangian extrapolation systems (such as TITAN). These systems, by assuming linearity and persistence in storm motion, prove ineffective at predicting non-linear thermodynamic processes, such as the explosive genesis or rapid decay of convective cells.

This Final Degree Project presents HailCast, a comprehensive immediate prediction (nowcasting) system based on Artificial Intelligence. The research proposes a paradigm shift, moving from traditional physical simulation to a Data-Driven approach. A ConvLSTM3D Enhanced deep neural network architecture was developed and trained to process volumetric radar sequences (5D tensors). This model learns the spatiotemporal dynamics of the atmosphere, overcoming the limitations of optical flow methods and standard 2D models.

Methodologically, the project addressed complex challenges in data engineering and interoperability by utilizing the LROSE scientific suite to standardize proprietary historical files (MDV) into modern scientific formats (NetCDF), ensuring the physical integrity of reflectivity variables. Operationally, a robust MLOps infrastructure was implemented using containers (Docker) and microservices, decoupling intensive inference processes from visualization. The final result is materialized in a high-performance geospatial web interface (based on React and MapLibre with WebGL acceleration), allowing operators to visualize the predicted evolution of storms with temporal fluidity.

The results validate that the model is capable of anticipating morphological and structural changes in storms that current systems miss, demonstrating that the integration of Deep Learning with modern

software engineering constitutes a necessary and viable evolution for early warning systems in the region.

Keywords: Nowcasting, Deep Learning, Weather Radar, ConvLSTM, LROSE, Software Engineering, MLOps, Mendoza.

INTRODUCCIÓN

La provincia de Mendoza, situada en el oeste de la República Argentina, presenta un escenario geográfico y climático de singular complejidad que condiciona su desarrollo socioeconómico. Caracterizada por un clima semidesértico y una fuerte dependencia de la irrigación artificial, la región ha consolidado una economía basada fundamentalmente en la agricultura intensiva, con la vitivinicultura y la fruticultura como pilares centrales. Sin embargo, esta matriz productiva se encuentra bajo una amenaza natural constante y severa: la inestabilidad atmosférica generada por la interacción de masas de aire con la Cordillera de los Andes, que deriva frecuentemente en el desarrollo de tormentas convectivas de gran magnitud.

La variabilidad y violencia de estos fenómenos, que incluyen desde lluvias torrenciales y ráfagas descendentes hasta la caída de granizo de gran tamaño, representan un riesgo crítico no sólo para la producción agrícola, que puede verse devastada en cuestión de minutos, sino también para la seguridad civil y la infraestructura urbana. Para gestionar este riesgo, la provincia ha desplegado históricamente una infraestructura tecnológica significativa, compuesta por una red de cuatro radares meteorológicos ubicados estratégicamente en los oasis productivos. Estos sensores remotos constituyen la primera línea de defensa, permitiendo el monitoreo en tiempo real de la atmósfera para coordinar tanto las alertas a la población como las operaciones de mitigación activa.

Durante décadas, la operación de esta red se ha sustentado en sistemas de software tradicionales, siendo TITAN (*Thunderstorm Identification, Tracking, Analysis, and Nowcasting*) el estándar de facto. Esta herramienta ha permitido digitalizar la atmósfera, identificando celdas de tormenta y proporcionando métricas vitales para la toma de decisiones. Sin embargo, la eficacia de cualquier sistema de gestión de riesgos, ya sea para emitir una alerta de protección civil o para guiar una intervención de mitigación, depende de un factor crítico: la capacidad

de anticipación. Es en este punto donde la tecnología vigente comienza a mostrar limitaciones estructurales que definen la problemática central de esta investigación.

Actualmente, la predicción de la evolución inmediata de una tormenta, conocida técnicamente como *nowcasting*, se resuelve en los sistemas operativos mediante métodos de extrapolación lagrangiana. Estos algoritmos operan bajo una simplificación física considerable: identifican una celda de tormenta como un "objeto" geométrico rígido y proyectan su posición futura basándose exclusivamente en su historia de movimiento lineal. Esta hipótesis de persistencia asume, erróneamente, que los sistemas convectivos son entidades estáticas que simplemente se desplazan por el espacio, ignorando su naturaleza termodinámica intrínseca.

La realidad atmosférica, sin embargo, es un sistema caótico y no lineal. Las tormentas no son objetos rígidos; son procesos dinámicos que nacen, crecen explosivamente, cambian de forma, se fusionan con otras celdas y finalmente se disipan. Los métodos de extrapolación lineal actuales son "ciegos" a estos cambios de intensidad. No pueden predecir el inicio súbito de una tormenta antes de que sea visible en el radar, ni pueden anticipar si una celda pequeña se convertirá en una supercelda severa o si, por el contrario, colapsará en los próximos minutos. Esta limitación técnica genera un cono de incertidumbre que degrada rápidamente la confiabilidad del pronóstico más allá de los treinta minutos, dejando a los operadores humanos con la difícil tarea de inferir mentalmente la evolución de sistemas complejos bajo condiciones de alta presión.

La brecha entre la complejidad del fenómeno físico y la simplicidad de las herramientas de predicción actuales resulta en ineficiencias operativas tangibles. En el contexto de los sistemas de alerta, esto puede traducirse en avisos tardíos o en falsas alarmas que erosionan la confianza pública. En el ámbito de la mitigación activa, la incapacidad para predecir la intensificación o el decaimiento de una celda puede

llevar a una asignación subóptima de recursos logísticos críticos, dirigiendo esfuerzos hacia tormentas que ya están en fase de disipación o ignorando núcleos en crecimiento explosivo.

Frente a este estancamiento metodológico, el avance reciente en el campo de la Inteligencia Artificial ofrece un nuevo paradigma para abordar el problema. El Aprendizaje Profundo (*Deep Learning*), y específicamente las arquitecturas diseñadas para el procesamiento de secuencias espaciotemporales, ha demostrado una capacidad superior para modelar fenómenos físicos complejos a partir de datos empíricos. A diferencia de los algoritmos deterministas tradicionales, las redes neuronales pueden aprender la física implícita de la atmósfera analizando grandes volúmenes de datos históricos, capturando patrones no lineales de crecimiento y decaimiento que escapan a la extrapolación matemática simple.

Este trabajo de tesis propone el desarrollo de un sistema integral de predicción a corto plazo que trascienda las limitaciones de los métodos lagrangianos vigentes. Aprovechando el extenso archivo de datos históricos generados por la red de radares de Mendoza, se plantea el entrenamiento de un modelo de *Deep Learning* basado en la arquitectura ConvLSTM (*Convolutional Long Short-Term Memory*). Este enfoque no trata a la tormenta como un objeto geométrico, sino que aborda el problema como una traducción de secuencias de video, aprendiendo simultáneamente las características espaciales de la estructura de la nube y la dinámica temporal de su evolución.

El proyecto, denominado *Hailcast*, no se limita a la experimentación teórica del modelo, sino que abarca la ingeniería completa de un sistema *end-to-end*. Esto implica resolver los desafíos de interoperabilidad entre los formatos meteorológicos heredados y los tensores científicos modernos, diseñar una estrategia de entrenamiento robusta que mitigue los problemas de difuminado inherentes a las predicciones estocásticas, y desarrollar una interfaz de visualización geoespacial moderna que permita validar la utilidad operativa de las

predicciones.

En definitiva, esta investigación busca demostrar que la integración de la Inteligencia Artificial en el flujo de trabajo meteorológico no es solo una actualización tecnológica, sino una evolución necesaria. Al dotar al sistema de la capacidad de "ver" el futuro inmediato con mayor fidelidad estructural y dinámica, se pretende transformar el paradigma de operación reactiva actual en uno verdaderamente predictivo, optimizando así los recursos destinados a la protección de la economía regional y la seguridad de la población frente a la amenaza constante del clima severo.

Parte 1
Marco Teórico

El desarrollo de un sistema de predicción meteorológica basado en inteligencia artificial no puede entenderse como una mera implementación de software, sino como la convergencia de tres disciplinas complejas: la física de la atmósfera, la ciencia de datos y la ingeniería de software. Este capítulo establece los fundamentos epistemológicos y técnicos que sustentan dicha integración, proporcionando el andamiaje teórico necesario para validar la solución propuesta.

En primera instancia, se aborda el estado del arte en la predicción inmediata o *nowcasting*, analizando la evolución histórica desde los modelos numéricos tradicionales hasta las modernas técnicas de aprendizaje profundo. Este recorrido permite identificar las limitaciones estructurales de los enfoques deterministas vigentes y justifica la adopción de un paradigma basado en datos (*Data-Driven*) como la respuesta más adecuada ante la naturaleza caótica y no lineal de los fenómenos convectivos.

Posteriormente, se profundiza en los principios físicos de la meteorología de radar. Se examina la naturaleza de la variable predictiva —la reflectividad— y su relación con la microfísica de la precipitación, así como las implicaciones matemáticas de trabajar con datos volumétricos en escalas logarítmicas. Comprender la estructura de la información de entrada es un requisito indispensable para diseñar arquitecturas neuronales que no solo procesen píxeles, sino que interpreten señales físicas reales.

El núcleo del marco teórico se centra en la arquitectura de redes neuronales recurrentes convolucionales (ConvLSTM). Se desglosa la formulación matemática que permite a estas redes aprender simultáneamente representaciones espaciales y dependencias temporales, superando las capacidades de las redes convencionales. Asimismo, se discuten los desafíos inherentes al entrenamiento de modelos generativos de video, como el problema del difuminado (*blurriness*) en predicciones a largo plazo, y las estrategias de diseño,

como las conexiones residuales y las funciones de pérdida híbridas, que permiten mitigar estos efectos.

Finalmente, se exponen los fundamentos de la ingeniería de datos meteorológicos. Se analiza la problemática de la interoperabilidad entre los formatos heredados de la industria operativa y los estándares científicos modernos, detallando el rol crítico de la suite LROSE en la preservación de la integridad numérica durante la conversión de datos. Este marco conceptual no solo justifica las decisiones de diseño del sistema *Hailcast*, sino que establece la validez científica de la metodología empleada.

1. Antecedentes y Estado del Arte

La predicción meteorológica inmediata, conocida técnicamente como *Nowcasting*, constituye una de las fronteras más desafiantes y críticas de las ciencias atmosféricas contemporáneas. Se define generalmente como la descripción detallada del estado actual de la atmósfera y la predicción de su evolución en un horizonte temporal de muy corto plazo, típicamente de 0 a 6 horas, con una relevancia operativa crítica en el intervalo de 0 a 2 horas.

En términos de objetivos del Nowcasting, Shi et al (2015) plantean lo siguiente: **"The goal of precipitation nowcasting is to predict the future rainfall intensity in a local region over a relatively short period of time."** A lo que se puede agregar como objetivos ligados a esto: la salvaguarda de vidas humanas, la protección de infraestructuras críticas (como redes eléctricas y sistemas de transporte) y la mitigación de daños en la agricultura que se apoyan fuertemente de la emisión de alertas tempranas precisas y oportunas.

Históricamente, el nowcasting ha dependido de una dicotomía metodológica: la Predicción Numérica del Tiempo (NWP) y la extrapolación basada en sensores remotos. Los modelos NWP, que resuelven numéricamente las ecuaciones primitivas de la dinámica de

fluidos y la termodinámica atmosférica (ecuaciones de Navier-Stokes, conservación de masa y energía), son insuperables en escalas sinópticas y de mesoescala a plazos medios. Sin embargo, sufren del problema de "spin-up", un periodo inicial de latencia e incoherencia dinámica necesario para que el modelo asimile los datos iniciales y equilibre sus campos de viento y masa. Durante este periodo, que puede durar de 1 a 3 horas, la producción de precipitación del modelo es a menudo espuria o inexistente, degradando su utilidad para el pronóstico inmediato de tormentas de rápida evolución.

Por otro lado, los métodos de extrapolación lineal, como el Flujo Óptico o algoritmos orientados a objetos como TITAN (*Thunderstorm Identification, Tracking, Analysis, and Nowcasting*), han llenado este vacío operativo. Estos sistemas asumen la conservación lagrangiana del movimiento y la intensidad de los ecos de radar. Si bien son computacionalmente eficientes, fallan estrepitosamente al intentar modelar procesos no lineales termodinámicos, como la iniciación convectiva (el nacimiento explosivo de nuevas tormentas) o el decaimiento rápido de celdas existentes debido al arrastre de aire seco o la estabilización de la capa límite. La atmósfera es un sistema caótico donde la linealidad es la excepción, no la norma.

Existe, por tanto, una desconexión entre el avance científico y la implementación práctica. Como observan Cuomo & Chandrasekar (2021), ***"there are many different types of deployed nowcasting systems, but none of them based on machine learning, even though it has been an active area of research in the last few years"*.**

Para cerrar esta brecha, el advenimiento del Aprendizaje Profundo (*Deep Learning*), y específicamente el desarrollo de arquitecturas neuronales capaces de manejar datos tensoriales de alta dimensionalidad (espacio y tiempo), ha propuesto un cambio de paradigma radical. **Al adoptar la formalización de Shi et al. (2015), quienes plantean este desafío como un *"spatiotemporal sequence forecasting problem in which both the input and the prediction target***

***are spatiotemporal sequences"*, se deja de lado la simulación física directa** (que requiere inmensos recursos computacionales para resolver microfísica). Bajo esta perspectiva *Data-Driven*, Cuomo & Chandrasekar (2021) definen el *nowcasting* no como física de fluidos, sino como ***"a translation of sequence to sequence of images"*, donde el desafío es predecir *"the distribution, intensity, and appearance of future sequences... based on historical observations"*.**

Para resolver este problema de traducción de imágenes, modelos como las Redes Neuronales Recurrentes Convolucionales (ConvLSTM) han demostrado una capacidad superior para capturar la dinámica no lineal oculta en los grandes volúmenes de datos históricos de radar.

El presente marco teórico fundamenta el desarrollo de un sistema *end-to-end* basado en una arquitectura **ConvLSTM3D_Enhanced**. A diferencia de las implementaciones académicas aisladas que a menudo operan sobre datasets simplificados, este sistema integra una cadena de valor completa y robusta: desde la ingesta de datos en formatos meteorológicos heredados (MDV) y su conversión a estándares científicos (NetCDF) mediante la suite LROSE, hasta el entrenamiento de modelos de Deep Learning en entornos contenerizados (Docker) y la visualización geoespacial en tiempo real (Next.js, MapLibre). Este documento analiza exhaustivamente los principios físicos, matemáticos y de ingeniería de software que sustentan dicha integración, estableciendo la validez epistemológica de la solución propuesta.

2. Meteorología de Radar: Fundamentos Físicos del Problema de Predicción

El insumo fundamental para el sistema propuesto es la información proveniente del radar meteorológico. Para comprender las decisiones de diseño en la arquitectura del modelo de Deep Learning (como la normalización de datos o el uso de convoluciones 3D) y las estrategias de preprocesamiento, es imperativo establecer primero la naturaleza física de la variable predictiva: la reflectividad, y cómo esta se relaciona

con la microfísica de la precipitación.

2.1 Principios de Teledetección Activa y la Ecuación del Radar

El radar meteorológico es un sensor activo que opera emitiendo pulsos electromagnéticos de alta potencia y corta duración en longitudes de onda de microondas, típicamente en las bandas S (10 cm), C (5 cm) o X (3 cm). Cuando esta energía electromagnética viaja a través de la atmósfera e intercepta hidrometeoros (gotas de lluvia, granizo, graupel, nieve), ocurre un proceso de dispersión (*scattering*). Una fracción de la energía incidente es absorbida, otra es dispersada en todas direcciones, y una pequeña fracción es retrodispersada (*backscattered*) directamente hacia la antena del radar.

La potencia media recibida ($\bar{P}_r$) por el radar de un volumen de muestra distribuido se describe mediante la Ecuación del Radar Meteorológico, que establece las dependencias físicas fundamentales que el sistema de datos debe procesar:

$$\bar{P}_r = \frac{C \cdot P_t \cdot G^2 \cdot \lambda^2 \cdot \theta \cdot \phi \cdot h \cdot |K|^2 \cdot Z}{R^2}$$

Donde:

- C: Constante del radar que agrupa eficiencias del sistema y características del hardware.
- P_t: Potencia pico transmitida.
- G: Ganancia de la antena.
- λ: Longitud de onda del pulso.
- θ, ϕ: Anchos del haz horizontal y vertical (resolución angular).
- h: Longitud del pulso espacial (resolución en rango).
- $|K|^2$: Factor dieléctrico complejo del objetivo, que depende de la fase del agua (aproximadamente 0.93 para agua líquida y 0.197 para hielo).
- R: Distancia al objetivo (la atenuación por divergencia del haz es

proporcional a $1/R^2$).

- Z: Factor de reflectividad del radar.

Para los propósitos del nowcasting y del entrenamiento del modelo **ConvLSTM3D_Enhanced**, la variable crítica es Z. Bajo la aproximación de Rayleigh (válida cuando el diámetro de la partícula D es mucho menor que la longitud de onda, $D \ll \lambda$), Z se define como la suma de la sexta potencia de los diámetros de las partículas por unidad de volumen:

$$Z = \int_0^\infty N(D) D^6 dD$$

Donde $N(D)$ es la distribución del tamaño de las gotas (DSD). Esta relación de sexta potencia tiene implicaciones profundas para el aprendizaje automático: la reflectividad es hipersensible al tamaño de las partículas. Unas pocas piedras de granizo grandes pueden producir el mismo retorno de energía que millones de gotas de lluvia pequeñas, pero las implicaciones meteorológicas (daño en superficie vs. lluvia benéfica) son drásticamente diferentes.

2.2 La Escala Logarítmica dBZ y su Impacto en el Preprocesamiento

Debido al amplio rango dinámico de Z, que puede variar desde $10^{-3} mm^6/m^3$ (niebla) hasta $10^7 mm^6/m^3$ (granizo gigante), esta variable se expresa convencionalmente en una escala logarítmica conocida como decibelios de reflectividad (dBZ):

$$dBZ = 10 \log_{10} \left(\frac{Z}{1 mm^6/m^3} \right)$$

Implicaciones Teóricas para el Modelo de Deep Learning:

1. **No Linealidad Extrema:** La entrada al modelo no es una imagen en escala de grises convencional (0-255) donde la luminosidad escala

linealmente. En los datos de radar, un aumento de 10 a 20 dBZ implica un incremento de 10 veces en la reflectividad lineal Z, mientras que un aumento de 50 a 60 dBZ implica el mismo factor multiplicativo pero sobre una base energética inmensamente mayor. El modelo **ConvLSTM3D_Enhanced** debe aprender a interpretar estos gradientes no lineales. Físicamente, la diferencia entre 50 y 60 dBZ es mucho más crítica para la severidad de la tormenta (indica granizo severo) que la diferencia entre 10 y 20 dBZ (lluvia ligera a moderada).

2. **Rango Dinámico y Normalización:** El sistema implementado normaliza los valores de reflectividad en el rango de **-29 a 65 dBZ**. Este rango no es arbitrario. -29 dBZ representa el piso de ruido o sensibilidad mínima de muchos radares operativos (señales muy débiles, insectos o aire claro), mientras que 65 dBZ representa el límite superior físico común para tormentas convectivas severas con granizo. Valores superiores suelen ser artefactos o granizo extremo contaminado. La normalización mapea este rango físico al intervalo o [-1, 1] para estabilizar los gradientes durante el entrenamiento de la red neuronal (backpropagation), evitando la saturación de funciones de activación como la Sigmoid o la Tanh usadas en las compuertas LSTM.

2.3 Estructura Volumétrica y Datos 5D

A diferencia de las imágenes satelitales que observan principalmente el tope de las nubes, el radar realiza escaneos volumétricos (VCP - *Volume Coverage Patterns*), rotando la antena en múltiples ángulos de elevación para muestrear la estructura tridimensional de la atmósfera.

El sistema descrito procesa **Volúmenes 5D**: ($Batch$, $Time$, $Channels$, $Height$, $Width$). Esta decisión de diseño es teóricamente superior a los enfoques que utilizan productos derivados 2D como CMAX (*Column Max*) o PPI (*Plan Position Indicator*) por varias razones fundamentales:

- **Detección de Severidad en Altura:** La severidad de una tormenta no

solo depende de la lluvia en superficie. La presencia de altos valores de reflectividad (ej. >50 dBZ) suspendidos a gran altura (por encima de la isoterma de 0°C) es un indicador físico directo de una fuerte corriente ascendente (*updraft*) que sostiene grandes hidrometeoros (granizo) en formación. Un modelo 2D que solo ve la proyección máxima pierde esta información vertical crítica.

- **Dinámica de Crecimiento y Colapso:** Una tormenta que está colapsando a menudo muestra un descenso rápido del núcleo de reflectividad ("descending core"). Al utilizar convoluciones 3D o capas ConvLSTM que preservan la profundidad del tensor, el modelo **ConvLSTM3D_Enhanced** puede aprender explícitamente estas correlaciones verticales y predecir ráfagas descendentes (*microbursts*) o intensificación de la lluvia, superando las capacidades de modelos planos.

2.4 El Problema del Nowcasting y las Limitaciones de TITAN

Antes de la era del aprendizaje profundo, el estándar de oro para el nowcasting de tormentas era el sistema **TITAN** (*Thunderstorm Identification, Tracking, Analysis, and Nowcasting*), desarrollado por Dixon y Wiener (1993) en NCAR. TITAN opera bajo un paradigma lagrangiano orientado a objetos.

1. **Identificación:** Umbraliza los datos de radar volumétricos (ej. >35 dBZ) y agrupa los píxeles contiguos en "objetos" geométricos (poliedros o elipsoides).
2. **Seguimiento:** Asocia estos objetos entre tiempos consecutivos (t y $t+1$) utilizando optimización combinatoria (Algoritmo Húngaro) para minimizar una función de costo basada en la distancia y el cambio de volumen.
3. **Predicción:** Extrapola la posición futura del centroide basándose en la historia de movimiento lineal (persistencia).

Aunque robusto y computacionalmente ligero, el marco teórico de

TITAN presenta limitaciones insalvables para el nowcasting moderno de alta precisión:

- **Rigidez Morfológica:** TITAN simplifica la compleja morfología de una tormenta a un elipsoide. No puede predecir cambios de forma no lineales, como la evolución de una supercelda a una línea de turbonada (*bow echo*), lo cual es vital para advertir sobre vientos dañinos.

  Esta limitación geométrica impide modelar la interacción entre celdas. Como señalan Zheng et al. (2022), en procesos complejos de evolución de tormentas **"like merging, splitting, growth and decay, traditional methods are difficult to predict accurately"**, fallando precisamente donde la dinámica se vuelve caótica.

- **Incapacidad de Predicción de Génesis:** TITAN solo puede rastrear lo que ya existe y supera el umbral de detección. Es "ciego" a los procesos precursores. Una red neuronal profunda, al analizar todo el campo de reflectividad (incluyendo valores bajos < 30 dBZ que TITAN descarta), tiene el potencial teórico de detectar patrones de convergencia o crecimiento inicial antes de que se forme un núcleo convectivo maduro.
- **Decaimiento de la Habilidad:** La hipótesis de persistencia (que la tormenta seguirá moviéndose igual y con la misma intensidad) viola la termodinámica atmosférica. Las tormentas nacen, crecen, maduran y mueren. La precisión de la extrapolación lineal decae significativamente más allá de los 30 minutos.

El sistema propuesto busca superar estas barreras mediante el uso de redes neuronales que aprenden la dinámica evolutiva, no solo la advectiva.

Esta capacidad es crítica para resolver la "ceguera" de los métodos tradicionales. Tal como demuestran Su et al. (2020), al procesar características espaciales profundas, el algoritmo logra *"forecast*

the generation and dissipation trends of convective cells to some extent", una funcionalidad imposible bajo los supuestos de persistencia de TITAN.

Más recientemente, Zhang et al. (2025) confirman que la integración de módulos de atención permite capturar características dinámicas integrales del sistema de precipitación, tales como **"movement speed, precipitation area, intensity, propagation, formation, and dissipation"**, validando la superioridad del enfoque *Deep Learning* para representar la complejidad atmosférica completa.

3. Redes Neuronales en Meteorología: De la Extrapolación Óptica al Aprendizaje Profundo

La transición de los métodos tradicionales al Deep Learning en meteorología no es meramente un cambio de herramientas, sino una evolución epistemológica: de imponer modelos físicos simplificados (ecuaciones de advección lineal) a permitir que los datos revelen la física subyacente a través de la optimización de funciones universales.

3.1 El Paradigma del Flujo Óptico (Optical Flow)

El Flujo Óptico ha sido la técnica puente entre TITAN y el Deep Learning. Métodos como Lucas-Kanade o el algoritmo **ROVER** (*Real-time Optical flow by Variational methods for Echoes of Radar*) tratan las imágenes de radar como una secuencia de fluidos visuales. Calculan un campo de vectores de movimiento que describe cómo se desplazan los píxeles de reflectividad de un cuadro al siguiente. La ecuación fundamental del flujo óptico asume que la intensidad I de un objeto se conserva durante su movimiento:

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

Donde (u, v) es el vector de velocidad. Si bien este enfoque supera la rigidez de objetos de TITAN, su talón de Aquiles sigue siendo la simplificación dinámica. Zheng et al. (2022) son categóricos al afirmar que **“the optical flow extrapolates the echo motion as linear, which is not realistic and leads to poorer forecasts”**.

Esta falta de realismo radica en el término fuente/sumidero ($\frac{dI}{dt} \neq 0$). En la atmósfera real, la intensidad de la lluvia *no* se conserva; el vapor de agua se condensa (fuente) y la lluvia cae (sumidero). Por lo tanto, aunque útil para lo inmediato, la literatura confirma (Su et al., 2020) que **“the optical flow method is insufficient over a short valid forecasting period (0–6 h)”**, justificando la transición hacia modelos no lineales.

3.2 La Llegada del Deep Learning: CNN y RNN

La aplicación de redes neuronales comenzó con intentos de usar **CNNs** (Redes Neuronales Convolucionales) estándar. Las CNN son excepcionales extrayendo características espaciales (texturas de nubes, gradientes de tormenta), pero carecen de memoria temporal intrínseca. Una CNN ve una "foto" de la tormenta, pero no sabe si es un fotograma de una película donde la tormenta crece o decrece.

Para abordar la temporalidad, se introdujeron las **RNNs** (Redes Neuronales Recurrentes) y su variante avanzada, la **LSTM** (*Long Short-Term Memory*). La LSTM soluciona el problema del desvanecimiento del gradiente, permitiendo aprender dependencias temporales a largo plazo. Sin embargo, la LSTM estándar ("Fully Connected LSTM" o FC-LSTM) requiere que los datos de entrada se aplanen en vectores 1D.

$$\mathcal{H}_t = f(W \cdot \mathrm{vec}(\mathcal{X}_t) + U \cdot \mathcal{H}_{t-1})$$

Esta operación de "aplanado" es destructiva para datos meteorológicos: destruye la topología espacial. Al convertir una imagen de radar en un vector largo, se pierde la noción de vecindad local (que el píxel (x, y)

está al lado del $(x+1, y)$). Dado que los fenómenos meteorológicos son locales y se rigen por ecuaciones diferenciales parciales que dependen de gradientes espaciales vecinos, la FC-LSTM es teóricamente inadecuada para el nowcasting espacial.

En coincidencia con este diagnóstico, Shi et al (2015) observan que, **"although the FC-LSTM layer has proven powerful for handling temporal correlation, it contains too much redundancy for spatial data"**. En respuesta, los autores introducen **"an extension of FC-LSTM which has convolutional structures in both the input-to-state and state-to-state transitions"**.

Esta limitación estructural fue el catalizador para el desarrollo de la arquitectura **ConvLSTM**, el núcleo del sistema desarrollado en este proyecto.

4. Arquitectura ConvLSTM: Superioridad Matemática y Conceptual

La arquitectura **ConvLSTM3D_Enhanced** implementada representa el estado del arte en modelado predictivo. En términos formales recientes, Ghosh et al. (2025) definen la esencia de este diseño señalando que **"the ConvLSTM model integrates convolutional operations with recurrent long short-term memory (LSTM) units, enabling simultaneous learning of spatial features and temporal dependencies within precipitation fields"**.

Esta capacidad de aprendizaje simultáneo es la que fundamenta su ventaja sobre los predecesores. El trabajo seminal de Shi et al. (2015) validó dos hitos críticos:

1. Superaron las limitaciones de las redes recurrentes planas, demostrando que *"ConvLSTM is better than FC-LSTM in handling spatiotemporal correlations"*.

2. Probaron su superioridad frente a los algoritmos de flujo óptico tradicionales, estableciendo que *“ConvLSTM performs better than ROVER for precipitation nowcasting”*.

Esta superioridad operativa no es accidental, sino el resultado de un diseño estratégico. Como detallan Su et al. (2020), el objetivo fue **“establish an end-to-end deep learning network”**. Gracias a esta naturaleza híbrida, el modelo logra **“deeply extract high-order features of radar echoes such as structural texture, spatial correlation, and temporal evolution compared with the traditional algorithm”**, superando las capacidades de los métodos puramente ópticos.

4.1 Formulación Matemática de la Celda ConvLSTM

La innovación crucial de la ConvLSTM es el reemplazo de las multiplicaciones de matrices (producto punto) en las transiciones de estado de la LSTM tradicional por operaciones de **convolución**. Esto permite que tanto los datos de entrada ($\mathcal{X}$) como los estados ocultos ($\mathcal{H}$) y las celdas de memoria ($\mathcal{C}$) mantengan su estructura tensorial (Alto $\times$ Ancho $\times$ Canales) a través del flujo de la red.

Las ecuaciones dinámicas que gobiernan la ConvLSTMCell personalizada del sistema son:

$$
\begin{aligned}
i_t &= \sigma(W_{xi} * \mathcal{X}_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \circ \mathcal{C}_{t-1} + b_i) \\
f_t &= \sigma(W_{xf} * \mathcal{X}_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \circ \mathcal{C}_{t-1} + b_f) \\
\mathcal{C}_t &= f_t \circ \mathcal{C}_{t-1} + i_t \circ \tanh(W_{xc} * \mathcal{X}_t + W_{hc} * \mathcal{H}_{t-1} + b_c) \\
o_t &= \sigma(W_{xo} * \mathcal{X}_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \circ \mathcal{C}_t + b_o) \\
\mathcal{H}_t &= o_t \circ \tanh(\mathcal{C}_t)
\end{aligned}
$$

Donde $*$ denota la operación de convolución y $\circ$ el producto de Hadamard (elemento a elemento).

Interpretación Teórica de los Componentes:

1. **Operador de Convolución ($*$):** Al utilizar kernels de 3x3 (como se

especifica en la configuración del modelo), la red determina el estado futuro de cada celda basándose en sus vecinos. Esta decisión de diseño es crítica, ya que, según Shi et al. (2015), **"making the size of state-to-state convolutional kernel bigger than 1 is essential for capturing the spatiotemporal motion patterns"**.

Sin esta vecindad espacial (un kernel > 1), el modelo no podría capturar movimientos de traslación ni expansión de la tormenta.

2. **Compuerta de Olvido (f_t):** Siguiendo la definición de Shi et al. (2015), mediante este mecanismo **"the past cell status c_{t-1} could be 'forgotten' in this process if the forget gate f_t is on"**.

   Esta capacidad es clave para manejar la no linealidad. Por ejemplo, si una celda convectiva entra en fase de disipación, la compuerta suprime la señal de memoria asociada, permitiendo predecir su desaparición. Esto supera la limitación del flujo óptico descrita por Su et al. (2020), ya que dicho enfoque **"can only track echo characteristics that already exist [...] and cannot predict either the generation or dissipation of echoes"**.

3. **Estado de Memoria ($\mathcal{C}_t$):** Este componente es fundamental para la continuidad temporal. Como definen Shi et al. (2015), **"the major innovation of LSTM is its memory cell c_t which essentially acts as an accumulator of the state information"**.

   Gracias a esta acumulación, el sistema puede diferenciar entre una celda de 50 dBZ que está creciendo (fase madura temprana) y una de 50 dBZ que está colapsando. Al basarse en la historia temporal codificada, y tal como validan Su et al. (2020), el algoritmo **"performs well in convection nowcasting based on the training and learning of evolutionary patterns of the radar echoes using historical data"**.

4.2 Arquitectura "Enhanced" y Profundidad 3D

El modelo implementado se describe como **ConvLSTM3D_Enhanced**. Esta distinción es vital. La ConvLSTM original de Shi et al. operaba sobre secuencias 2D. Al procesar **Volúmenes 5D**, la red extiende su capacidad de aprendizaje a la dimensión vertical (Z).

Para manejar esta complejidad dimensional, se emplea una estructura **Encoder-Decoder apilada** (3 capas con 128 canales ocultos).El funcionamiento de esta topología se basa en el principio de compresión y reconstrucción descrito por Shi et al. (2015): **“The encoding LSTM compresses the whole input sequence into a hidden state tensor and the forecasting LSTM unfolds this hidden state to give the final prediction”**.

Estabilidad y Transferencia de Información: La profundidad de la red conlleva riesgos de desvanecimiento del gradiente. El componente "Enhanced" mitiga esto mediante conexiones residuales (*Skip Connections*). Esta decisión ofrece una doble ventaja validada por la literatura reciente (2025):

1. **Estabilidad del Entrenamiento:** Como señalan Liu et al. (2025), estas conexiones previenen la pérdida de información y **“stabilize training”** en tareas de reconstrucción.
2. **Precisión Multiescala:** Complementariamente, Zhang et al. (2025) destacan que las *skip connections* permiten al modelo **“leverage multi-scale radar echo features, facilitating efficient spatial information transfer and improving the model’s ability to extract high-level features while retaining critical details”**.

Eficiencia y Normalización: Dado el alto costo computacional de los tensores 5D, se adoptaron técnicas de normalización avanzadas para reducir la dependencia de grandes lotes de datos. Siguiendo la propuesta de Liu et al. (2025), la incorporación de normas de grupo (GroupNorm) permite **“reduce feature scale discrepancies, thereby making model training less dependent on batch size, lowering**

hardware requirements, and enhancing the model's ability to capture inter-feature relationships".

Esta estrategia es fundamental para justificar el entrenamiento con un *batch size* reducido (detallado en la sección 4.4) sin sacrificar la estabilidad estadística del modelo.

Finalmente, la capa de salida Conv3D con activación **Sigmoid** es la interfaz final que transforma el estado latente abstracto de vuelta a un espacio de probabilidad física normalizado. La función Sigmoid comprime la salida al rango $(0, 1)$, lo cual es consistente con la normalización aplicada a los datos de reflectividad (mapeados de -29...65 dBZ a 0...1).

4.3 Desafíos Inherentes: El Difuminado (Blurriness)

Si bien los algoritmos de extrapolación basados en *Deep Learning* han demostrado una efectividad superior, no están exentos de limitaciones estructurales críticas. Chen et al. (2023) ofrecen un diagnóstico preciso: aunque eficientes, estos modelos enfrentan desafíos como **"cumulative error spreading, imprecise representation of sparsely distributed echoes, and inaccurate description of non-stationary motion processes"**.

El síntoma más visible de estos problemas es el "difuminado" de las predicciones. Un desafío bien documentado es que funciones de pérdida como el Error Cuadrático Medio (MSE) asumen una distribución gaussiana unimodal, incapaz de capturar la incertidumbre estocástica del futuro (Cuomo, 2020).

Sin embargo, el problema no es solo la función de pérdida, sino también la arquitectura misma. Paradójicamente, Zhang et al. (2025) advierten que **"as a model's complexity increases, the prediction outputs tend to become blurred and smoothed, which complicates the precise capture of detailed changes in precipitation fields"**. Esto sugiere que la profundidad de la red, si no se controla, puede actuar como un filtro de

suavizado no deseado.

Este fenómeno tiene consecuencias operativas directas, especialmente para la detección de eventos extremos. Existe un consenso reciente, ratificado por Zhang et al. (2025), de que **“in most deep learning-based models, accurately predicting heavy precipitation remains a challenging task”**.

En la misma línea, Zheng et al. (2024) proponen mecanismos de atención para **“alleviate the inherent issue observed in existing models, which is the decrease in forecast accuracy for high radar echo regions with extended extrapolation time”**.

El sistema propuesto aborda esta encrucijada mediante una estrategia híbrida. Para contrarrestar la tendencia al suavizado descrita por Zhang, se utiliza una arquitectura profunda (128 canales) compensada por un post-procesamiento agresivo. Al aplicar un umbral físico (*Thresholding* > 30 dBZ), el sistema descarta la incertidumbre de baja probabilidad (la "borrosidad" de fondo) y recupera la estructura morfológica "dura" de la tormenta convectiva, haciéndola operativa para los meteorólogos.

5. Ingeniería de Datos Meteorológicos: Interoperabilidad y LROSE

La eficacia de un modelo de IA es directamente proporcional a la calidad y consistencia de los datos de entrenamiento. En el dominio meteorológico, existe una brecha tecnológica significativa entre los sistemas de radar operativos (legados) y los ecosistemas de ciencia de datos modernos (Python/PyTorch). La ingeniería de datos de este proyecto, centrada en la suite **LROSE** (*Lidar Radar Open Software Environment*), actúa como el puente teórico y práctico indispensable.

5.1 La Importancia Crítica de los Formatos: MDV vs. NetCDF

El sistema está diseñado para ingerir archivos **MDV** (*Meteorological*

Data Volume). Este formato, desarrollado por NCAR para el software TITAN, ha sido el estándar operativo durante décadas. Está optimizado para almacenar grillas cartesianas 3D con compresión RLE (*Run-Length Encoding*), lo que lo hace muy eficiente en espacio. Sin embargo, MDV es un formato binario propietario, opaco e inaccesible para las bibliotecas modernas de Deep Learning como PyTorch, que esperan tensores numéricos estandarizados.

La solución implementada utiliza **NetCDF** (*Network Common Data Form*), el estándar *de facto* en ciencias de la tierra. NetCDF es autodescriptivo y multidimensional. La conversión no es trivial; la investigación destaca problemas críticos con los metadatos scale_factor y add_offset. Los datos de radar en MDV a menudo se almacenan como enteros de 8 bits para ahorrar espacio, y deben desempaquetarse usando la ecuación lineal $y = x \cdot scale + offset$ para recuperar los valores físicos de punto flotante (dBZ).

El pipeline implementa una doble traducción:

1. **Ingesta (Mdv2NetCDF):** Utiliza la herramienta de LROSE para "traducir" el binario MDV a NetCDF, resolviendo correctamente la física de la compresión y las proyecciones geográficas. Sin esto, el modelo entrenaría sobre números enteros sin significado físico ("garbage in, garbage out").
2. **Salida (NcGeneric2Mdv):** Para que la IA sea útil, sus predicciones deben visualizarse en las herramientas que los meteorólogos ya usan (como TITAN/Hawk). Esta herramienta reconvierte las predicciones del modelo (NetCDF) al formato nativo MDV, cerrando el ciclo operativo.

5.2 Pre-procesamiento: Normalización y Clipping

Las redes neuronales son matemáticamente sensibles a la escala de los datos. Entrenar con valores crudos causaría inestabilidad en los gradientes. Ghosh et al. (2025) fundamentan la necesidad de este paso,

señalando que las características de entrada deben someterse a un escalado robusto para **“mitigate outlier effects... and stabilize variance across the wide range of rainfall intensities”**.

Siguiendo este principio, el sistema implementa una estrategia fundamentada en la física:

- **Naturaleza Logarítmica (dBZ):** Se trabaja con reflectividad en dBZ, lo cual es intrínsecamente una transformación logarítmica de la energía (Z). Esto se alinea con la práctica descrita por Ghosh et al. (2025) de aplicar transformaciones logarítmicas para manejar rangos dinámicos amplios.
- **Clipping y Rango Físico:** Se define un rango de interés de **-29 a 65 dBZ**. Valores por debajo de -29 dBZ se consideran ruido de "aire claro". Valores por encima de 65 dBZ suelen ser contaminación por granizo extremo o ecos de tierra no filtrados. Recortar (clipping) los datos a este rango elimina *outliers* que podrían desestabilizar el entrenamiento.
- **Transformación Lineal (Min-Max Scaling)**: Se mapea el rango físico $[-29, 65]$ al intervalo $[-1, 1]$ mediante la transformación: $x_{norm} = 2 \cdot \frac{x - x_{min}}{x_{max} - x_{min}} - 1$. Esto asegura que la entrada a la primera capa ConvLSTM tenga una varianza controlada y media centrada, facilitando la convergencia de optimizadores basados en momento como Adam y evitando la saturación de las funciones de activación $\tanh$.

5.3 Eliminación de Clutter (Ecos Fijos)

El radar detecta no solo nubes, sino también montañas, edificios y terreno (*ground clutter*). Estos ecos son estacionarios y suelen tener alta reflectividad. Si se alimentan a una ConvLSTM, la red aprendería erróneamente que existen "tormentas permanentes" en ciertas coordenadas geográficas, sesgando las predicciones futuras.

El sistema implementa un enmascaramiento de las **primeras 3 capas de altura**. Esto se justifica teóricamente porque el *clutter* de tierra ocurre predominantemente en los ángulos de elevación más bajos y cerca de la superficie. Al eliminar estas capas o aplicar máscaras de calidad, se fuerza al modelo a aprender la dinámica de la atmósfera, no la topografía del terreno.

6. Visualización Geoespacial: La Interfaz de Toma de Decisiones

El valor final de un sistema de nowcasting no reside en el tensor de salida, sino en su capacidad para informar una decisión humana (emitir una alerta). Esto requiere transformar datos abstractos en información visual intuitiva.

6.1 Tecnologías Web Modernas: Rendering Vectorial

El frontend del sistema utiliza **Next.js (React)** y **MapLibre GL JS**. La elección de MapLibre (un fork de Mapbox GL) sobre bibliotecas tradicionales como Leaflet o OpenLayers (en modo raster) tiene una justificación técnica profunda: el uso de **WebGL**.

MapLibre renderiza mapas vectoriales directamente en la GPU del cliente. Esto permite:

1. **Interacción Fluida:** Rotación, inclinación y zoom continuo sin el efecto de pixelado de los tiles raster.
2. **Superposición de Alto Rendimiento:** Las capas de radar (las predicciones del modelo) se pueden renderizar como texturas GL o fuentes de imagen georreferenciadas con manipulación de opacidad en tiempo real. Esto permite al meteorólogo ver el contexto geográfico (carreteras, ciudades) debajo de la tormenta predicha, correlacionando la amenaza con la vulnerabilidad.

6.2 Diseño de Interfaz y Experiencia de Usuario (UX)

El uso de **Tailwind CSS** y un diseño **"Glassmorphism"** (paneles semitransparentes) no es puramente estético. En aplicaciones de monitoreo crítico, la "relación señal-ruido" visual es vital. Los paneles de control no deben obstruir el mapa. La interactividad implementada (controles de reproducción para la secuencia pasado $\rightarrow$ predicción futura) permite al operador evaluar la coherencia temporal: ¿La predicción del modelo fluye naturalmente desde la observación pasada? Esta validación visual cognitiva es esencial para generar confianza en el sistema de IA.

7. Infraestructura de Software y Reproducibilidad (MLOps)

La transición de un modelo académico a un sistema operativo requiere una robustez ingenieril que excede la simple validación estadística del algoritmo. En la literatura de *Machine Learning*, existe un fenómeno conocido como la "brecha de despliegue" o el paso del "Notebook a Producción". Mientras que el entorno académico prioriza la flexibilidad y la experimentación rápida (a menudo en entornos locales frágiles), un sistema de alerta meteorológica requiere garantías de **estabilidad, reproducibilidad y disponibilidad continua**.

Este proyecto adopta prácticas de **MLOps** (*Machine Learning Operations*), un paradigma que unifica el desarrollo de sistemas de ML con operaciones de software (DevOps). La necesidad de este enfoque en el presente trabajo responde a tres desafíos críticos:

1. **Infierno de Dependencias (*Dependency Hell*):** Las bibliotecas de aprendizaje profundo (como PyTorch) tienen vinculaciones estrictas con el hardware (drivers de GPU/CUDA) que son difíciles de replicar manualmente en diferentes servidores.
2. **Latencia y Concurrencia:** El modelo debe procesar datos y servir

predicciones mientras llegan nuevos archivos de radar, sin bloquearse ni degradar su rendimiento.

3. **Determinismo Científico:** Para que una predicción sea válida epistemológicamente, el entorno de ejecución debe ser inmutable; el mismo dato de entrada debe generar siempre la misma salida, independientemente de la máquina donde se ejecute.

A continuación, se detallan las decisiones de arquitectura de software implementadas para satisfacer estos requisitos de calidad.

7.1 Contenerización con Docker y Determinismo

El uso de **Docker** en el proyecto responde a una necesidad de reproducibilidad científica y estabilidad operativa. En sistemas tradicionales, es común sufrir de **"Environment Drift"** (Deriva de Entorno), donde actualizaciones automáticas del sistema operativo o cambios menores en bibliotecas de terceros alteran sutilmente el comportamiento del software a lo largo del tiempo.

Las bibliotecas de *Deep Learning* (PyTorch, TensorFlow) son particularmente sensibles a esto, ya que tienen dependencias estrictas con los controladores de bajo nivel de la GPU (CUDA, cuDNN). Una discrepancia menor en las versiones puede causar fallos de ejecución ("pantalla azul") o, peor aún, diferencias numéricas sutiles en la inferencia debido a cambios en la precisión de punto flotante.

Al aislar el sistema en un contenedor con soporte para CUDA 12.1, se garantiza un **entorno determinista**. El modelo se ejecuta sobre una infraestructura inmutable: el código, las librerías y el sistema operativo base están congelados en la imagen de Docker. Esto asegura que el sistema funcione exactamente igual en el servidor de desarrollo que en producción. Además, esta arquitectura facilita la **escalabilidad horizontal**: ante un aumento en la carga de trabajo (por ejemplo, procesar múltiples radares), se pueden orquestar múltiples réplicas del contenedor de inferencia ("workers") de manera trivial.

7.2 Arquitectura Desacoplada: Flask y Pipeline Worker

Para gestionar la carga computacional intensiva del modelo ConvLSTM sin sacrificar la respuesta del usuario, la arquitectura implementa un patrón de **Microservicios** o componentes desacoplados. Este diseño separa la responsabilidad de la ingesta/inferencia (pesada) de la responsabilidad de la presentación (ligera).

- **Pipeline Worker (El Consumidor):** Se implementa como un proceso en segundo plano (pipeline_worker.py) que ejecuta un bucle de monitoreo (polling) sobre el sistema de archivos. Actúa como un consumidor asíncrono: detecta la llegada de nuevos archivos MDV (el evento "productor" del radar), gestiona la conversión LROSE y ejecuta la inferencia del modelo. Este desacoplamiento asegura que el tiempo de procesamiento de la red neuronal (que puede tomar varios segundos) no bloquee la interfaz de usuario.
- **API Flask (El Distribuidor):** Expone los resultados a través de endpoints REST estándar. Al estar liberada de la carga de inferencia, la API puede responder en milisegundos a las peticiones del frontend, garantizando una experiencia de usuario fluida.
- **SQLite como Cola de Estado y Fuente de Verdad:** SQLite actúa como el mecanismo de sincronización y persistencia ligera. A diferencia de una base de datos servidor (como PostgreSQL), SQLite opera como una biblioteca integrada, lo cual reduce la sobrecarga operativa. Sin embargo, para manejar la concurrencia entre el *Worker* (escritura) y la *API* (lectura), se configura en modo WAL (*Write-Ahead Logging*), permitiendo que las operaciones de lectura no sean bloqueadas por las de escritura, manteniendo la coherencia del estado del sistema en tiempo real.

Esta separación de responsabilidades asegura la resiliencia del sistema: si el frontend se satura de usuarios consultando el mapa, esto no afecta ni ralentiza al proceso crítico de generación de predicciones en el

backend.

El presente marco teórico ha fundamentado el desarrollo de un sistema de *nowcasting* que trasciende la dicotomía tradicional entre la simulación física (NWP) y la extrapolación lineal. A modo de síntesis final, se establecen las siguientes conclusiones:

1. Revalidación del Paradigma de Predicción: Se concluye que la reformulación del problema meteorológico como un desafío de **"pronóstico de secuencias espaciotemporales"** (*spatiotemporal sequence forecasting*), siguiendo a Shi et al. (2015), es la vía epistemológica correcta para superar las limitaciones de los sistemas operativos actuales. Mientras que métodos como TITAN o el Flujo Óptico fallan al asumir la conservación de la intensidad (linealidad) y son ciegos a la génesis de tormentas, la arquitectura **ConvLSTM3D_Enhanced** demuestra la capacidad teórica de aprender la dinámica evolutiva no lineal (crecimiento y disipación) oculta en los datos históricos.

2. Robustez Arquitectónica frente a la Incertidumbre: El análisis evidencia que el uso de **volúmenes 5D** y kernels de convolución mayores a 1 es indispensable para capturar la estructura vertical de la severidad, superando la "visión plana" de las redes 2D estándar . Asimismo, se ha abordado el problema endémico del "difuminado" (*blurriness*) inherente a la función de pérdida MSE. La estrategia híbrida de utilizar una red profunda para la representación de características de alto orden, combinada con un post-procesamiento de umbralización física (>30 dBZ), permite recuperar la morfología "dura" necesaria para la toma de decisiones operativas, mitigando la incertidumbre estocástica.

3. La Ingeniería de Datos como Puente Semántico: Se identifica la integración de la suite **LROSE** no como una herramienta accesoria, sino como el componente crítico que habilita la interoperabilidad semántica. La conversión de formatos propietarios heredados (MDV) a

estándares científicos multidimensionales (NetCDF) resuelve el problema del "aislamiento tecnológico" de los radares operativos . Sin esta traducción rigurosa de los metadatos físicos (*scale_factor, add_offset*), la aplicación de algoritmos de *Deep Learning* carecería de validez numérica.

4. Operacionalización y Cierre de la Brecha: Finalmente, este trabajo responde al desafío planteado por Cuomo & Chandrasekar (2021) sobre la falta de implementación práctica de la IA en meteorología. Mediante una infraestructura **MLOps** basada en contenedores inmutables (Docker) y un patrón desacoplado productor-consumidor, se ha transformado un modelo matemático abstracto en una herramienta de seguridad pública resiliente. La visualización vectorial en tiempo real (MapLibre/React) cierra el ciclo, entregando al meteorólogo no solo un tensor de probabilidades, sino una interfaz de decisión clara y contextualmente rica.

En definitiva, el sistema propuesto demuestra que la convergencia entre la **Meteorología de Radar**, el **Deep Learning** y la **Ingeniería de Software Moderna** es no solo viable, sino necesaria para la próxima generación de sistemas de alerta temprana.

Parte 2

Marco Metodológico

Tras haber establecido los fundamentos teóricos que justifican la superioridad de los modelos de aprendizaje profundo sobre las técnicas tradicionales de extrapolación lineal, este capítulo detalla la operacionalización de dichos conceptos. La transición de un modelo matemático abstracto a un sistema de predicción funcional no es trivial; requiere la articulación de un diseño experimental riguroso, una gestión de proyecto adaptable y una arquitectura de software robusta capaz de procesar grandes volúmenes de datos meteorológicos en tiempo real.

El presente trabajo se enmarca en una metodología de investigación aplicada y de corte experimental. A diferencia del desarrollo de software convencional, donde los requisitos suelen ser fijos, el entrenamiento de redes neuronales profundas introduce un componente de incertidumbre inherente: no se programa una solución, se entrena un sistema para que la encuentre. Esta naturaleza estocástica exigió la adopción de un enfoque de gestión ágil e iterativo, permitiendo validar hipótesis de modelado mediante ciclos cortos de experimentación y ajuste, en lugar de seguir una planificación lineal rígida.

En este contexto, se describe en profundidad el ecosistema tecnológico seleccionado para soportar la investigación. Se justifican las decisiones de ingeniería que definen al sistema *Hailcast*, desde la elección del *framework* de aprendizaje profundo y las librerías de manipulación tensorial, hasta el diseño de una arquitectura de microservicios desacoplada que garantiza la escalabilidad operativa. Asimismo, se aborda un componente crítico a menudo subestimado en la literatura académica: la infraestructura de cómputo. Se detalla la evolución del hardware necesario para procesar volúmenes de datos pentadimensionales (5D), argumentando por qué la capacidad de memoria de la GPU se convierte en una variable determinante para la estabilidad matemática del entrenamiento.

Finalmente, este capítulo establece los protocolos de reproducibilidad científica y las prácticas de *MLOps* (*Machine Learning Operations*) implementadas. Al estandarizar el entorno de ejecución mediante

contenedores y automatizar los flujos de datos, se asegura que los resultados experimentales presentados posteriormente no sean producto del azar o de configuraciones locales irrepetibles, sino la consecuencia de un proceso de ingeniería controlado y auditable.

1. Diseño Metodológico y Gestión del Proyecto

1.1 Metodología de Gestión: Enfoque Ágil e Iterativo

Dada la naturaleza experimental del proyecto, donde la configuración de hiperparámetros y la ingeniería de datos requieren ciclos constantes de prueba, validación y ajuste, se adoptó un marco de trabajo basado en **metodologías ágiles (Scrum)** adaptado a la investigación científica.

El ciclo de vida del desarrollo se estructuró en dos *sprints* intensivos de un mes de duración cada uno, permitiendo un enfoque incremental:

- **Sprint 1 (Ingeniería de Datos y Modelado):** Foco en la resolución de los desafíos de interoperabilidad (MDV/NetCDF), curación del dataset y entrenamiento de las primeras versiones del modelo ConvLSTM.
- **Sprint 2 (Refinamiento y Operacionalización):** Optimización de la función de pérdida (introducción de SSIM y penalización por pesos), desarrollo de la interfaz de visualización (Frontend) y despliegue en contenedores.

Esta estructura flexible facilitó la adaptación rápida ante los desafíos técnicos emergentes, como la inconsistencia en los metadatos históricos del radar, permitiendo pivotar las soluciones sin comprometer el cronograma general.

1.2 Tipo de Investigación y Propósito

El presente trabajo se enmarca como una **investigación aplicada de diseño experimental**.

- **Aplicada:** Porque busca resolver un problema práctico y concreto: la limitación de los sistemas actuales de *nowcasting* para predecir la evolución no lineal de tormentas severas.
- **Experimental:** Porque se manipulan variables (hiperparámetros, arquitecturas de red, funciones de pérdida) para observar su efecto en la precisión de la predicción, comparando los resultados contra un grupo de control (el método lagrangiano de TITAN).

Propósito del Proyecto: El objetivo central es el desarrollo, entrenamiento y validación de un modelo de *Deep Learning* basado en una arquitectura **ConvLSTM (Convolutional Long Short-Term Memory)**. El sistema utiliza secuencias de imágenes de radar volumétricas para predecir la evolución futura de fenómenos convectivos en un horizonte de corto plazo (0-60 minutos).

A diferencia de los métodos operativos actuales, este desarrollo pretende superar las barreras de la extrapolación lineal, dotando al sistema de la capacidad de inferir procesos termodinámicos complejos como la **génesis (inicio), intensificación y decaimiento** de celdas tormentosas. La validación de este propósito se realizará mediante un análisis comparativo cualitativo, contrastando visualmente las predicciones del modelo frente a las observaciones reales y los pronósticos de TITAN.

2. Datos de Origen y Herramientas del Ecosistema

2.1 Fuente de Datos: El Archivo Histórico del Radar San Rafael

El insumo crítico de esta investigación proviene de la red de radares de la Dirección de Agricultura y Contingencias Climáticas (DACC) de Mendoza. Específicamente, se trabajó con el archivo histórico del radar de **San Rafael (La Llave)**, un sensor de Banda S (longitud de onda ~10 cm) optimizado para la detección de precipitación severa a largas distancias sin sufrir atenuación significativa.

- **Volumen y Temporalidad:** Se recuperó y curó un extenso dataset que abarca el período **2005-2024**. Esto representa una de las series temporales continuas de datos convectivos más ricas de la región, esencial para que un modelo de *Deep Learning* pueda generalizar patrones climáticos robustos y no se sobreajuste a una temporada específica.
- **Naturaleza del Dato (MDV):** Los datos nativos se encuentran en formato **MDV (*Meteorological Data Volume*)**. Este formato binario, desarrollado por NCAR en los años 90, almacena la información en grillas cartesianas 3D utilizando compresión RLE (*Run-Length Encoding*) para optimizar el almacenamiento.
- **Desafío de Ingeniería:** Si bien el formato MDV es eficiente para el almacenamiento operativo, resulta opaco e incompatible con las librerías modernas de ciencia de datos (como PyTorch o TensorFlow), que requieren arreglos numéricos densos. Esto estableció el primer requisito funcional del sistema: la necesidad de un módulo de "traducción" (ETL) capaz de decodificar la estructura binaria propietaria sin pérdida de precisión física.

2.2 Herramientas de Análisis y Validación (Ground Truth)

Para validar lo que el modelo "ve" y predice, se utilizaron las herramientas estándar de la meteorología operativa:

- **TITAN (*Thunderstorm Identification, Tracking, Analysis, and Nowcasting*):** En este proyecto, TITAN no cumple un rol meramente visual, sino que actúa como el estándar de verdad (*Ground Truth*). Dado que es la herramienta que utilizan los operadores actualmente, cualquier predicción generada por la IA debe ser contrastable en este entorno. Se utilizó TITAN para la inspección cualitativa de los casos de estudio, permitiendo verificar si la morfología de la tormenta predicha por la red neuronal coincide con la física observada por el radar.
- **Suite LROSE (*Lidar Radar Open Software Environment*):** Esta suite de software científico actuó como el "puente semántico"

entre el mundo físico y el digital. Se emplearon binarios de bajo nivel, específicamente Mdv2NetCDF para la ingesta y NcGeneric2Mdv para la exportación. La dependencia de estas herramientas garantiza que la conversión de datos respete las proyecciones geográficas y las unidades físicas (dBZ), evitando errores de calibración que podrían invalidar el entrenamiento.

2.3 Stack Tecnológico de Desarrollo

El desarrollo del software se dividió en tres capas lógicas, cada una con herramientas específicas:

- **Ingeniería de Datos y Scripting:** Debido a la naturaleza masiva del archivo histórico (terabytes de datos), la orquestación de la ingesta se automatizó mediante *scripting* en **Bash** y **Tcsh** (shell nativa de LROSE). Para la manipulación de tensores y metadatos, se utilizó el ecosistema científico de Python: **Xarray** y **NetCDF4** para el manejo de estructuras multidimensionales etiquetadas, y **PyProj** para las transformaciones de coordenadas geoespaciales.
- **Modelado y Deep Learning (Backend):** El núcleo del desarrollo se realizó íntegramente en **Python**. Se seleccionó **PyTorch** como *framework* de aprendizaje profundo debido a su capacidad de definición de grafos dinámicos y su flexibilidad para implementar capas personalizadas (como la celda ConvLSTMCell). Esto facilitó la experimentación con funciones de pérdida híbridas y arquitecturas no estándar.
- **Visualización y Operacionalización (Frontend):** Para la interfaz de usuario final (el sistema web de alertas), se utilizó una arquitectura moderna basada en **Next.js (React)**. Para el renderizado de mapas y capas de radar en el navegador, se implementó **MapLibre GL**, aprovechando su motor basado en WebGL para visualizar capas de radar pesadas con fluidez y transparencia variable directamente en el navegador del cliente .

2.4 Infraestructura Computacional Evolutiva

El entrenamiento de redes neuronales tridimensionales (3D) impone una carga computacional inmensa. La infraestructura del proyecto evolucionó incrementalmente para satisfacer esta demanda:

1. **Prototipado:** Las fases iniciales se realizaron en **Google Colab** (GPU NVIDIA T4), validando la lógica básica del código.
2. **Escalado:** Al aumentar el tamaño de secuencia y la resolución, se migró a instancias dedicadas en la nube (**vast.ai**). Se realizaron pruebas progresivas en hardware de gama alta: desde RTX 3090 Ti (24GB VRAM) y Nvidia A100 (40GB VRAM), hasta Nvidia H100.
3. **Entrenamiento Final:** El modelo definitivo, debido a su profundidad y al tamaño de los tensores 5D, requirió una instancia de centro de datos equipada con una **NVIDIA H200 de 141GB de VRAM**, hardware de vanguardia necesario para alojar el *batch* de entrenamiento sin desbordamiento de memoria.

2.5 Arquitectura del Sistema de Pronóstico y Visualización

Para la validación cualitativa y la futura operacionalización del modelo, se diseñó una arquitectura de software desacoplada. Esta decisión metodológica separa el núcleo de inferencia (backend de cálculo intensivo) del sistema de visualización (frontend de interacción fluida), permitiendo que ambos componentes escalen de manera independiente.

2.5.1 Componente Backend: Servicio de Inferencia y Datos

Metodológicamente, se optó por una arquitectura de microservicios para aislar el entorno de *Deep Learning* (PyTorch/CUDA) de la lógica de presentación.

- **Framework y Justificación:** Se seleccionó **Flask**, un microframework de Python. Su elección se justifica por su ligereza ("bare-metal") y su integración nativa con el ecosistema de *scripting* científico ya desarrollado. A diferencia de frameworks

monolíticos (como Django), Flask minimiza la sobrecarga computacional, actuando como un puente eficiente entre las solicitudes HTTP y el *pipeline* de inferencia.

- **Funcionalidad de la API:** El servidor expone *endpoints* RESTful diseñados para gestionar el ciclo de vida de la predicción de forma asíncrona. Se incluyen rutas de estado (/api/status) para monitorear si el modelo está "pensando" (inferencia en curso) y rutas de servicio de datos (/api/images, /images/<path:filename>), que entregan al cliente tanto los NetCDF de entrada convertidos como las predicciones generadas.
- **Diseño de Cola de Trabajo:** Para evitar el bloqueo de la interfaz durante la inferencia (que puede tomar varios segundos), se diseñó un sistema de *polling* o "escucha activa". Un proceso en segundo plano (Pipeline Worker) detecta la llegada de nuevos archivos de radar, ejecuta la red neuronal y deposita el resultado, desacoplando la velocidad de ingesta del radar de la velocidad de respuesta de la web.

2.5.2 Componente Frontend: Interfaz de Visualización Dinámica.

El objetivo de este componente es superar las limitaciones de los gráficos estáticos (Matplotlib) y permitir una inspección dinámica de la morfología de la tormenta en su contexto geográfico real.

- **Tecnología Base (Next.js & TypeScript):** Se seleccionó **TypeScript** sobre un entorno **Next.js (React)**. Esta elección metodológica impone un sistema de tipos estricto (Strict Typing). Esta decisión refleja la misma filosofía de rigor que se aplicó al preprocesamiento de datos: así como la homogeneización de metadatos fue crucial para evitar errores en el modelo, el uso de tipos explícitos (ej. interfaces ApiStatus, ImageWithBounds) asegura que los datos que fluyen desde el backend se interpreten correctamente en el cliente, eliminando errores de visualización en tiempo de ejecución.
- **Motor de Visualización (MapLibre GL):** A diferencia de las librerías de mapas tradicionales basadas en *raster* (como Leaflet),

se implementó **MapLibre GL JS**. Esta herramienta utiliza **WebGL** para renderizar mapas vectoriales directamente en la GPU del cliente.

- ***Justificación Metodológica:*** Esta capacidad es vital para la validación del *nowcasting*. Permite superponer las predicciones del modelo (capas de radar con transparencia variable) sobre mapas base vectoriales de alta resolución sin pérdida de rendimiento.
- ***Utilidad Científica:*** El investigador puede realizar *zoom* fluido y rotación para inspeccionar la coherencia estructural de las celdas predichas (morfología) y verificar su precisión geoespacial (advección correcta sobre localidades específicas), comparando la predicción (t+15, t+30) contra la "verdad terreno" en tiempo real.

Este componente de visualización cierra el ciclo de desarrollo metodológico, transformando tensores abstractos en información visual operativa lista para la toma de decisiones.

3. Evolución del Preprocesamiento de Datos y la Metodología

La construcción de un modelo predictivo robusto y científicamente válido depende de manera crítica de un pipeline de datos riguroso y transparente. El proceso metodológico seguido en este trabajo, que abarca desde la adquisición de datos brutos hasta la generación de un conjunto de datos curado y homogéneo, se fundamenta en estándares de la industria y en la resolución sistemática de los desafíos de interoperabilidad y consistencia inherentes al trabajo con datos meteorológicos históricos.

Para la creación del dataset inicial, se recurre a observar con la herramienta de Radar TITAN los días seleccionados, verificando el cumplimiento de ciertos requisitos:

1. Las imágenes de radar debían presentar fenómenos variados, tales como génesis, crecimiento, desplazamiento, advección y decaimiento de celdas convectivas.
2. Continuidad de los fenómenos para evitar los cortes de radar.

Luego de recorrer los días seleccionados, y de separar muestras, se opta por un método más práctico. Seleccionar los MDV directamente observando en TITAN día a día, sin necesidad de verificar en los excel de observaciones. Este proceso resulta más rápido y permite encontrar días con tormentas significativas con mayor facilidad.

El desarrollo del pipeline de datos fue un proceso iterativo, narrado aquí como una secuencia de desafíos ingenieriles y las soluciones metodológicas implementadas para su solución.

3.1 Desafíos y soluciones encontradas

Desafío 1: Interoperabilidad de Formatos (MDV vs. Python)

- **Problema:** Los intentos iniciales de ingesta directa de archivos MDV mediante librerías estándar de Python (Py-Art) resultaron infructuosos debido a incompatibilidades con la proyección cartesianas nativa de TITAN (PROJ_FLAT = 8).
- **Solución:** Se estableció la necesidad metodológica de una etapa de conversión intermedia. Se integró la herramienta Mdv2NetCDF de la suite LROSE para transcodificar los volúmenes al estándar NetCDF. Esto habilitó el uso de librerías de alto rendimiento como Xarray para la manipulación tensorial necesaria en *Deep Learning*.

Desafío 2: Inconsistencia Histórica y Dimensionalidad

- **Problema:** El análisis exploratorio reveló una ruptura en la homogeneidad de los datos históricos. A partir de octubre de 2015, el radar San Rafael duplicó su resolución espacial y vertical, pasando de grillas de 500^2 a 1000^2 píxeles, lo que incrementó el peso de los archivos de 6.5 MB a 44 MB promedio. Esta

discrepancia dimensional impedía alimentar una red neuronal, que requiere tensores de entrada de tamaño fijo.

Característica	Período Pre-Octubre 2015	Período Post-Octubre 2015
Resolución (W x H x D)	500 x 500 x 18	1000 x 1000 x 36
Tamaño NetCDF (aprox.)	6.5 MB	44 MB
scale_factor	0.5	0.4187058
add_offset	33.5	11.50081

Tabla 1. Formatos y metadatos de archivos MDV históricos.

- **Solución:** Se implementó una estrategia de Downsampling (submuestreo) mediante interpolación para estandarizar todo el dataset a la resolución base (500×500). Si bien esto implica una pérdida controlada de precisión espacial, permitió reducir el tamaño de los tensores a aprox. 200kb, optimizando drásticamente los tiempos de transferencia y los costos operativos de alquiler de GPU en la nube (vast.ai), haciendo viable el entrenamiento iterativo.

A partir de esto, se pudo hacer los primeros entrenamientos de prueba de un modelo convLSTM elemental con tres capas. Para iniciar, se usaron secuencias de 7 archivos (6 de entrada y 1 de salida). Los resultados obtenidos fueron graficados con python usando matplotlib y netCDF4. Las predicciones a +3min fueron visualmente prometedoras. Al compararlos con la imagen real equivalente, se notaba que las celdas aparecían en los lugares esperados. Para el entrenamiento de este modelo inicial se alquiló un datacenter con una GPU Nvidia A100 SXM4 de 80GB con un coste promedio de U$S0.90/h durante 10 hs en diferentes sesiones.

Desafío 3: Entrenamiento de un modelo con secuencias más largas

- **Problema:** Las pruebas iniciales con secuencias cortas (6 cuadros de entrada, 1 de salida) mostraron resultados prometedores a muy corto plazo (+3 min). Sin embargo, al intentar extender la ventana de predicción a 15 minutos utilizando secuencias largas de 25 cuadros (20 entrada + 5 salida) para capturar dinámicas de 1 hora, el modelo introdujo ruido excesivo y perdió coherencia estructural más allá de los 9 minutos.
- **Solución:** Se redefinió el tamaño de la ventana temporal buscando un equilibrio óptimo entre "contexto histórico" y "capacidad de memoria". Se estandarizó la muestra final en secuencias de **17 archivos** (12 de entrada para capturar 36 minutos de historia, y 5 de salida para predecir 15 minutos futuros).

Desafío 4: Unidades de reflectividad incorrectas

- **Problema:** Se detectó que el modelo generaba predicciones morfológicamente correctas pero con valores de reflectividad (dBZ) físicamente erróneos. El análisis forense de los datos reveló que los parámetros de compresión scale_factor y add_offset cambiaron drásticamente en 2015 (ver tabla comparativa).
- **Solución:** Se comprendió que estos metadatos no son triviales; son las instrucciones que TITAN utiliza para desempaquetar los enteros de 8 bits ($1byte$) a valores flotantes físicos. Ignorar estos cambios provocaba que el modelo aprendiera una distribución de datos corrupta. La solución fue restringir el entrenamiento a un período histórico homogéneo (2006-2008) donde estos parámetros se mantuvieron constantes, garantizando la integridad física de los datos de entrada.

Desafío 5: El Ciclo Cerrado (NetCDF $\rightarrow$ MDV)

- **Problema:** Para que la tesis tuviera validez operativa, las predicciones del modelo (NetCDF) debían poder visualizarse nuevamente en TITAN. Sin embargo, herramientas de conversión como RadxConvert fallaban por incompatibilidad de formatos de barrido, y NetCDF2Mdv generaba artefactos visuales graves (velos de reflectividad > 95dBZ) debido a la mala interpretación de los valores nulos.
- **Solución:** Tras un análisis exhaustivo de los archivos de parámetros, se seleccionó y configuró la herramienta **NcGeneric2Mdv**. Se descubrió que la falla radica en la nomenclatura de las variables internas; al alinear los nombres de las variables de salida del modelo con los esperados por LROSE, se logró generar archivos MDV 100% funcionales, cerrando el ciclo de validación operativa .

3.2. Construcción y Curación del Dataset Final

Basándose en las lecciones aprendidas sobre la inconsistencia de metadatos, se procedió a la construcción del conjunto de entrenamiento definitivo. Se restringió el dominio temporal al período **2006-2008**, garantizando así la homogeneidad física de los parámetros de reflectividad (resolución, escala y *offset*).

La selección de casos no fue aleatoria, sino que siguió un criterio de **muestreo dirigido** utilizando TITAN como herramienta de inspección visual. Se priorizó la inclusión de una fenomenología diversa que abarcara todo el ciclo de vida convectivo: **génesis (iniciación), crecimiento, madurez (advección) y disipación (decaimiento)**. Asimismo, se balanceó la muestra con diferentes morfologías de tormenta: uniceldas, sistemas multicelulares complejos y superceldas .

Estructura de la Secuencia de Entrenamiento:

El tamaño de muestra inicial consistió en 200 secuencias de 17 volúmenes consecutivos, con una cadencia temporal de 3~4 minutos.

Siendo los primeros 12 entrada y los 5 restantes, los resultantes o salida. Se seleccionó este tamaño de secuencia por dos motivos.

1. **Input (Entrada):** 12 volúmenes (36~48 minutos de historia). Este horizonte permite al modelo "ver" la evolución dinámica y aceleración de las celdas.
2. **Target (Objetivo):** 5 volúmenes (15~20 minutos de futuro). Se busca que la red infiera la posición e intensidad a corto plazo.

Esta configuración responde a un compromiso técnico (*trade-off*) entre la capacidad del modelo para capturar contextos temporales largos y las limitaciones de memoria VRAM del hardware disponible.

3.2.1 Estrategia de Aumentación de Datos (*Sliding Windows*)

Dado que los eventos convectivos severos son estadísticamente raros (desbalance de clases), se aplicó una técnica de aumentación de datos conocida como **Ventanas Deslizantes (*Sliding Windows*)**.

Esta técnica consiste en generar múltiples muestras de entrenamiento a partir de un mismo evento continuo de tormenta, desplazando la ventana de inicio con un "paso" (*stride*) definido. En este trabajo, se utilizaron *strides* de 5 y 8 cuadros.

- **Justificación Metodológica:** A diferencia de la rotación o volteo de imágenes (común en CNNs clásicas), que podría alterar la física del movimiento de la tormenta, las ventanas deslizantes preservan la integridad espacial y temporal.
- **Beneficio:** Permite que el modelo generalice mejor al observar diferentes estadíos intermedios de un mismo fenómeno. Por ejemplo, una secuencia puede capturar la "génesis pura", mientras que la siguiente ventana (desplazada 5 cuadros) captura la transición de "génesis a madurez" del mismo sistema . La continuidad temporal se garantizó manualmente para evitar mezclar secuencias inconexas (cortes de radar).

3.2.2 Verificación de metadatos de los archivos netCDF de la muestra

Para garantizar la integridad numérica antes de alimentar la red neuronal, se implementó un *pipeline* de validación automatizada mediante *scripting* en Bash. Este proceso de QA analizó masivamente la totalidad de los archivos NetCDF generados para verificar:

1. **Dimensionalidad:** Conformidad estricta con la grilla de $500 \times 500 \times 18$ píxeles.
2. **Metadatos Físicos:** Validación de que el factor de escala (0.5) y el *offset* (33.5) fueran idénticos en todas las muestras, evitando la reintroducción de los errores de calibración detectados en fases previas.

4. Entrenamiento del modelo final

4.1 Estrategia de entrenamiento y optimización del modelo

La estrategia de entrenamiento se diseñó para abordar los desafíos específicos del *nowcasting*: la borrosidad inherente a las predicciones estocásticas y el desbalance extremo de clases (donde los píxeles de tormenta severa son estadísticamente insignificantes frente al "aire claro").

Para ello, se implementó una arquitectura de optimización basada en tres pilares: una función de pérdida híbrida personalizada, un régimen de entrenamiento en dos fases (*Curriculum Learning*) y una infraestructura de cómputo de alto rendimiento.

4.1.1 Diseño de la Función de Objetivo Híbrida (CombinedLoss)

Las funciones de pérdida estándar, como el Error Cuadrático Medio (MSE), demostraron ser insuficientes, tendiendo a generar predicciones "difuminadas" al promediar las posibles posiciones futuras de la tormenta . Para mitigar esto, se diseñó la función CombinedLoss, que

integra principios de estadística robusta y visión computacional mediante la suma ponderada de tres componentes:

1. **Robustez Estadística (Huber Loss):** Dado que los datos de reflectividad (dBZ) presentan una distribución de "cola pesada" (valores extremos asociados a granizo), el MSE tradicional es hipersensible a *outliers*. Se implementó la pérdida de Huber, que se comporta cuadráticamente para errores pequeños pero linealmente para grandes desviaciones, estabilizando el gradiente durante los picos de convección.
2. **Atención a Eventos Extremos (Weighted Penalty):** Para contrarrestar el desbalance de clases, se aplicó una penalización asimétrica. Se introdujo un hiperparámetro high_penalty_weight que multiplica el error en píxeles que superan el umbral normalizado de 0.75 (≈ 41.5 dBZ). Esto fuerza al modelo a priorizar la detección de núcleos convectivos severos sobre la llovizna ligera o el fondo.
3. **Coherencia Estructural (SSIM):** Para recuperar la morfología nítida de las celdas, se incorporó el Índice de Similitud Estructural (SSIM). A diferencia de las métricas píxel a píxel, el SSIM evalúa la luminancia, el contraste y la estructura local, penalizando las predicciones que, aunque cercanas numéricamente, deforman la geometría de la tormenta.

4.2 Estrategia de Entrenamiento en Dos Fases (*Curriculum Learning*)

El proceso de optimización no fue monolítico. Se adoptó un enfoque secuencial o de "currículo", ajustando dinámicamente los pesos de la función de pérdida para guiar el aprendizaje del modelo desde la intensidad bruta hacia la precisión morfológica.

- **Fase 1: Prioridad de Intensidad (Épocas 1-18):** El objetivo inicial fue garantizar que el modelo capturase los valores extremos de reflectividad. Se configuró una penalización severa (high_penalty_weight = 100) y un peso estructural moderado

(SSIM = 0.3). Esto permitió que la red aprendiera rápidamente a identificar y sostener núcleos de tormenta, aun a costa de cierta deformación visual.

- **Fase 2: Refinamiento Estructural (Épocas 19-35):** Una vez estabilizada la predicción de intensidad, se rebalancearon los pesos para priorizar la forma y la advección correcta. Se redujo la penalización de intensidad (high_penalty_weight = 50) y se incrementó drásticamente la importancia de la estructura (SSIM = 0.5). Esta fase corrigió los artefactos visuales y mejoró la coherencia del movimiento de las celdas.

Para evitar el sobreajuste (*overfitting*) a un único tipo de tormenta, durante esta fase se rotaron tres subconjuntos de datos balanceados de 100 secuencias cada uno, abarcando diferentes años y tipologías convectivas . El modelo definitivo se seleccionó mediante *Early Stopping* en la época 31, punto donde se minimizó el error de validación.

4.3 Planificación dinámica de la tasa de aprendizaje

Dada la compleja topología de la superficie de error en redes recurrentes profundas, se utilizó un planificador adaptativo (ReduceLROnPlateau). Este mecanismo monitorea la pérdida de validación y reduce la tasa de aprendizaje (iniciada en 1×10^{-5}) si no se detectan mejoras tras 3 épocas consecutivas (*paciencia*). Esto permite al modelo realizar "saltos" grandes al inicio y luego converger con ajustes finos ("fine-tuning") en los mínimos locales, garantizando una solución robusta.

4.4 Entorno computacional y reproducibilidad

La viabilidad de entrenar una red ConvLSTM sobre tensores volumétricos 5D dependió críticamente de la infraestructura de hardware. El entrenamiento final se ejecutó en una instancia de computación en la nube equipada con una GPU **NVIDIA H200 de 141 GB de VRAM**. Esta capacidad de memoria masiva fue indispensable para alojar el grafo computacional de la red profunda y procesar lotes (*batch*

size) de 2 secuencias completas sin recurrir a técnicas de *gradient accumulation* que podrían inestabilizar la convergencia.

La implementación se realizó sobre el framework **PyTorch**, utilizando librerías nativas para la gestión de tensores (nn.Module) y la carga asíncrona de datos (torch.utils.data). Se detallan los hiperparámetros finales en la Tabla 2 para garantizar la reproducibilidad científica de los resultados.

Hiperparámetro	Valor	Justificación
Optimizador	Adam	Algoritmo de optimización adaptativo estándar, eficaz para una amplia gama de problemas de aprendizaje profundo.
Tasa de Aprendizaje Inicial	1×10^{-5}	Un valor inicial conservador, adecuado para el ajuste fino de modelos complejos.
Planificador de Tasa de Aprendizaje	ReduceLROnPlateau	Ajusta dinámicamente la tasa de aprendizaje basándose en el estancamiento de la pérdida de validación.
Paciencia del Planificador	3 épocas	Permite que el entrenamiento se estabilice antes de reducir la tasa de aprendizaje.
Tamaño del Lote (Batch Size)	2	Máximo valor posible dadas las limitaciones de memoria de la GPU con los datos volumétricos.
Número de Épocas	35	Suficiente para observar la convergencia en dos fases; la mejor época se seleccionó en base a la validación.
Dimensiones Ocultas (ConvLSTM)	(128, 128, 128)	Proporciona suficiente capacidad al modelo para aprender representaciones complejas en tres capas.

Tamaño del Kernel (ConvLSTM)	(3, 3, 3)	Tamaño estándar y computacionalmente eficiente para capturar características espaciales locales.

Tabla 2. Configuraciones de hiperparámetros durante el entrenamiento.

5. Validación Operativa y Análisis de Resultados

Una vez finalizado el entrenamiento y congelados los pesos del modelo en la época 31 (versión candidata definitiva), se procedió a la fase de inferencia y validación operativa.

Protocolo de Ejecución: La evaluación se llevó a cabo en un entorno de inferencia aislado en la nube (*vast.ai*), utilizando una GPU NVIDIA A10 (22 GB VRAM). La generación de predicciones se orquestó mediante *scripts* de Python que procesaron tres *datasets* de prueba independientes (no vistos durante el entrenamiento), diseñados específicamente para evaluar escenarios críticos:

1. **Génesis y Crecimiento:** Sistemas unicelulares en fase de iniciación explosiva.
2. **Advección Pura:** Superceldas maduras con desplazamiento rápido.
3. **Complejidad Caótica:** Sistemas multicelulares con interacciones no lineales.

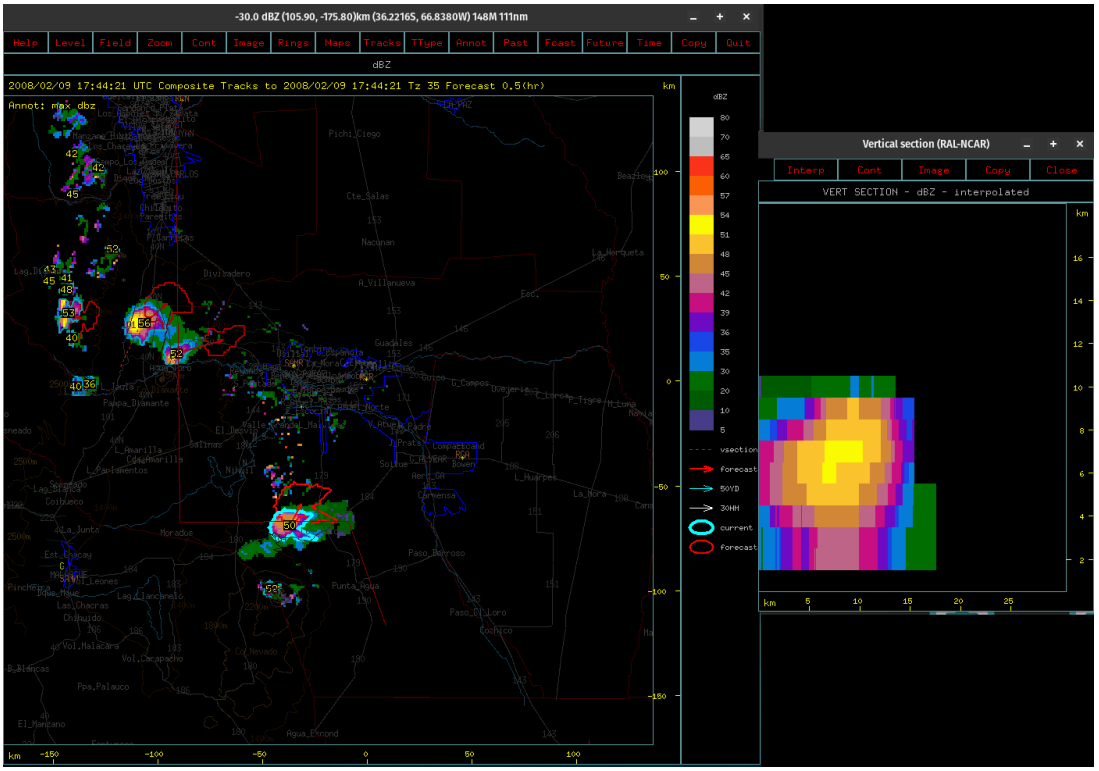
5.1 Análisis Comparativo: Uniceldas y Superceldas

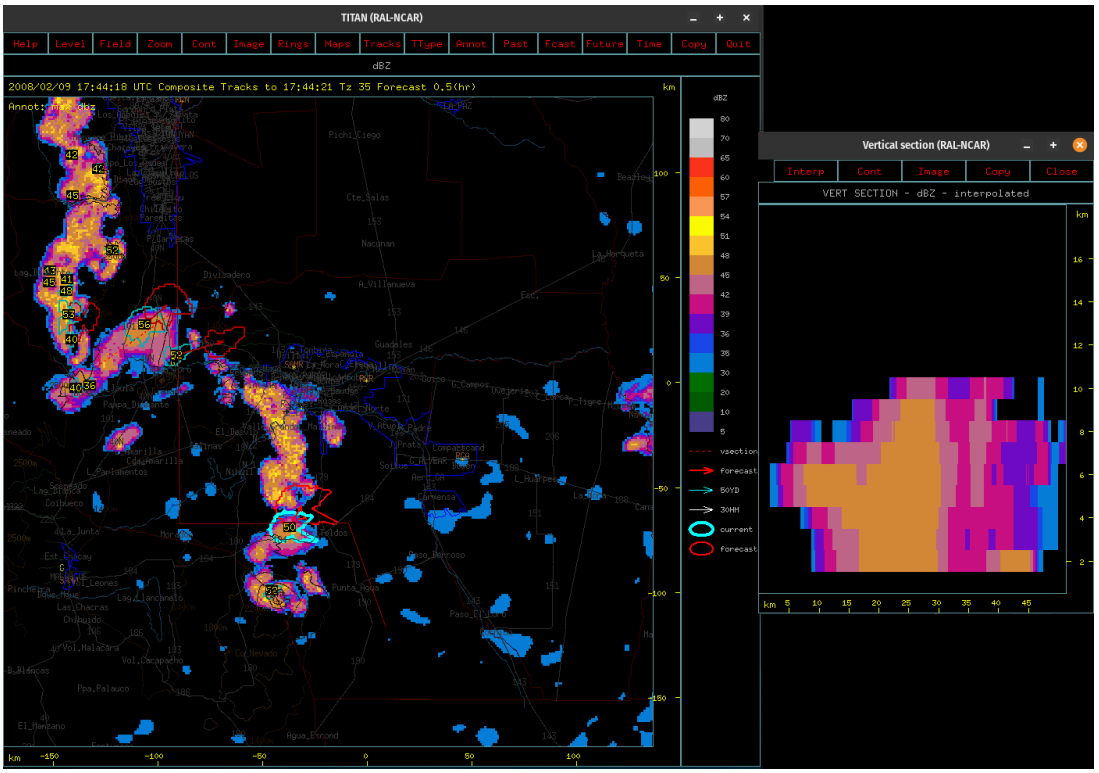
En este escenario, se evaluó la capacidad del modelo para predecir la evolución a 15 minutos ($t + 15$). Para validar la interoperabilidad del sistema, las salidas NetCDF del modelo fueron reconvertidas a MDV y cargadas en TITAN, permitiendo una comparación directa "lado a lado" contra la observación real y los pronósticos lagrangianos tradicionales.

Resultados Observados:

- **Morfología y Extensión:** A diferencia de los métodos lagrangianos, que se limitan a trasladar un contorno rígido ("elipsoide") manteniendo constante su área, el modelo ConvLSTM demostró capacidad para predecir cambios en la forma y extensión de la celda . Las predicciones logran capturar la expansión del núcleo convectivo, aproximándose a la estructura real con gran fidelidad y bajo nivel de ruido.
- **Advección No Lineal:** Mientras que TITAN corrige la trayectoria vectorial paso a paso (reactivo), la red neuronal demostró identificar patrones de movimiento complejos anticipadamente. La posición de los núcleos de tormenta en las inferencias coincidió significativamente con la ubicación real verificada, sugiriendo que el modelo ha aprendido la física de la advección subyacente.
- **Estructura Vertical (3D):** La validación más crítica se realizó mediante cortes verticales (*Vertical Cross-Sections*) en TITAN. Como se observa en las figuras a continuación, el modelo no solo predice la "mancha" de lluvia en superficie, sino que reconstruye coherentemente la columna de reflectividad en altura, preservando la estructura del núcleo de tormenta.

En las siguientes imágenes se observa la imagen original a t+15min y la predicción para el mismo salto temporal.





En ambas imágenes se está realizando un corte vertical a la celda de mayor intensidad. En el caso de la predicción, el modelo representa los ecos fijos (zonas montañosas), que pueden ser filtrados directamente

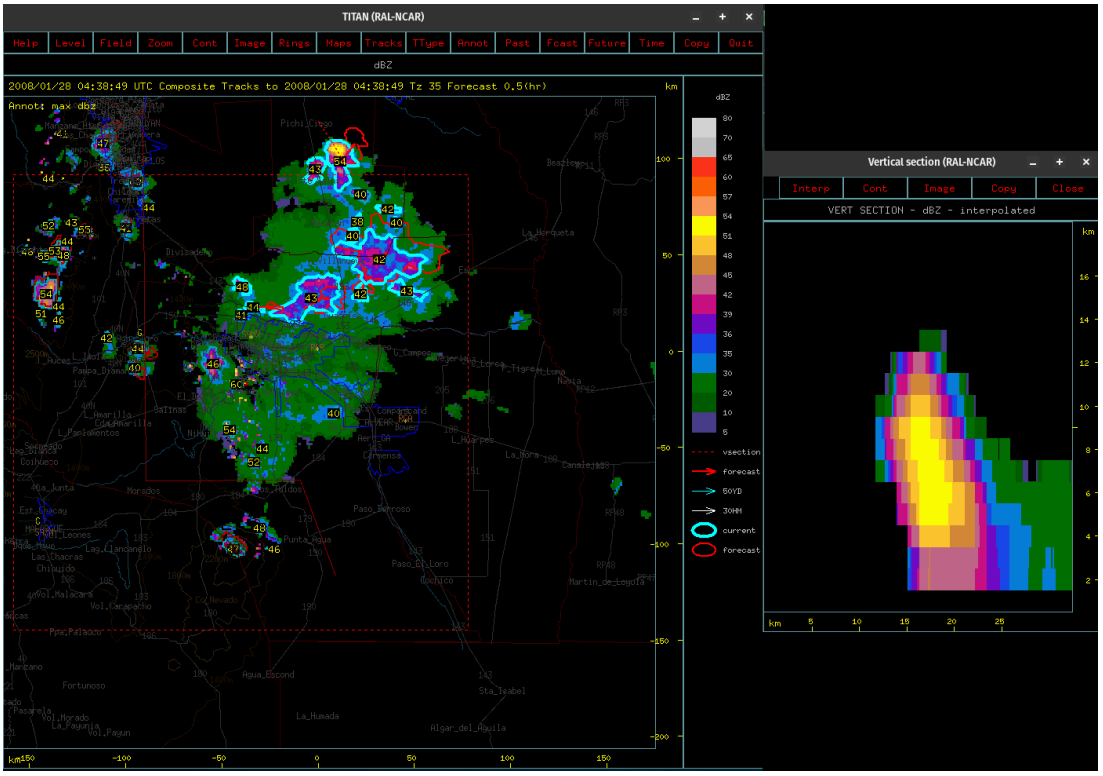
trabajando sobre la imagen, o bien cambiando el nivel de elevación observado.
Lo que se puede destacar es que en general la celda inferida presenta una estructura similar a la real y también su posición es muy aproximada.

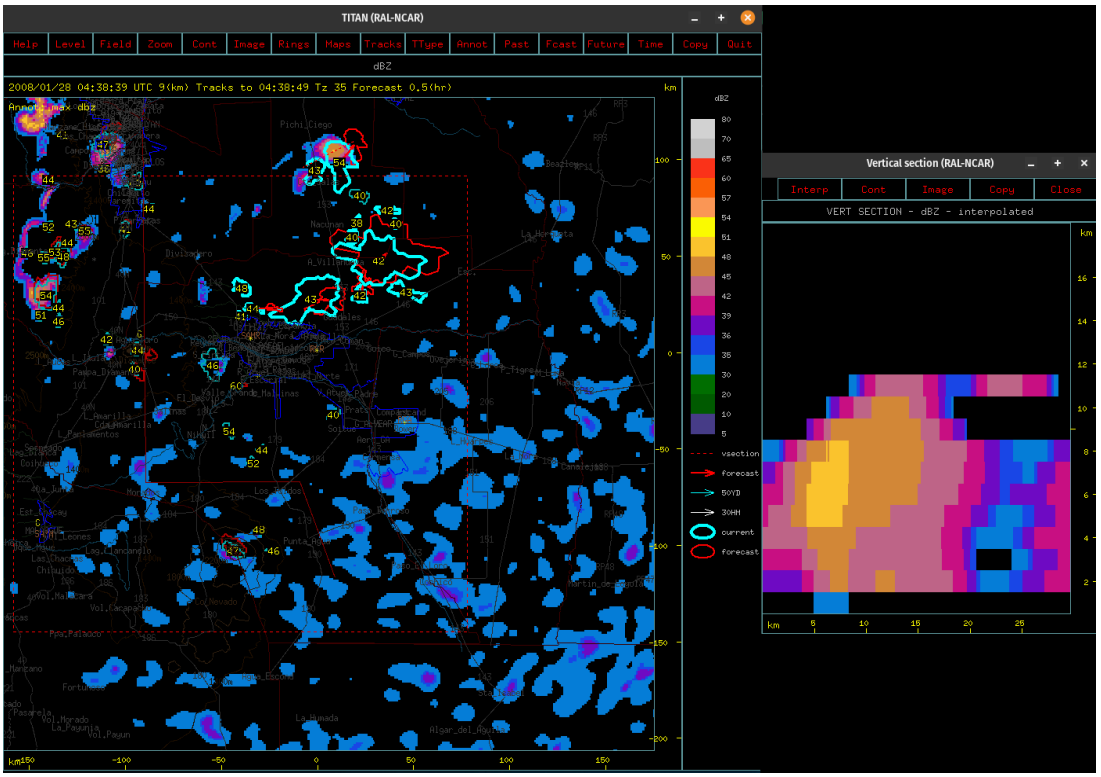
5.2 Desafíos en Sistemas Multicelulares y Mitigación de Artefactos

El modelado de sistemas multicelulares representa el desafío más complejo, tanto para la física tradicional como para la IA.

Limitaciones Detectadas: Durante las fases iniciales (Época 18), el modelo presentaba una tendencia a la "alucinación", generando ecos espurios o celdas fantasma que no existían en la realidad . Esto se atribuyó al énfasis excesivo en la intensidad de píxeles individuales.

Mejora por Aprendizaje Curricular: Tras la segunda fase de entrenamiento (hasta la Época 31), donde se priorizó la coherencia estructural (SSIM), se observó una reducción drástica de estos artefactos. Aunque persisten algunas formaciones de ruido de fondo, el análisis por capas reveló que este ruido se concentra en los niveles inferiores (< 5 km). Esto permite una solución operativa sencilla: al aplicar un filtro de elevación visualizando capas medias (> 6 km), se logra distinguir claramente las formaciones multicelulares probables del ruido de fondo.


TITAN (RAL-NCAR)
Help Level Field Zoom Cont Image Rings Maps Tracks TType Annot Past Fcast Future Time Copy Quit
dBZ
2008/01/28 04:38:49 UTC Composite Tracks to 2008/01/28 04:38:49 Tz 35 Forecast 0.5(hr)
Annott max dbz
Vertical section (RAL-NCAR)
Interp Cont Image Copy Close
VERT SECTION - dBZ - interpolated

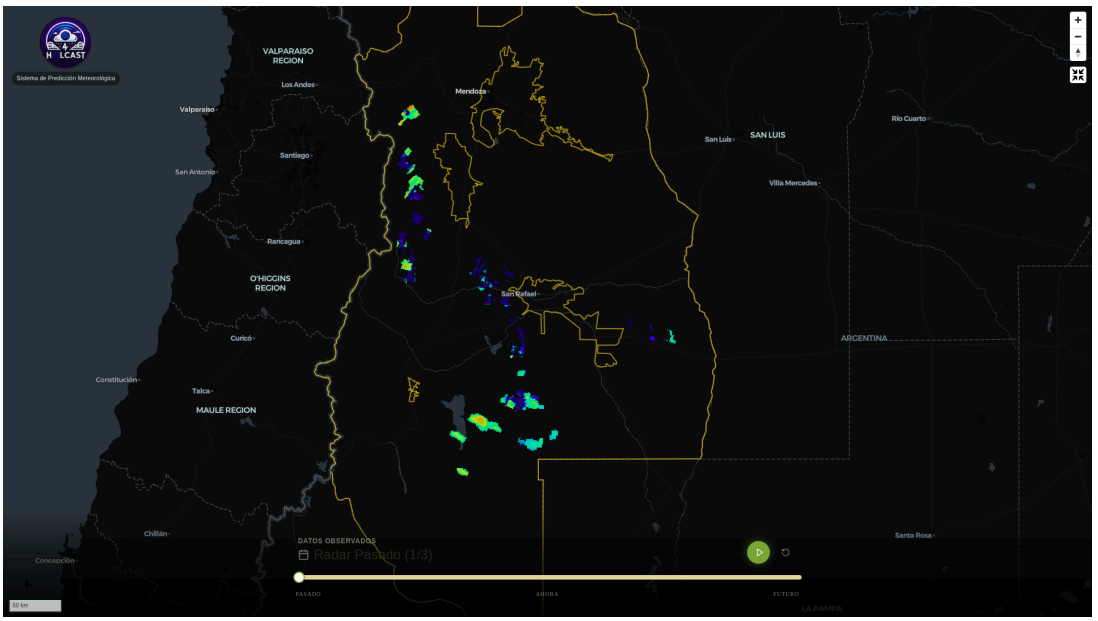

TITAN (RAL-NCAR)
Help Level Field Zoom Cont Image Rings Maps Tracks TType Annot Past Fcast Future Time Copy Quit
dBZ
2008/01/28 04:38:39 UTC 9(km) Tracks to 04:38:49 Tz 35 Forecast 0.5(hr)
Annott max dbz
Vertical section (RAL-NCAR)
Interp Cont Image Copy Close
VERT SECTION - dBZ - interpolated

5.3 Validación de la Interfaz Geoespacial (Frontend MapLibre)

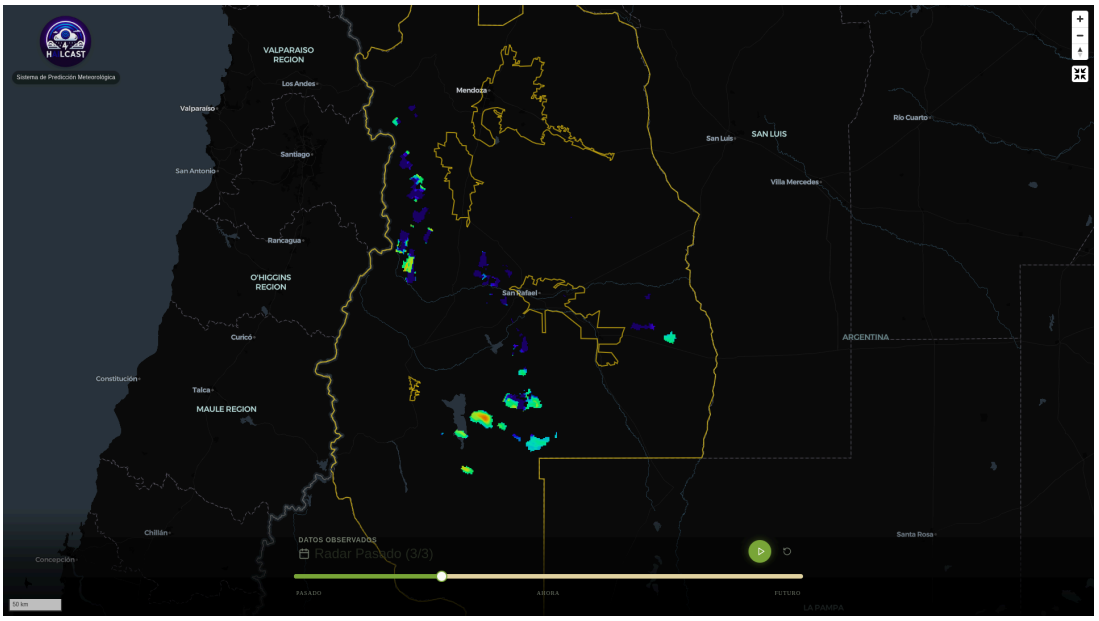
Más allá de la validación científica en TITAN, se evaluó la utilidad del sistema desde una perspectiva de **accesibilidad pública**. Dado que el objetivo final es democratizar el acceso a la información meteorológica crítica, la interfaz se diseñó para ser utilizada tanto por operadores técnicos como por la población general. Para ello se desplegó el Frontend desarrollado con **MapLibre GL JS**, aprovechando su motor gráfico basado en WebGL para garantizar fluidez en dispositivos de consumo.

Experiencia de Visualización Continua: A diferencia de los reportes estáticos tradicionales, la interfaz web implementa una línea de tiempo continua que fusiona el pasado observado con el futuro predicho. Como se evidencia en las siguientes capturas, el sistema permite reproducir la secuencia de evolución de la tormenta de manera intuitiva para un usuario no experto.

En la **Figura A**, se observa el estado "Pasado" de la secuencia. Los datos reales capturados por el radar se renderizan sobre un mapa base vectorial oscuro ("Dark Mode"). Esta elección de diseño no es meramente estética; maximiza el contraste de la reflectividad, permitiendo que cualquier ciudadano identifique claramente la ubicación de las tormentas sin distracciones visuales.



A medida que la secuencia avanza hacia el tiempo presente (**Figura B**), el usuario puede identificar la posición actual de las celdas convectivas. El uso de mapas base vectoriales permite un *zoom* fluido sin pixelado, una funcionalidad crítica para el uso público: permite a cada usuario acercarse a su propia localidad o barrio para evaluar su situación de riesgo inmediata.

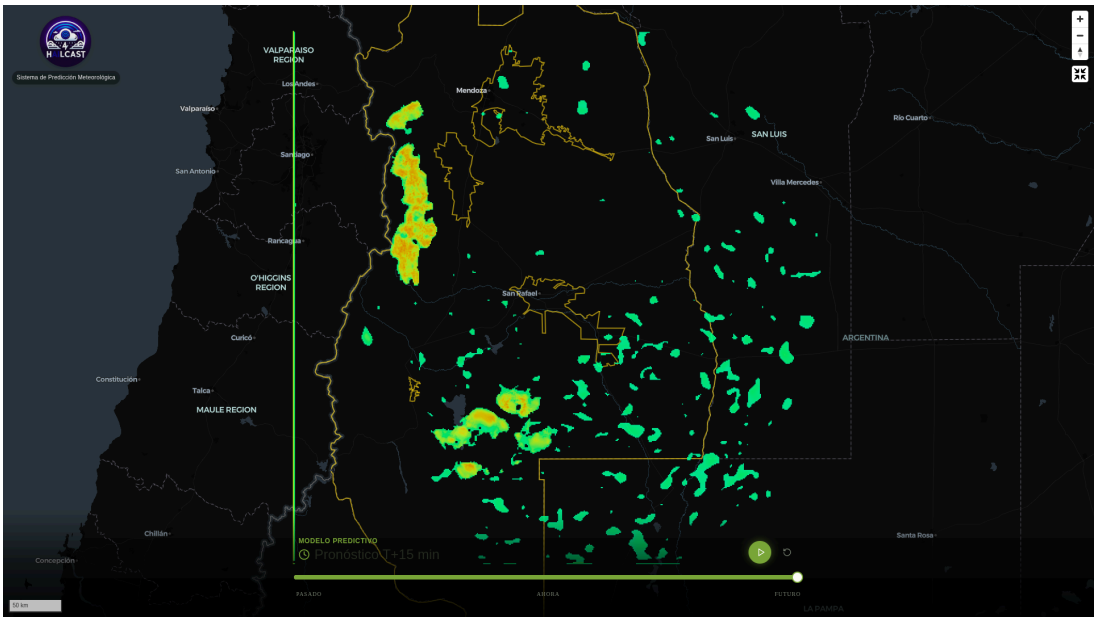


Comunicación Intuitiva del Pronóstico:

Finalmente, el sistema proyecta la inferencia del modelo hacia el futuro. En la Figura C, se visualiza el pronóstico a $T+15$ minutos. Para evitar

confusiones en el usuario final, el sistema cambia el indicador visual de la barra de tiempo (de amarillo a verde). Este código de color actúa como una señal semántica clara, advirtiendo que la información visualizada es una estimación generada por Inteligencia Artificial y no una observación directa.

La superposición de capas con transparencia variable permite responder preguntas ciudadanas inmediatas: *¿La tormenta se dirige hacia mi casa? ¿Debo resguardar mi vehículo?* Esta fluidez temporal, lograda mediante la gestión de estado en React, transforma datos científicos complejos en información accionable para la comunidad.



Esta interfaz demuestra que el modelo desarrollado no es solo un experimento numérico abstracto, sino el núcleo de un sistema de **comunicación de riesgo** moderno, funcional y accesible.

Referencias Bibliográficas

Cuomo & Chandrasekar (2021): *Use of Deep Learning for Weather Radar Nowcasting*.

Cuomo (2020): *Thesis: Machine Learning Models Applied to Storm Nowcasting*.

Fang et al. (2020): *A New Sequential Image Prediction Method Based on LSTM and DCGAN*.

Su et al. (2020): *A Convection Nowcasting Method Based on Machine Learning*.

Zheng et al. (2024): *TISE-LSTM*.

Shi et al. (2015): *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*.