

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO – CHILE



UNIVERSIDAD TÉCNICA
FEDERICO SANTA MARÍA

**“GENERACIÓN DE IMÁGENES SINTÉTICAS DE PROPAGACIÓN DE INCENDIOS
FORESTALES MEDIANTE REDES GENERATIVAS ADVERSARIALES
CONDICIONALES CON ROBUSTEZ ANTE DATOS METEOROLÓGICOS
INCOMPLETOS”**

HERNÁN ALONSO FUENTES TOBAR

**MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA**

Profesor Guía: Ernesto Vivanco

Profesor Correferente: José Luis Martí

Mayo – 2026



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: GENERACIÓN DE IMÁGENES SINTÉTICAS DE PROPAGACIÓN DE INCENDIOS FORESTALES MEDIANTE REDES GENERATIVAS ADVERSARIALES CONDICIONALES CON ROBUSTEZ ANTE DATOS METEOROLÓGICOS INCOMPLETOS

Nombre del candidato(a): Hernán Alonso Fuentes Tobar

Carrera / Grado: Ingeniería Civil Informática

Campus: San Joaquín **Departamento:** Departamento de informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Ernesto Vivanco, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por (**marcar una opción**):

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 27/05/2026

Firma: 

Estudiante o Candidato(a):

Fecha: 27/05/2026

Firma: 

DEDICATORIA

A mis padres, por su incondicional apoyo en cada etapa de este camino. Los admiro profundamente y espero retribuir todo lo que me han entregado, se merecen eso y más.

A mi abuela Luz, quien fue parte de mis primeros años, no me cabe duda de lo orgullosa que estaría del niño que tanto mimó.

A Mike, quien me acompañó a lo largo de todo este proceso y sin quererlo fue un apoyo fundamental, siempre estarás presente pequeño.

RESUMEN

Resumen— Se desarrolló un sistema de generación de imágenes sintéticas de propagación de incendios forestales mediante una cGAN, entrenada sobre mapas de calor Monte Carlo producidos por un autómata celular (AC) calibrado con datos históricos de la Corporación Nacional Forestal (CONAF). El proceso abarcó 48 experimentos en tres fases: exploratoria (Runs 1–14), corrección del simulador (Runs 15–20) y condicionamiento explícito (Runs 21–48), con clasificador auxiliar AC-GAN a partir del Run 44. El modelo definitivo (Run 48) alcanza $FID = 282,97$, $SSIM = 0,526 \pm 0,180$ y condicionamiento correcto en **8/8 direcciones de viento** con error medio $5,1^\circ$ (condiciones sintéticas) y $12,0^\circ \pm 10,0^\circ$ (*val set* real). El sistema incorpora *missingness flags* y *condition masking* para operar con registros incompletos: el SSIM se mantiene por encima de 0,50 hasta el 70% de variables enmascaradas. Este es el primer sistema generativo calibrado sobre incendios chilenos con condicionamiento direccional físico verificado.

Palabras Clave— Redes Generativas Adversariales; Incendios Forestales; Autómata Celular; Generación de Imágenes Sintéticas; Robustez ante Datos Incompletos.

ABSTRACT

Abstract— A conditional GAN system was developed to generate synthetic wildfire propagation images conditioned on meteorological, vegetation, and topographic variables from CONAF historical records. Training uses probabilistic heat maps produced by a calibrated Monte Carlo Cellular Automaton (CA) simulator. Development spanned 48 experiments in three phases: exploratory (Runs 1–14), simulator calibration (Runs 15–20), and explicit conditioning (Runs 21–48), introducing AC-GAN auxiliary classifier from Run 44 onward. The final model (Run 48) achieves FID = 282.97, SSIM = 0.526 ± 0.180 , and correct conditioning on **8/8 wind directions** with mean error 5.1° (synthetic conditions) and $12.0^\circ \pm 10.0^\circ$ (real *val-set*). The system incorporates *missingness flags* and *condition masking* to operate on incomplete records: SSIM stays above 0.50 up to 70% masking. This is the first generative system calibrated on Chilean wildfires with verified physical directional conditioning.

Keywords— Generative Adversarial Networks; Wildfires; Cellular Automaton; Synthetic Image Generation; Robustness to Incomplete Data.

GLOSARIO

AC / CA	Autómata Celular (<i>Cellular Automaton</i>). Modelo de simulación basado en una grilla de celdas que evolucionan en pasos discretos de tiempo según reglas locales de vecindad.
cGAN	Red Generativa Adversarial Condicional (<i>Conditional GAN</i>). Variante de GAN en que el generador y el discriminador reciben información de condicionamiento para guiar la síntesis hacia clases o atributos específicos.
CONAF	Corporación Nacional Forestal. Organismo del Estado de Chile encargado de la administración de áreas silvestres y la prevención y control de incendios forestales.
DCGAN	<i>Deep Convolutional GAN</i> . Arquitectura de GAN que emplea capas convolucionales transpuestas en el generador y convoluciones estándar en el discriminador.
DiffAugment	Aumentación diferenciable. Técnica que aplica transformaciones de imagen directamente en el grafo computacional, permitiendo que los gradientes fluyan a través de las aumentaciones durante el entrenamiento.
EMA	Promedio Exponencial de Pesos (<i>Exponential Moving Average</i>). Técnica que mantiene una copia del generador con pesos promediados en el tiempo, produciendo muestras de mayor calidad y menor varianza.
FID	Distancia de Fréchet de Inception (<i>Fréchet Inception Distance</i>). Métrica que cuantifica la similitud entre la distribución de imágenes reales y generadas en el espacio de features de InceptionV3. Valores más bajos son mejores.
GAN	Red Generativa Adversarial (<i>Generative Adversarial Network</i>). Arquitectura de aprendizaje profundo compuesta por un generador G y un discriminador D que compiten en un juego minimax: G aprende a generar datos realistas mientras D aprende a distinguirlos de los reales.
GPU	Unidad de Procesamiento Gráfico (<i>Graphics Processing Unit</i>). Procesador especializado en operaciones matriciales paralelas, ampliamente utilizado para entrenar redes neuronales profundas.
Hinge Loss	Función de pérdida de bisagra. Variante de pérdida para GANs que acota los scores del discriminador en $[-1, 1]$, produciendo gradientes más estables que la pérdida binaria cruzada.
Heatmap MC	Mapa de calor Monte Carlo. Imagen que representa la probabilidad acumulada de propagación del fuego, obtenida promediando múltiples simulaciones de autómata celular con perturbaciones estocásticas.

Mode Seeking Loss

Pérdida de búsqueda de modos. Regularización que penaliza al generador cuando produce imágenes similares para entradas latentes distintas, incentivando la diversidad de las muestras generadas.

R1

Penalización de gradiente R1. Regularización del discriminador que penaliza gradientes grandes sobre los datos reales, mejorando la estabilidad del entrenamiento frente a WGAN-GP.

SSIM

Índice de Similitud Estructural (*Structural Similarity Index*). Métrica de calidad de imagen que evalúa luminancia, contraste y estructura. En este trabajo se usa en versión condicionada: las imágenes generadas se comparan con las reales de condiciones similares.

veg_profile

Perfil de vegetación. Vector de 10 dimensiones que codifica la proporción de cada tipo de vegetación (arbolado nativo, eucaliptal, matorral, pastizal, etc.) en el área del incendio, usado como señal adicional de condicionamiento.

z-score

Puntuación estándar. Transformación que expresa un valor en número de desviaciones estándar respecto a la media de la distribución: $z = (x - \mu) / \sigma$.

ÍNDICE DE CONTENIDOS

DEDICATORIA	iii
RESUMEN	iv
ABSTRACT	v
GLOSARIO	vi
ÍNDICE DE FIGURAS	xiv
ÍNDICE DE TABLAS	xvii
CAPÍTULO 1: Introducción	1
1.1 Contexto y motivación	1
1.2 Planteamiento del problema	1
1.3 Objetivos	2
1.4 Alcances y limitaciones	3
1.5 Estructura del documento	3
CAPÍTULO 2: Marco Teórico	4
2.1 Redes Generativas Adversariales	4
2.1.1 Formulación original	4
2.1.2 DCGAN: Redes Convolucionales Profundas	4
2.1.3 GANs Condicionales	5
2.1.4 Condicionamiento por Proyección	5
2.2 Funciones de Pérdida Avanzadas	5
2.2.1 Hinge Loss	5

2.2.2	Regularización R1	6
2.2.3	Feature Matching Loss	6
2.2.4	Mode Seeking Loss	6
2.3	Técnicas de Estabilización	7
2.3.1	Normalización Espectral	7
2.3.2	Desviación Estándar de Minibatch	7
2.3.3	DiffAugment	7
2.3.4	Promedio Exponencial de Pesos (EMA)	8
2.4	Autómatas Celulares para Propagación de Incendios	8
2.4.1	Fundamentos	8
2.4.2	Probabilidad de Propagación	8
2.4.3	Heatmaps Monte Carlo	9
2.5	Métricas de Evaluación	9
2.5.1	Distancia de Fréchet de Inception (FID)	9
2.5.2	SSIM Condicionada	10
CAPÍTULO 3: Datos y Preprocesamiento		10
3.1	Dataset CONAF	10
3.2	Preprocesamiento del vector de condición	11
3.2.1	Imputación	11
3.2.2	Normalización de variables continuas	11
3.2.3	Codificación de variables categóricas	11
3.2.4	Layout del vector de condición	12
3.3	Perfil de vegetación	12
3.4	División train/val	12

3.5	Aumentación de condiciones	13
CAPÍTULO 4: Sistema de autómeta celular		13
4.1	Arquitectura del simulador	13
4.2	Generación de terreno sintético	13
4.3	Heatmaps Monte Carlo con horizontes temporales	14
4.4	Calibración de pasos por tamaño de incendio	14
4.4.1	Problema identificado	14
4.4.2	Corrección implementada	14
4.5	Normalización de parámetros meteorológicos	15
4.5.1	Problema identificado	15
4.5.2	Corrección implementada	15
CAPÍTULO 5: Arquitectura del Modelo Generativo		16
5.1	Visión general	16
5.2	Módulo de embedding de condición	16
5.3	Generador	16
5.4	Discriminador	17
CAPÍTULO 6: Proceso de Entrenamiento		18
6.1	Configuración general	18
6.2	Programación del learning rate	19
6.3	BestCheckpointCallback	19
6.4	Monitor de entrenamiento	19
6.5	Historial de experimentos	20
6.5.1	Run 1: Baseline	22

6.5.2	Run 2: Hinge loss y conditioning completo	22
6.5.3	Run 3: Minibatch stddev y eliminación de InstanceNorm (mejor FID con protocolo v0, artefacto)	22
6.5.4	Run 4: $\gamma_{R1} = 0,5$	23
6.5.5	Run 5: Pérdida de diversidad con pdist	23
6.5.6	Run 6: Mode Seeking Loss con $\lambda = 1,0$	23
6.5.7	Run 7: Dataset calibrado + Mode Seeking Loss $\lambda = 0,1$ + Cosine LR . .	23
6.5.8	Run 8: Perfil de vegetación como condicionamiento adicional	24
6.5.9	Resultados del Run 8	25
6.5.10	Run 9: diversity_weight $0,1 \rightarrow 0,2$	27
6.5.11	Run 10: Two Time-Scale Update Rule (TTUR)	27
6.5.12	Run 11: TTUR + dataset calibrado reducido (~ 6.000 imágenes)	29
6.5.13	Run 12: TTUR + dataset completo + diversity_weight = $0,15$	31
6.5.14	Run 13: Arquitectura escalada + correcciones de implementación . .	31
6.5.15	Run 14: Corrección de bugs de augmentación (flips + condition masking)	33
6.5.16	Validación estadística de Run 10	34
6.6	Fase de validación: corrección del simulador y reentrenamiento	35
6.6.1	Validación cualitativa: identificación de la limitación en el CA	35
6.6.2	Corrección de parámetros del simulador	36
6.6.3	Run 15: primer reentrenamiento con simulador calibrado (resultado negativo)	37
6.6.4	Run 16: segundo reentrenamiento con simulador calibrado (resultado negativo)	38
6.6.5	Run 17: mejor resultado exploratorio con simulador calibrado	39
6.6.6	Runs 18–20: corrección de convención meteorológica	40

6.7	Fase de condicionamiento explícito (Runs 21–48)	42
6.7.1	Corrección de infraestructura y datos (Runs 21–33)	42
6.7.2	Pérdida de centroide direccional (Runs 34–43)	43
6.7.3	AC-GAN: clasificador auxiliar de dirección (Runs 44–48)	45
CAPÍTULO 7: Resultados y Análisis		48
7.1	Resultados cuantitativos	48
7.2	Evolución del entrenamiento	50
7.3	Análisis cualitativo	50
7.4	Robustez ante datos faltantes	53
7.5	Autocorrelación espacial: Índice de Moran	55
7.6	Aplicación a casos reales	56
7.6.1	Incendio El Boldo (376 ha) — coherencia direccional sobre datos reales	56
7.6.2	Incendio Las Máquinas (159.812 ha) — limitación de escala documentada	58
7.6.3	Efecto de dirección de viento faltante	60
7.7	Evaluación del modelo definitivo Run 48	62
7.7.1	Validación del condicionamiento direccional sobre incendios reales .	64
7.8	Discusión de las métricas	65
CAPÍTULO 8: Discusión y Trabajo Futuro		65
8.1	Hallazgos principales	65
8.2	Limitaciones	66
8.3	Trabajo futuro	68
CAPÍTULO 9: CONCLUSIONES		69
9.1	Revisión de objetivos específicos	70

9.2	Contribuciones principales	71
9.3	Resultados y trade-offs del modelo definitivo	72
9.4	Lecciones aprendidas	73
9.5	Impacto y aplicabilidad	74
9.6	Trabajo futuro	74
REFERENCIAS BIBLIOGRÁFICAS		75

ÍNDICE DE FIGURAS

- 1 *Sweep* de dirección de viento con el simulador calibrado (velocidad media Maule, 4,6 km/h). Cada panel muestra el *heatmap* Monte Carlo promedio ($n = 15$ runs). El patrón en “mariposa” confirma que el simulador calibrado codifica correctamente la señal direccional. Fuente: Elaboración propia. . . . 37
- 2 *Sweep* de dirección de viento — Run 17 (mejor resultado exploratorio con simulador calibrado). Nueve imágenes generadas con el mismo vector latente z y distinta *Dir_viento*. El fuego se extiende hacia el lado *downwind*. Run 20 corrigió formalmente la convención meteorológica del simulador CA, sin afectar las métricas de Run 17. Fuente: Elaboración propia. 39
- 3 *Sweep* de dirección de viento — Run 20 (convención meteorológica corregida). Nueve condiciones: N, NE, E, SE, S, SW, W, NW, Calma. El fuego se extiende hacia el lado *downwind* en cada dirección; las condiciones opuestas producen patrones con orientación invertida. Fuente: Elaboración propia. . . 42
- 4 Comparación real vs. generado — **Run 3** (FID = 174,36, época 45). Modelo inicial sin perfil de vegetación ni condicionamiento direccional. **Fila superior:** *heatmaps* reales del simulador CA. **Fila inferior:** imágenes generadas con artefactos de color (mosaico verde/amarillo/negro) y núcleos sobreexpuestos (blanco), resultado del desequilibrio GAN sin TTUR ni perfil de vegetación. Sirve como punto de partida para comparar con Run 10 (Figura 5) y Run 48 (Figura 6). Fuente: Elaboración propia. 51
- 5 Comparación real vs. generado — **Run 10** (mejor modelo fase exploratoria, FID = $168,47 \pm 0,47$, SSIM = 0,4634, TTUR + perfil de vegetación, época 85). Mayor coherencia morfológica respecto a Run 3, pero aún sin condicionamiento direccional explícito. Fuente: Elaboración propia. 51

- 6 Comparación real vs. generado — **Run 48** (modelo definitivo, FID = 282,97, SSIM = $0,526 \pm 0,180$, AC-GAN, época 70). Un caso representativo por cada dirección de viento (N, NE, E, SE, S, SW, W, NW), usando el mismo vector de condición para real y generado. Ambas filas se visualizan mediante extracción de intensidad de fuego con colormap *inferno* para comparación equitativa. Fila superior: simulación CA individual (horizonte $t = 80$, una realización Monte Carlo). Fila inferior: imagen generada por la cGAN con idéntico vector de condición. La diferencia de apariencia entre filas responde a causas estructurales y no a errores del modelo: (i) cada imagen CA es una *realización estocástica individual* — celdas quemadas de forma binaria con ramificaciones aleatorias —, mientras que la cGAN genera la *tendencia central* de la distribución aprendida, resultado de promedio implícito sobre miles de realizaciones, lo que produce manchas más difusas con gradientes suaves (*regresión a la media* típica de modelos generativos condicionales); (ii) la función objetivo AC-GAN optimiza la dirección del centroide de propagación, no la compacidad espacial ni la fidelidad píxel a píxel. La estructura condicional —posición y orientación de la mancha según el viento— es capturada correctamente en las 8 direcciones. Fuente: Elaboración propia. 52

- 7 Progresión temporal del autómata celular Monte Carlo para un registro representativo. Cada panel corresponde a un horizonte temporal distinto, mostrando la expansión gradual del frente de fuego desde el punto de ignición central. Fuente: Elaboración propia. 53

- 8 Curvas de robustez ante datos faltantes (Run 48, época 70). **Izquierda:** FID permanece en el rango 284–294 para todos los niveles de enmascaramiento. **Derecha:** SSIM parte en $\approx 0,52$ y cae gradualmente, sin caída abrupta, manteniéndose $> 0,50$ hasta el 70% de enmascaramiento y alcanzando 0,455 con 90% de condición oculta. La degradación gradual confirma que el modelo desarrolló robustez ante información parcial gracias al *conditioning masking* del entrenamiento. Fuente: Elaboración propia. 54

- 9 Distribución del índice de Moran I para el CA real y las imágenes generadas (Run 48, $n = 200$). La distribución real se concentra en $I > 0,9$; la generada es amplia y plana ($\mu = 0,6355$). El $|\Delta I| = 0,2690$ refleja que el objetivo AC-GAN prioriza desplazamiento del centroide direccional sin optimizar coherencia espacial local (véase Sección 8.2). Fuente: Elaboración propia. . . . 56

- 10 Condicionamiento direccional sobre caso real — **El Bordo**, Cauquenes, Maule (02/01/2019, 376 ha, viento Sur, 28,0°C, HR 31 %). **Panel izquierdo:** mosaico de las 6 realizaciones individuales, ilustrando la variabilidad estocástica del generador. **Panel derecho:** promedio de las 6 realizaciones (*heatmap*); la flecha cian indica la dirección del centroide de propagación ponderado por intensidad, apuntando hacia el norte, coherente con la dirección *downwind* esperada para viento Sur. Fuente: Elaboración propia. 57

- 11 Salida del modelo sobre caso real extremo — **Las Máquinas**, Cauquenes (20/01/2017, 159.812 ha). Seis realizaciones del modelo Run 48 (época 70) condicionadas en las variables meteorológicas reales CONAF. *Colormap inferno*: rojo/naranja = alta probabilidad de incendio, negro = baja. La saturación uniforme es el comportamiento esperado para un megaincendio fuera del rango de entrenamiento (≤ 5.000 ha). No se trata de validación geoespacial: el área quemada real ($\text{lon} \approx -72,80^\circ\text{O}$) no se solapa con el grid GAN ($\text{lon} \approx -72,28^\circ\text{O}$). Fuente: Elaboración propia. 59

- 12 Incendio **SAN JOSE** (12 ha, viento Sur 9,8 km/h) — efecto de la dirección de viento faltante. Cuatro realizaciones del generador Run 48 (época 70) para cada condición; imágenes raw 64×64. **Columna izquierda**: Dir_viento=S → mancha quemada elongada hacia el norte (arriba), coherente con propagación *downwind*. La flecha cian indica la dirección de propagación esperada. **Columna derecha**: Dir_viento=Missing, resto de condiciones idéntico → patrón naranja centrado, sin elongación direccional. Ambas columnas usan el mismo seed aleatorio para comparación directa. Fuente: Elaboración propia. 61

- 13 Validación direccional cuantitativa — **Run 48** (época 70, condiciones reales *val set*). **Izquierda**: diagrama polar; cada línea de color representa una dirección de viento; la línea gris discontinua es el eje *downwind* esperado. Error medio $12,0^\circ \pm 10,0^\circ$; todas las direcciones dentro del umbral de 45° . **Derecha**: *heatmaps* promedio por dirección, confirmando la elongación geométrica coherente con la convención meteorológica. Fuente: Elaboración propia. 63

- 14 Validación del condicionamiento direccional sobre incendios reales (Run 48, época 70). Grilla 2×4: cada panel corresponde a un incendio real con una dirección de viento distinta (N, NE, E, SE en fila superior; S, SW, W, NW en fila inferior), mostrando una realización por dirección. La flecha cian indica la dirección de propagación esperada (*downwind*). En las ocho direcciones la mancha quemada generada coincide con la dirección señalada, evidenciando que el condicionamiento funciona sobre vectores de condición reales y no solo en evaluaciones sintéticas. Fuente: Elaboración propia. 64

ÍNDICE DE TABLAS

1	Variables del dataset CONAF utilizadas en el pipeline. Fuente: Elaboración propia.	10
2	Mapeo de columnas Excel al perfil de vegetación (10 dims). Fuente: Elaboración propia.	12
3	Resumen de la arquitectura del modelo. Fuente: Elaboración propia.	16
4	Arquitectura del generador. Fuente: Elaboración propia.	17
5	Arquitectura del discriminador. Fuente: Elaboración propia.	18
6	Hiperparámetros de entrenamiento. Fuente: Elaboración propia.	19
7	Resultados de la fase exploratoria (Runs 1–14), entrenados con el simulador isotrópico. Runs 1–6: protocolo v0 (<i>val set</i> original). Runs 7–14: protocolo v2 (condiciones alineadas, 2.400 imágenes <i>val</i>). Run 13 no re-evaluable con arquitectura v1. Estos runs son comparables entre sí internamente; la fase de validación (Runs 15–20) con simulador calibrado se presenta en la sección siguiente. Fuente: Elaboración propia.	21
8	Resumen del cambio arquitectónico del Run 8. Fuente: Elaboración propia. .	25
9	Evolución de métricas del monitor durante Run 8. Fuente: Elaboración propia.	26
10	Resultados FID/SSIM del Run 8. Fuente: Elaboración propia.	26
11	Resultados FID/SSIM del Run 9. Fuente: Elaboración propia.	27
12	Resultados FID/SSIM del Run 10 (evaluación interna durante entrenamiento, <i>val set</i> local 60 imágenes con bug de n_{steps}). Los valores corregidos con <i>val set</i> calibrado de 72 imágenes se reportan en la Tabla 13: FID = 209,91, SSIM = 0,4462. Los valores definitivos del protocolo v2 (2.400 imágenes, 5 semillas) se reportan en la Sección 6.5.16: FID = $168,47 \pm 0,47$, SSIM = $0,4634 \pm 0,0023$. Fuente: Elaboración propia.	28
13	Re-evaluación con <i>val set</i> calibrado (72 imágenes). Fuente: Elaboración propia.	29
14	Resultados de Run 10 en 5 evaluaciones con distinto ruido z . Fuente: Elaboración propia.	35
15	Parámetros del CA antes y después de la corrección. Fuente: Elaboración propia.	36

16	Métrica de centroide direccional — Run 20. Error angular entre centroide medido y dirección <i>downwind</i> esperada. Fuente: Elaboración propia.	41
17	Barrido de hiperparámetros de la pérdida de centroide (Runs 37–43). El plateau en $\approx -0,062$ fue consistente con todas las variantes. WRS: WeightedRandomSampler; nc: n_critic. Fuente: Elaboración propia.	44
18	Evolución del condicionamiento en Run 48. std(cx) y std(cy): desviación estándar del centroide entre las 8 direcciones con z fijo — valores próximos a 0 indican ausencia de respuesta; valores $> 0,3$ indican desplazamiento geométrico sustancial. Fuente: Elaboración propia.	47
19	Resumen del test geométrico — Run 48 (época 70). Se presentan dos evaluaciones: condiciones sintéticas fijas (diag_05, z fijo, 16 muestras/dirección) y condiciones reales del <i>val set</i> (script 09, 15 muestras/dirección). std(cx): desviación estándar del centroide entre las 8 direcciones. Fuente: Elaboración propia.	48
20	Resumen de modelos representativos del proceso experimental. FID de Runs 17, 20 y 48 evaluado con protocolo v2 (2.400 imágenes, <i>val set</i> calibrado). Dirs OK: direcciones con error angular $< 45^\circ$ en test geométrico (diag_05). Fuente: Elaboración propia.	49
21	Comparación de FID en distintos dominios y modelos. Fuente: Elaboración propia.	50
22	Índice de Moran I — Real CA vs. Generado ($n = 200$). Run 17 (sin restricción direccional) preserva coherencia espacial; Run 48 (AC-GAN, modelo definitivo) muestra regresión que refleja el trade-off entre condicionamiento direccional y compacidad espacial. Fuente: Elaboración propia.	55

CAPÍTULO 1

Introducción

1.1 Contexto y motivación

Pocos países enfrentan la amenaza de los incendios forestales con la regularidad que lo hace Chile. El promedio del último quinquenio registrado por la Corporación Nacional Forestal (CONAF) supera los 6.500 incendios anuales y las 150.000 hectáreas afectadas [21]; la temporada 2016–2017, con más de 500.000 hectáreas quemadas, sigue siendo la más devastadora de la historia del país. La temporada 2025–2026 acumula a la fecha 4.933 incendios y 78.078 hectáreas afectadas a nivel nacional, con focos de especial intensidad en las regiones del Biobío (32.523 ha) y Maule (11.043 ha) [22].

La zona central concentra la mayor parte de la población, las plantaciones forestales de pino y eucalipto y la actividad agrícola, lo que la convierte también en la región con mayor frecuencia e intensidad de siniestros. El escenario se agrava: el calentamiento progresivo, la reducción de precipitaciones y la extensión de las sequías apuntan a temporadas aún más severas en las décadas venideras [23, 24].

Ante un incendio activo, saber hacia dónde se propagará el fuego en las próximas horas es decisivo para planificar evacuaciones, posicionar brigadas y diseñar cortafuegos. Los simuladores físicos más usados, FARSITE¹ [27] y FlamMap, requieren datos de campo detallados, ajuste experto de parámetros y tiempos de cómputo que dificultan generar decenas de escenarios alternativos en paralelo. Los modelos generativos de aprendizaje profundo, una vez entrenados, pueden producir imágenes de propagación en milisegundos condicionadas a variables ambientales, abriendo la posibilidad de explorar el espacio de escenarios de forma ágil.

1.2 Planteamiento del problema

Hoy no existe un modelo generativo calibrado sobre incendios chilenos que condicione la síntesis a datos meteorológicos y vegetacionales reales de registros históricos nacionales. Los simuladores disponibles para Chile son deterministas y no generativos; los modelos generativos de la literatura internacional no están ajustados a las condiciones ecológicas y climáticas locales.

Este trabajo se focaliza en la región del Maule, una de las más afectadas históricamente: con un promedio quinquenal de 812 incendios anuales y 10.669 ha quemadas, y una temporada

¹FARSITE (*Fire Area Simulator*) es un sistema de simulación de área de incendios desarrollado por el Servicio Forestal de EE.UU. (USDA Forest Service). Requiere como inputs modelos de combustible, humedad, viento y topografía de alta resolución, además de calibración experta de parámetros para cada ecosistema específico.

2025–2026 que ya acumula 11.043 ha (+117% respecto a la anterior) [22], el Maule concentra las condiciones de vegetación, topografía y comportamiento del viento que hacen que un modelo de propagación entrenado allí tenga valor práctico real. Los registros de CONAF para esta región constituyen la fuente de datos del trabajo.

El problema central de este trabajo es aprender la distribución condicional $p(\mathbf{x} \mid \mathbf{c})$ a partir de simulaciones de autómata celular, donde $\mathbf{x} \in \mathbb{R}^{3 \times 64 \times 64}$ es un mapa de calor de propagación y $\mathbf{c} \in \mathbb{R}^{32}$ codifica temperatura, humedad, velocidad del viento, superficie total, exposición topográfica, dirección del viento y topografía del registro correspondiente. La composición vegetal se entrega como vector separado $\mathbf{v} \in \mathbb{R}^{10}$.

Un segundo problema, más práctico, surge del estado real de los datos: los registros de CONAF tienen campos faltantes de forma habitual. Dirección de viento, temperatura, humedad y exposición topográfica aparecen como *Missing* en una fracción no menor de los incendios históricos. Si el modelo no está preparado para operar con esta información parcial, simplemente no puede usarse en producción con los datos existentes.

Respecto a por qué usar un simulador en vez de imágenes satelitales: no hay alternativa viable. Construir un dataset directamente desde perímetros reales exigiría contar con miles de incendios que tuvieran perímetros georreferenciados a alta frecuencia temporal, condiciones meteorológicas completas sincronizadas con cada etapa de propagación y escala espacial homogénea. Ese tipo de datos no existe en Chile con la densidad mínima para entrenar un modelo generativo. El autómata celular es la solución estándar cuando los datos observacionales son escasos en simulación de riesgos naturales [26]: permite generar muestras a demanda con condiciones controladas y consistencia física garantizada.

1.3 Objetivos

Objetivo general

Diseñar, implementar y evaluar un sistema de generación de imágenes sintéticas de propagación de incendios forestales mediante una Red Generativa Adversarial Condicional (cGAN), entrenada sobre simulaciones de autómata celular con Monte Carlo condicionadas a datos históricos de CONAF.

Objetivos específicos

1. Construir un pipeline completo de preprocesamiento de datos que transforme los registros históricos de incendios de CONAF en vectores de condición y perfiles de vegetación utilizables por el modelo generativo.
2. Implementar un simulador de propagación de incendios basado en autómata celular

con Monte Carlo que genere mapas de calor probabilísticos calibrados según el tamaño real del incendio registrado.

3. Diseñar y entrenar una cGAN con función de pérdida compuesta (*hinge loss*, regularización R1, *feature matching* y *mode seeking*) que genere imágenes de propagación de alta diversidad y fidelidad visual.
4. Evaluar la calidad de las imágenes generadas mediante la Distancia de Fréchet de Inception (FID) y similitud estructural condicionada (SSIM), y documentar el proceso iterativo de mejora del modelo.
5. Identificar y corregir inconsistencias en el pipeline de datos que afecten la calidad del entrenamiento.
6. Diseñar e implementar un mecanismo de robustez ante datos meteorológicos incompletos mediante *missingness flags* explícitos y *condition masking* durante el entrenamiento, y evaluar cuantitativamente la degradación del SSIM en función del porcentaje de variables ausentes.

1.4 Alcances y limitaciones

El trabajo cubre el pipeline completo: desde la carga de los registros Excel de CONAF hasta la evaluación cuantitativa del modelo generativo. Las imágenes de salida son de 64×64 píxeles en formato de mapa de calor probabilístico, dimensión compatible con los grids del autómata celular.

Tres limitaciones deben tenerse en cuenta. La más importante es que el modelo aprende la distribución del simulador CA, no la del incendio real; la correspondencia con la física del fuego depende enteramente de cuán bien esté calibrado el CA, y esa calibración contra datos reales queda como trabajo futuro. Las otras dos son de naturaleza técnica: la resolución de 64×64 es baja para representar incendios de gran extensión, y el FID calculado con InceptionV3/ImageNet puede no ser representativo de la calidad perceptual en este dominio específico. Ninguna de estas tres limitaciones es evitable dentro del alcance de este trabajo.

1.5 Estructura del documento

El **Capítulo 2** cubre el marco teórico: GANs, variantes condicionales, funciones de pérdida y métricas de evaluación. El **Capítulo 3** describe los datos y el preprocesamiento. El **Capítulo 4** detalla el autómata celular y la generación del dataset. La arquitectura del modelo se presenta en el **Capítulo 5**. El **Capítulo 6** narra el proceso experimental completo (48 runs). Los resultados y su análisis están en el **Capítulo 7**. El **Capítulo 8** discute las limitaciones y el trabajo futuro, y el **Capítulo 9** cierra con las conclusiones.

CAPÍTULO 2

Marco Teórico

2.1 Redes Generativas Adversariales

2.1.1 Formulación original

Las Redes Generativas Adversariales (GANs, por sus siglas en inglés) fueron introducidas por Goodfellow et al. en 2014 [1]. Una GAN consiste en dos redes neuronales que se entrenan simultáneamente en un juego adversarial: un *generador* G y un *discriminador* D .

El generador $G : \mathbb{R}^{n_z} \rightarrow \mathbb{R}^d$ mapea un vector de ruido latente $\mathbf{z} \sim p_z(\mathbf{z})$ (típicamente una distribución normal estándar) al espacio de datos. El discriminador $D : \mathbb{R}^d \rightarrow [0, 1]$ estima la probabilidad de que una muestra dada provenga de la distribución de datos real p_{data} en lugar del generador.

El objetivo de entrenamiento original se expresa como el juego minimax:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_z} [\log(1 - D(G(\mathbf{z})))] \quad (1)$$

En el óptimo teórico, G aprende la distribución de datos real ($p_G = p_{\text{data}}$) y $D(\mathbf{x}) = 1/2$ para todo $\mathbf{x}$. Sin embargo, en la práctica el entrenamiento de GANs es notoriamente inestable, sufriendo de problemas como desaparición de gradientes (*vanishing gradients*), oscilaciones y colapso de modos (*mode collapse*), en el que G aprende a generar una variedad limitada de muestras que engañan a D [1, 7].

2.1.2 DCGAN: Redes Convolucionales Profundas

Las DCGAN (Deep Convolutional GAN) [3] introdujeron el uso sistemático de capas convolucionales en ambas redes, eliminando las capas completamente conectadas en las secciones intermedias. El discriminador usa convoluciones con *stride* en lugar de *pooling*, y el generador usa convoluciones transpuestas (*ConvTranspose2d*) para el *upsampling*. *Batch Normalization* [16] en ambas redes estabiliza el entrenamiento, y las activaciones son LeakyReLU en el discriminador y ReLU en el generador (excepto la salida, con Tanh).

2.1.3 GANs Condicionales

Las GANs Condicionales (cGANs) [2] extienden el modelo original incorporando información adicional $\mathbf{c}$ (condición) tanto al generador como al discriminador. El objetivo se modifica a:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [\log D(\mathbf{x} \mid \mathbf{c})] + \mathbb{E}_{\mathbf{z} \sim p_z} [\log(1 - D(G(\mathbf{z} \mid \mathbf{c})))] \quad (2)$$

La condición $\mathbf{c}$ puede ser una etiqueta de clase, un texto, o —como en este trabajo— un vector continuo de alta dimensión que codifica múltiples variables ambientales.

2.1.4 Condicionamiento por Proyección

Miyato y Koyama [4] propusieron el *discriminador de proyección*, que integra la condición de forma más natural que la simple concatenación. La salida del discriminador se define como:

$$D(\mathbf{x}, \mathbf{c}) = \psi(\phi(\mathbf{x})) + \langle \phi(\mathbf{x}), \mathbf{V}\mathbf{c} \rangle \quad (3)$$

donde $\phi(\mathbf{x})$ son las activaciones internas del discriminador, ψ es el componente incondicional, y $\mathbf{V}$ es una matriz de proyección aprendida. El producto interno entre las activaciones de la imagen y el embedding de la condición penaliza naturalmente la incoherencia imagen-condición.

2.2 Funciones de Pérdida Avanzadas

2.2.1 Hinge Loss

La pérdida *hinge* para GANs [14] reemplaza la BCE original y produce scores naturalmente acotados, lo que evita que el discriminador se vuelva excesivamente dominante. Se define como:

$$\mathcal{L}_D = \mathbb{E}[\max(0, 1 - D(\mathbf{x}_{\text{real}}))] + \mathbb{E}[\max(0, 1 + D(G(\mathbf{z})))] \quad (4)$$

$$\mathcal{L}_G = -\mathbb{E}[D(G(\mathbf{z}))] \quad (5)$$

El discriminador es penalizado cuando el score de los datos reales cae por debajo de 1, o cuando el score de los datos falsos sube por encima de -1 . Esta formulación produce gradientes más estables y evita la saturación típica de la BCE.

2.2.2 Regularización R1

Mescheder et al. [6] demostraron que muchas instancias de entrenamiento de GANs convergen sólo cuando se aplica regularización de gradiente. La penalización R1 actúa exclusivamente sobre los datos reales:

$$\mathcal{R}_1 = \frac{\gamma}{2} \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [\|\nabla_{\mathbf{x}} D(\mathbf{x})\|^2] \quad (6)$$

con $\gamma = 10,0$ en este trabajo. La pérdida total del discriminador es $\mathcal{L}_D + \mathcal{R}_1$. A diferencia de WGAN-GP [13], que aplica penalización sobre puntos interpolados entre datos reales y falsos, R1 actúa sólo sobre datos reales, lo que Mescheder et al. demostraron como condición suficiente de convergencia local del entrenamiento adversarial.

2.2.3 Feature Matching Loss

Salimans et al. [7] propusieron la pérdida de coincidencia de características (*feature matching*) como técnica de estabilización del generador. En lugar de optimizar directamente el score del discriminador, el generador minimiza la diferencia entre estadísticas de activaciones intermedias de D para datos reales y falsos:

$$\mathcal{L}_{\text{FM}} = \sum_{i=1}^L \frac{1}{N_i} \|\mathbb{E}_{\mathbf{x}}[f_i(\mathbf{x})] - \mathbb{E}_{\mathbf{z}}[f_i(G(\mathbf{z}))]\|_1 \quad (7)$$

donde $f_i(\cdot)$ denota las activaciones en la capa i del discriminador y N_i el número de activaciones en esa capa. Con un peso $\lambda_{\text{FM}} = 5,0$, esta pérdida fuerza al generador a reproducir las estadísticas de características del discriminador, lo que en la práctica reduce el colapso de modos y mejora la calidad visual.

2.2.4 Mode Seeking Loss

Mao et al. [10] propusieron la pérdida de búsqueda de modos para maximizar la diversidad de las salidas del generador. Dados dos vectores de ruido independientes $\mathbf{z}_1$ y $\mathbf{z}_2$, la pérdida se define como:

$$\mathcal{L}_{\text{MS}} = -\mathbb{E}_{\mathbf{z}_1, \mathbf{z}_2} \left[\frac{\|G(\mathbf{z}_1) - G(\mathbf{z}_2)\|_2 / \sqrt{d}}{\|\mathbf{z}_1 - \mathbf{z}_2\|_2 / \sqrt{n_z} + \epsilon} \right] \quad (8)$$

donde $d = C \times H \times W$ es el número de píxeles totales, n_z es la dimensión del espacio latente, y $\epsilon = 10^{-5}$ evita división por cero. Minimizar $\mathcal{L}_{\text{MS}}$ equivale a maximizar la diversidad de salidas relativa a la diversidad de entradas.

Esta pérdida presenta una ventaja crucial sobre el uso de `torch.pdist`: cuando el generador colapsa y todas las salidas son idénticas, el gradiente de `pdist` es cero y el optimizador no

tiene señal para escapar del colapso. Con $\mathcal{L}_{MS}$, dado que $\mathbf{z}_1 \neq \mathbf{z}_2$ con probabilidad 1, siempre existe gradiente no nulo, incluso en el colapso completo.

2.3 Técnicas de Estabilización

2.3.1 Normalización Espectral

La normalización espectral [5] normaliza las matrices de peso de las capas del discriminador por su mayor valor singular:

$$\hat{\mathbf{W}} = \mathbf{W} / \sigma(\mathbf{W}) \quad (9)$$

donde $\sigma(\mathbf{W})$ es la norma espectral (mayor valor singular) de $\mathbf{W}$. Esto impone una restricción de Lipschitz sobre el discriminador, evitando que sus gradientes exploten y estabilizando el entrenamiento.

2.3.2 Desviación Estándar de Minibatch

Salimans et al. [7] y posteriormente Karras et al. [11] utilizaron la desviación estándar de minibatch como mecanismo explícito de anti-colapso de modos. La idea es concatenar, como característica adicional del discriminador, la desviación estándar media del minibatch:

$$\text{std}_{\text{batch}} = \frac{1}{C \cdot H \cdot W} \sum_{c,h,w} \text{Std}_b[\phi_{b,c,h,w}] \quad (10)$$

donde $\phi_{b,c,h,w}$ son las activaciones de la última capa convolucional del discriminador. Este escalar se expande y concatena al vector de características antes de la capa lineal de salida.

Si el generador colapsa y todas las muestras son idénticas, $\text{std}_{\text{batch}} \approx 0$ para los datos falsos pero no para los reales; el discriminador puede usar esta señal para distinguir datos reales de falsos, forzando al generador a diversificar sus salidas.

2.3.3 DiffAugment

Zhao et al. [9] propusieron aplicar aumentación diferenciable (*DiffAugment*) tanto a imágenes reales como a imágenes sintéticas antes de pasarlas al discriminador. A diferencia de la aumentación tradicional (sólo en datos reales), aquí ambas ramas reciben la misma transformación, lo que permite calcular gradientes a través de la aumentación.

Las políticas utilizadas en este trabajo son:

- **Color:** perturbaciones aleatorias de brillo, saturación y contraste.

- **Traslación:** recorte y relleno aleatorio.
- **Cutout:** enmascaramiento rectangular aleatorio.

DiffAugment es especialmente relevante cuando el dataset de entrenamiento proviene de un único simulador (en este trabajo, ~ 68.500 imágenes generadas a partir de ~ 6.500 registros CONAF), ya que previene que el discriminador memorice artefactos de bajo nivel del CA.

2.3.4 Promedio Exponencial de Pesos (EMA)

El promedio exponencial de los pesos del generador (*Exponential Moving Average*, EMA) [11, 25] mantiene una copia θ_{shadow} que se actualiza suavemente:

$$\theta_{\text{shadow}} \leftarrow \alpha \theta_{\text{shadow}} + (1 - \alpha) \theta_G \quad (11)$$

con $\alpha = 0,999$. Las muestras se generan siempre con la copia EMA, que es más estable y de mayor calidad que los pesos directos del generador durante el entrenamiento.

2.4 Autómatas Celulares para Propagación de Incendios

2.4.1 Fundamentos

Los autómatas celulares (AC) son modelos computacionales discretos en los que un grid de celdas evoluciona según reglas locales deterministas o estocásticas [18]. Su aplicación a la simulación de propagación de incendios forestales ha sido ampliamente estudiada [19, 20].

En el modelo de AC utilizado, cada celda puede estar en uno de cuatro estados: UNBURNED (no quemada), BURNING (en llamas), BURNED_OUT (extinguida) y NON_FLAMMABLE (no inflamable). La vecindad de Moore (8 celdas adyacentes) determina los posibles caminos de propagación.

2.4.2 Probabilidad de Propagación

La probabilidad de que una celda (i, j) en estado BURNING encienda a su vecino en dirección d en el siguiente paso es:

$$P_{i,j \rightarrow d} = \min(p_{\text{base}} \cdot f(\text{veg}) \cdot (1 - \lambda_h h) + \delta_T + w_d + s_{i,j,d}, 0,95) \quad (12)$$

donde:

$$\begin{aligned}
p_{\text{base}} &= 0,15 \quad (\text{probabilidad base de propagación}) \\
f(\text{veg}) &\in [0, 0,85] \quad (\text{ininflamabilidad por tipo de vegetación}) \\
\lambda_h &= 0,40 \quad (\text{factor de reducción por humedad}) \\
\delta_T &= 0,20 \cdot T \cdot 0,1 \quad (\text{bonus por temperatura}) \\
w_d &= 0,50 \cdot v \cdot \max(0, \langle \hat{d}, \hat{w} \rangle) \quad (\text{bonus por alineación con el viento}) \\
s_{i,j,d} &= 0,15 \cdot \max(0, \Delta e_{i,j,d} \cdot 5) \quad (\text{bonus por pendiente ascendente})
\end{aligned}$$

Aquí v es la velocidad del viento normalizada, $\hat{w}$ el vector unitario de dirección del viento, $\hat{d}$ el vector unitario de la dirección vecina, y $\Delta e_{i,j,d}$ la diferencia de elevación hacia el vecino en la dirección d .

2.4.3 Heatmaps Monte Carlo

Las simulaciones individuales del AC producen resultados estocásticos (la propagación depende de generadores de números aleatorios). Para obtener una representación probabilística robusta, se ejecutan $N_{\text{MC}} = 30$ simulaciones independientes con el mismo terreno y condiciones meteorológicas pero distintas semillas. El mapa de calor se define como:

$$H(i, j) = \frac{1}{N_{\text{MC}}} \sum_{k=1}^{N_{\text{MC}}} \mathbf{1}[\text{celda}(i, j) \text{ quemada en simulación } k] \quad (13)$$

$H(i, j) \in [0, 1]$ representa la probabilidad empírica de que la celda (i, j) sea alcanzada por el fuego bajo las condiciones dadas. Los *heatmaps* producen una representación continua y visualmente informativa de la distribución de propagación esperada.

2.5 Métricas de Evaluación

2.5.1 Distancia de Fréchet de Inception (FID)

La FID [8] mide la distancia entre la distribución de imágenes reales y la distribución de imágenes generadas en el espacio de características del modelo InceptionV3 preentrenado. Formalmente:

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}) \quad (14)$$

donde (μ_r, Σ_r) y (μ_g, Σ_g) son las medias y covarianzas de las activaciones de InceptionV3 para imágenes reales y generadas respectivamente. Valores más bajos de FID indican mayor similitud entre distribuciones.

2.5.2 SSIM Condicionada

La similitud estructural (SSIM) [15] mide la similitud perceptual entre imágenes considerando luminancia, contraste y estructura. En este trabajo se utiliza una versión condicionada: para cada imagen generada con condición $\mathbf{c}$, se buscan los $k = 5$ vecinos más cercanos en el conjunto de validación (por distancia coseno en el espacio de condiciones), y se calcula la SSIM media entre la imagen generada y sus vecinos reales. Esto penaliza al modelo si genera imágenes que no guardan coherencia estructural con los ejemplos reales de condiciones similares.

CAPÍTULO 3

Datos y Preprocesamiento

3.1 Dataset CONAF

Los datos utilizados en este trabajo provienen de los registros históricos de incendios forestales de la Corporación Nacional Forestal (CONAF), disponibles en formato de hoja de cálculo (Excel). Cada registro corresponde a un incendio individual e incluye las variables descritas en la Tabla 1.

Tabla 1: Variables del dataset CONAF utilizadas en el pipeline. Fuente: Elaboración propia.

Variable	Tipo	Descripción
Temperatura	Continua	Temperatura ambiente al momento del incendio (°C)
Humedad	Continua	Humedad relativa (%)
Velocidad viento	Continua	Velocidad del viento (km/h)
Superficie total	Continua	Superficie quemada total (ha)
Exposición	Categórica	Orientación del predio (10 categorías)
Dirección viento	Categórica	Dirección del viento (10 categorías)
Topografía	Categórica	Tipo de terreno (4 categorías)
Arbolado–Pastizal	Continua	Coberturas vegetales (ha por tipo)

La distribución de superficie afectada presenta una fuerte asimetría positiva: el 63% de los registros corresponde a incendios menores de 1 ha, mientras que el 5% más grande supera las 50 ha. Esta distribución de cola pesada justifica la transformación logarítmica $\log(1 + \text{ha})$ aplicada antes de la normalización.

3.2 Preprocesamiento del vector de condición

3.2.1 Imputación

Dado que una fracción de los registros presenta valores faltantes en algunas variables, se aplica la siguiente estrategia antes de la normalización (respetando el orden de operaciones para que las máscaras reflejen la ausencia original):

1. Se computa la máscara de valores faltantes $\mathbf{m} \in \{0, 1\}^{32}$ *antes* de cualquier imputación.
2. Las variables continuas con valores faltantes se imputan con la mediana del conjunto completo.
3. Las variables categóricas faltantes se asignan a una categoría especial `Missing`, que tiene su propia dimensión en el vector one-hot.

Esta estrategia preserva la información sobre ausencia de datos, lo que es fundamental para el objetivo de robustez ante datos incompletos.

3.2.2 Normalización de variables continuas

Las cuatro variables continuas (Temperatura, Humedad, Velocidad del viento, Superficie total) se normalizan con `StandardScaler`:

$$z = \frac{x - \mu}{\sigma} \quad (15)$$

A la variable Superficie total se le aplica previamente la transformación $\log(1 + x)$ para corregir la asimetría. El escalador ajustado se guarda en disco (`scaler.pkl`) para su posterior uso en la generación inversa de valores físicos durante la simulación CA.

3.2.3 Codificación de variables categóricas

Las tres variables categóricas se codifican como vectores one-hot, lo que produce un total de $10 + 10 + 4 = 24$ dimensiones:

- Exposición: $\{\text{N, NE, E, SE, S, SW, W, NW, Plano, Missing}\}$ (10 cats.)
- Dirección del viento: $\{\text{N, NE, E, SE, S, SW, W, NW, Calma, Missing}\}$ (10 cats.)
- Topografía: $\{\text{Suave, Irregular, Abrupta, Missing}\}$ (4 cats.)

3.2.4 Layout del vector de condición

El vector de condición final $\mathbf{c} \in \mathbb{R}^{32}$ tiene el siguiente layout:

$$\mathbf{c} = \left[\underbrace{c_0, c_1, c_2, c_3}_{4 \text{ continuos}} \mid \underbrace{c_4, \dots, c_{13}}_{\text{Exposición (10)}} \mid \underbrace{c_{14}, \dots, c_{23}}_{\text{Dir. viento (10)}} \mid \underbrace{c_{24}, \dots, c_{27}}_{\text{Topografía (4)}} \mid \underbrace{c_{28}, \dots, c_{31}}_{\text{Miss. flags (4)}} \right] \quad (16)$$

Los cuatro últimos componentes son los *missingness flags*: un 1 indica que la variable continua correspondiente estaba ausente en el registro original.

3.3 Perfil de vegetación

El perfil de vegetación $\mathbf{v} \in \mathbb{R}^{10}$ codifica la composición vegetal del predio afectado como proporciones que suman a 1. La Tabla 2 muestra el mapeo desde las columnas del Excel a las diez dimensiones del perfil.

Tabla 2: Mapeo de columnas Excel al perfil de vegetación (10 dims). Fuente: Elaboración propia.

Índice	Nombre interno	Columna Excel	Ininflamabilidad CA
0	Arbolado_Nat	Arbolado	0,60
1	Arbolado_Exot	Eucalipto	0,75
2	Arbolado_Mixto	Otras plantaciones	0,65
3	Matorral	Matorral	0,70
4	Pastizal	Pastizal	0,80
5	Agrícola	Agrícola	0,50
6	Desechos	Desechos	0,85
7	Plantacion_Joven	Pino 0 a 10	0,70
8	Sin_Vegetacion	Calculado*	0,00
9	Otro_Comb	Pino 11+ años	0,40

* $\max(0, \text{Sup_Total} - \sum \text{otras coberturas})$

3.4 División train/val

El conjunto de datos se divide en 90% entrenamiento y 10% validación (fracción configurable), mediante permutación aleatoria con semilla fija ($\text{seed}=42$) para reproducibilidad. La división se realiza al nivel de registro, de modo que todas las imágenes generadas para un registro pertenecen al mismo subconjunto.

3.5 Aumentación de condiciones

Durante el entrenamiento, el vector de condición es sometido a *condition masking*: con probabilidad $p_m = 0,3$ por campo, cada campo del vector (excluyendo los *missingness flags*) es enmascarado en su totalidad (puesto a cero) y el flag de *missingness* correspondiente se actualiza. Los campos one-hot (exposición, dirección de viento, topografía) se enmascaran como bloque completo para preservar la validez de la codificación. Esto entrena al modelo a generar imágenes plausibles incluso cuando parte de la información de condición está ausente — escenario habitual en incendios forestales en el que los registros meteorológicos suelen tener campos vacíos.

CAPÍTULO 4

Sistema de autómatas celulares

4.1 Arquitectura del simulador

El simulador de propagación de incendios implementa un autómata celular estocástico vectorizado sobre un grid de 64×64 celdas. Cada celda almacena su tipo de vegetación (índice 0–9) y su estado actual. Todas las operaciones de propagación se realizan sobre el grid completo usando operaciones de *numpy*, eliminando los bucles celda por celda de Python para mayor eficiencia.

La ignición se inicia en un radio central de 2 celdas, simulando un foco puntual inicial. En cada paso de tiempo:

1. Se identifican todas las celdas BURNING.
2. Para cada dirección vecina $d \in \{1, \dots, 8\}$, se computa la probabilidad de propagación $P_{i,j \rightarrow d}$ según la ecuación (12).
3. Se genera una muestra aleatoria $u \sim \mathcal{U}(0, 1)$ por celda candidata; si $u < P_{i,j \rightarrow d}$, la celda vecina pasa a BURNING.
4. Las celdas BURNING pasan a BURNED_OUT con probabilidad $p_{\text{out}} = 0,15 + 0,1 \cdot (1 - f(\text{veg}))$.

4.2 Generación de terreno sintético

Para cada registro del dataset se generan $n_v = 3$ variaciones de terreno independientes, que capturan la variabilidad espacial no observada. El terreno incluye dos capas:

Grid de vegetación. Generado mediante ruido de Perlin con parámetros (escala = 0,08, octavas = 4). El campo de ruido se discretiza en tipos de vegetación según las proporciones del perfil $\mathbf{v}$.

Grid de elevación. También generado con ruido de Perlin, con amplitud proporcional al tipo de topografía: Suave (0,2), Irregular (0,4), Abrupta (0,8) y Missing (0,5 por defecto). La elevación afecta la probabilidad de propagación mediante diferencias de elevación entre vecinos.

4.3 Heatmaps Monte Carlo con horizontes temporales

Para cada variación de terreno se ejecutan $N_{MC} = 30$ simulaciones con la misma configuración pero distintas semillas (espaciadas con números primos: $seed_k = seed_0 + k \times 7919$ para $k = 0, \dots, N_{MC} - 1$).

Se capturan $n_h = 4$ horizontes temporales por simulación, en los pasos $\{n/4, n/2, 3n/4, n\}$ donde n es el número total de pasos calibrado. Esto produce mapas de calor H_1, H_2, H_3, H_4 que muestran la evolución probabilística del incendio a distintos momentos.

El total de imágenes por registro es $n_v \times n_h = 12$.

4.4 Calibración de pasos por tamaño de incendio

4.4.1 Problema identificado

En la implementación inicial del pipeline, todos los registros del dataset utilizaban $n = 80$ pasos de simulación, independientemente de la superficie real del incendio registrada en `Sup_Total`. Dado que el 63% de los incendios tiene menos de 1 ha, esto generaba una inconsistencia fundamental: el vector de condición $\mathbf{c}$ codificaba el tamaño real del incendio (en c_3), pero la imagen entrenada mostraba siempre 80 pasos de propagación, equivalente a incendios grandes. Esta discrepancia entre condición e imagen debilita la capacidad del generador para aprender la relación condición–imagen.

4.4.2 Corrección implementada

Se implementó la función `calibrate_n_steps`:

$$n(z) = n_{\min} + \text{round}\left(\frac{\text{clip}(z, -2, 2) + 2}{4} \cdot (n_{\max} - n_{\min})\right) \quad (17)$$

donde $z = c_3$ es el z-score de $\log(1 + \text{ha})$, $n_{\min} = 8$ y $n_{\max} = 80$ en la implementación inicial. El mapeo produce:

- $z \leq -2$ (incendios muy pequeños, < 1 ha): $n = 8$ pasos
- $z = 0$ (incendio mediano): $n = 44$ pasos
- $z \geq +2$ (incendios grandes, > 100 ha): $n = 80$ pasos

Esta corrección alinea la imagen generada con el vector de condición, garantizando que el generador reciba pares (condición, imagen) coherentes. En la configuración definitiva (Run 17 en adelante) se usa $n_{\min} = 40$, produciendo $z = -2 \rightarrow 40$ pasos y $z = 0 \rightarrow 60$ pasos (véase Sección 6.6.5).

4.5 Normalización de parámetros meteorológicos

4.5.1 Problema identificado

La función `extract_weather_params` que transforma el vector de condición (z-scores) a parámetros del CA (rango $[0, 1]$) utilizaba mapeos ad hoc:

Listing 1: Mapeo aproximado original

```
1 temperature = np.clip((temp_z + 2) / 4, 0, 1) # numeros magicos
2 humidity    = np.clip((hum_z + 2) / 4, 0, 1)
3 wind_speed  = np.clip((wind_z + 1) / 4, 0, 1)
```

Este mapeo asume z-scores en $[-2, 2]$ y trunca información para valores fuera de este rango (outliers legítimos del dataset).

4.5.2 Corrección implementada

Se usa el escalador `StandardScaler` guardado en preprocesamiento para invertir el z-score a valores físicos, y luego se normaliza por rangos de dominio conocidos:

$$x_{\text{físico}} = z \cdot \sigma + \mu, \quad x_{[0,1]} = \text{clip}\left(\frac{x_{\text{físico}}}{x_{\max}}, 0, 1\right) \quad (18)$$

con $T_{\max} = 50^\circ\text{C}$, $H_{\max} = 100\%$, $v_{\max} = 120$ km/h. Este enfoque respeta la distribución real de los datos y no trunca información de outliers.

CAPÍTULO 5

Arquitectura del Modelo Generativo

5.1 Visión general

El modelo implementado es una GAN condicional profunda convolucional (cDCGAN) cuyos componentes principales se resumen en la Tabla 3.

Tabla 3: Resumen de la arquitectura del modelo. Fuente: Elaboración propia.

Componente	Entrada	Salida
ConditionEmbedding	$\mathbb{R}^{74}$ (condición + máscara + vegetación)	$\mathbb{R}^{128}$
Generator	$\mathbb{R}^{128}$ (ruido) + $\mathbb{R}^{128}$ (embed.)	$\mathbb{R}^{3 \times 64 \times 64}$
Discriminator	$\mathbb{R}^{3 \times 64 \times 64} + \mathbb{R}^{128}$ (embed.)	$(\mathbb{R}^1, \mathbb{R}^8)$

5.2 Módulo de embedding de condición

Tanto el generador como el discriminador comparten el mismo módulo de embedding de condición. La entrada es la concatenación del vector de condición enmascarado, la máscara y el perfil de vegetación:

$$\mathbf{e}_{in} = [\mathbf{c} \odot (1 - \mathbf{m}); \mathbf{m}; \mathbf{v}] \in \mathbb{R}^{74} \quad (19)$$

donde $\odot$ denota el producto elemento a elemento y el punto y coma la concatenación. Esta formulación pone a cero los valores ausentes y entrega la máscara explícitamente, permitiendo al modelo distinguir entre un valor cero real y un valor faltante. El perfil de vegetación $\mathbf{v} \in \mathbb{R}^{10}$ no requiere máscara: siempre está disponible al calcularse directamente de las coberturas registradas.

El módulo es un MLP de tres capas:

$$\text{CondEmbed} : \mathbb{R}^{74} \xrightarrow{\text{Lin+LReLU}} \mathbb{R}^{256} \xrightarrow{\text{Lin+LReLU}} \mathbb{R}^{256} \xrightarrow{\text{Lin}} \mathbb{R}^{128} \quad (20)$$

5.3 Generador

El generador recibe el vector de ruido $\mathbf{z} \in \mathbb{R}^{128}$ y el embedding de condición $\mathbf{e} \in \mathbb{R}^{128}$, los concatena, y produce una imagen 64×64 en el rango $[-1, 1]$:

Tabla 4: Arquitectura del generador. Fuente: Elaboración propia.

Capa	Tipo	Canales	Resolución	Activación
Proyección	Linear	256 $\rightarrow$ 512 $\times$ 16	—	BN1d + ReLU
Reshape	—	512	4 $\times$ 4	—
UpBlock 1	ConvT 4 $\times$ 4 s=2	512 $\rightarrow$ 256	8 $\times$ 8	CBN + ReLU
UpBlock 2	ConvT 4 $\times$ 4 s=2	256 $\rightarrow$ 128	16 $\times$ 16	CBN + ReLU
UpBlock 3	ConvT 4 $\times$ 4 s=2	128 $\rightarrow$ 64	32 $\times$ 32	CBN + ReLU
UpBlock 4	ConvT 4 $\times$ 4 s=2	64 $\rightarrow$ 3	64 $\times$ 64	Tanh

CBN: *Conditional Batch Normalization* — γ, β predichos linealmente desde el embedding de condición en cada bloque; BatchNorm2d con `affine=False` internamente.

Nota sobre BatchNorm vs InstanceNorm. Se evaluó el uso de InstanceNorm en los bloques de *upsampling*. InstanceNorm normaliza cada instancia de forma independiente, destruyendo la señal de condición que el generador necesita respetar (dado que la condición lleva información per-instancia). BatchNorm, en cambio, preserva estadísticas de nivel de batch compatibles con la generación condicional. La eliminación de InstanceNorm en favor de BatchNorm fue uno de los cambios clave que produjo la mejora de FID de 222 a 174 (Run 2 $\rightarrow$ Run 3).

5.4 Discriminador

El discriminador recibe una imagen (real o generada) y su condición asociada, y produce dos salidas: un score escalar real/falso y logits de clasificación de dirección de viento en 8 clases (cabeza AC-GAN, activa desde Run 44). Incorpora normalización espectral en todas sus capas y desviación estándar de minibatch en la capa final.

Tabla 5: Arquitectura del discriminador. Fuente: Elaboración propia.

Capa	Tipo	Canales	Resolución	Activación
DownBlock 1	Conv 4×4 $s=2$ + SN	$3 \rightarrow 64$	32×32	LReLU(0.2)
DownBlock 2	Conv 4×4 $s=2$ + SN	$64 \rightarrow 128$	16×16	LReLU(0.2)
DownBlock 3	Conv 4×4 $s=2$ + SN	$128 \rightarrow 256$	8×8	LReLU(0.2)
DownBlock 4	Conv 4×4 $s=2$ + SN	$256 \rightarrow 512$	4×4	LReLU(0.2)
GlobalAvgPool	$\text{mean}(\text{dim}=[2,3])$	512	—	—
Minibatch std	concat std	$512 \rightarrow 513$	—	—
Linear + SN	—	$513 \rightarrow 1$	—	score
dir_head	Linear + SN	$512 \rightarrow 8$	—	logits AC-GAN

+ Proyección: $\langle \phi(\mathbf{x}), \mathbf{V}\mathbf{e} \rangle$ sumado al score (Runs 1–43)
+ dir_head: clasificador AC-GAN de 8 clases de dirección (Run 48, modelo definitivo)

La pérdida total del discriminador combina *hinge loss* y regularización R1. En el modelo definitivo (Run 48) se añade además la pérdida de clasificación AC-GAN $\mathcal{L}_D^{\text{dir}}$ (entropía cruzada sobre las 8 clases de dirección):

$$\mathcal{L}_D^{\text{total}} = \mathcal{L}_D + \mathcal{R}_1 + \mathcal{L}_D^{\text{dir}} \quad (21)$$

La pérdida total del generador combina *hinge loss*, *feature matching*, *mode seeking* y, en el modelo definitivo, la pérdida de clasificación AC-GAN:

$$\mathcal{L}_G^{\text{total}} = \mathcal{L}_G + \lambda_{\text{FM}} \mathcal{L}_{\text{FM}} + \lambda_{\text{MS}} \mathcal{L}_{\text{MS}} + \lambda_{\text{dir}} \mathcal{L}_G^{\text{dir}} \quad (22)$$

con $\lambda_{\text{FM}} = 5,0$, $\lambda_{\text{MS}} = 0,1$ y $\lambda_{\text{dir}} = 1,0$.

CAPÍTULO 6

Proceso de Entrenamiento

6.1 Configuración general

El entrenamiento se realizó sobre una instancia GPU RTX 5080 (16 GB VRAM) en la plataforma Vast.ai. Los principales hiperparámetros se detallan en la Tabla 6.

Tabla 6: Hiperparámetros de entrenamiento. Fuente: Elaboración propia.

Parámetro	Valor	Justificación
Épocas	150	Suficiente para convergencia con cosine annealing
Batch size	64	Balance entre calidad de gradiente y VRAM
n_z (dimensión latente)	128	Capacidad representacional estándar
lr_G	1×10^{-4}	TTUR: lr_G menor para convergencia Nash [8]
lr_D	4×10^{-4}	TTUR: lr_D $4 \times$ mayor que lr_G
β_1	0,0	Sin momentum de gradiente (reduce oscilaciones)
β_2	0,99	Estimación estable de la varianza [17]
n_{critic}	4	4 pasos D por cada paso G (desde Run 42)
γ_{R1}	10,0	Regularización R1 con mayor penalización de gradiente
λ_{FM}	5,0	Feature matching moderado
λ_{MS}	0,1	Anti-colapso suave
EMA decay	0,999	Suavizado estable del generador

6.2 Programación del learning rate

Se utiliza *Cosine Annealing* con $T_{\max} = 150$ épocas y $lr_{\min} = 10^{-5}$:

$$lr(e) = lr_{\min} + \frac{1}{2}(lr_0 - lr_{\min}) \left(1 + \cos\left(\frac{\pi e}{T_{\max}}\right) \right) \quad (23)$$

Este esquema permite una tasa de aprendizaje alta durante las primeras épocas (convergencia rápida) y la reduce suavemente en épocas tardías para evitar oscilaciones. Se aplica el mismo scheduler a ambos optimizadores.

6.3 BestCheckpointCallback

Para no perder el mejor modelo en caso de que el entrenamiento degrade posteriormente, se implementó `BestCheckpointCallback`: cada $n_{\text{interval}} = 5$ épocas se generan muestras con condiciones fijas del conjunto de validación, se calcula la diversidad inter-muestras como $d = \text{std}(G(\mathbf{z})).\text{mean}()$, y si d supera el máximo histórico se guarda `checkpoint_best.pt`. Esto asegura que el mejor modelo (cuando D y G están en equilibrio) queda preservado.

6.4 Monitor de entrenamiento

Se desarrolló el script `monitor_training.py` que analiza en tiempo real los grids de muestras guardados periódicamente. Las métricas calculadas son:

- **Diversidad:** std inter-muestras (promedio de std por posición de píxel). Valor $< 0,015$ indica colapso crítico.
- **Textura:** std intra-muestra. Valor $< 0,02$ indica imágenes planas/uniformes.
- **Similitud coseno media:** entre pares de muestras. Valor $> 0,97$ indica colapso severo.
- **Entropía:** del histograma global de píxeles.

6.5 Historial de experimentos

Se realizaron catorce experimentos de entrenamiento en la fase exploratoria, cada uno a partir del diagnóstico del run previo. La Tabla 7 resume los resultados cuantitativos.

Tabla 7: Resultados de la fase exploratoria (Runs 1–14), entrenados con el simulador isotrópico. Runs 1–6: protocolo v0 (*val set* original). Runs 7–14: protocolo v2 (condiciones alineadas, 2.400 imágenes *val*). Run 13 no re-evaluable con arquitectura v1. Estos runs son comparables entre sí internamente; la fase de validación (Runs 15–20) con simulador calibrado se presenta en la sección siguiente. Fuente: Elaboración propia.

Run	Cambio principal	Descripción	FID ↓	SSIM ↑
1	Baseline (BCE)	Loss BCE, sin conditioning completo	206,53	—
2	Hinge + conditioning	Hinge loss, conditioning por proyección	222,07	—
3	+Minibatch std, sin InstanceNorm	Antes mejor [†] ; protocolo v2: 220,79	174,36 [†]	0,3911 [†]
4	$\gamma_{R1} = 0,5$	Test con menor regularización R1	213,04	—
5	pdist diversity	Pérdida de diversidad con pdist	Colapso ep. 5	—
6	Mode Seeking $\lambda = 1,0$	Diversidad dominante	255,88	—
<i>Runs 7–12: dataset calibrado, protocolo de evaluación v2 (ver nota metodológica)</i>				
7	Dataset calibrado + MS $\lambda=0,1$	Sin veg_profile	173,76	0,1997*
8	+Perfil vegetación (74 dims)	Primera vez sin colapso completo	—	— [‡]
9	diversity_weight 0,2	Plateau idéntico al Run 8	201,60	0,3715
10	TTUR $\text{lr}_G=10^{-4}$, $\text{lr}_D=4 \times 10^{-4}$	Mejor balance FID/SSIM	168,47 ± 0,47[¶]	0,4634 ± 0,0023[¶]
11	Dataset reducido ~6.000 imgs	TTUR + dataset compacto	220,60	0,5267
12	diversity_weight 0,1→0,15	TTUR + ~68.5k, mayor presión diversidad	214,75	0,4076
13	Arquitectura escalada (9,3M params)	g_features×2, cond_hidden 512	— [§]	— [§]
14	Fix flips + fix masking por campo	Corrección bugs augmentación, base Run 10	258,75	0,2400

* SSIM bajo por mayor diversidad morfológica sin conditioning de vegetación.

[†] Val set con bug de n_{steps} : FID = 174,36 y SSIM = 0,4964 son valores del protocolo v0 (artefactos); protocolo v2 da FID = 220,79 y SSIM = 0,3911.

[‡] Run 8: checkpoint no disponible para re-evaluación.

[§] Run 13: arquitectura distinta (9,3M params); no comparable con protocolo v2 arquitectura v1.

[¶] Run 10: media ± desv. est. sobre 5 evaluaciones independientes con distinto ruido z (semillas 1–5).

Nota metodológica: protocolo de evaluación v2. Durante el desarrollo se identificaron dos inconsistencias en el pipeline de evaluación que motivaron la definición de un protocolo corregido (v2): **Corrección A** — alineación de condiciones: el evaluador original generaba imágenes con `conditions[:n]` del dataset completo, mientras las imágenes reales provenían del subconjunto de validación. Las condiciones generadas y reales no correspondían al mismo subconjunto, sesgando FID y SSIM. Corregido en `src/evaluation/evaluator.py`: `prepare_real_images` ahora extrae condiciones, máscaras y perfiles de vegetación directamente del `val_loader`. **Corrección B** — tamaño del *val set*: el *val set* original contenía 72 imágenes (60 registros), insuficiente para FID estable (se requieren ≥ 500 imágenes). Se regeneró el *val set* con `--max-records 2000` obteniendo 2.400 imágenes; la evaluación usa las 2.400 imágenes completas (`fid_n_samples = 2400`). Los valores de FID del protocolo v1 (FID $\sim$ 210) son sistemáticamente más altos que los del v2 (FID $\sim$ 184) porque el *val set* pequeño amplificaba la varianza del estimador. Los rankings relativos entre runs se mantienen en ambos protocolos.

6.5.1 Run 1: Baseline

El entrenamiento con BCE loss resultó en colapso de modos hacia la época 50, con imágenes de patrones uniformes verde/marrón. El discriminador converge demasiado rápido y los gradientes para el generador se vuelven despreciables.

6.5.2 Run 2: Hinge loss y conditioning completo

La adopción de *hinge loss* mejoró la estabilidad del discriminador, y el conditioning por proyección permitió al generador incorporar información de condición. El FID de 222,07 fue paradójicamente peor que el baseline, pero la diversidad cualitativa mejoró. $D(\text{real})$ se mantuvo en aproximadamente $-0,10$, indicando que el discriminador no distinguía bien entre datos reales y falsos.

6.5.3 Run 3: Minibatch stddev y eliminación de InstanceNorm (mejor FID con protocolo v0, artefacto)

Dos cambios simultáneos produjeron la mayor mejora. Primero, la adición de desviación estándar de minibatch en el discriminador proveyó señal anti-colapso. Segundo, la eliminación de InstanceNorm del generador en favor de BatchNorm permitió al generador preservar la señal de condición a través de las capas de normalización. El FID bajó a 174,36 y el SSIM alcanzó 0,4964 en la época 45.

6.5.4 Run 4: $\gamma_{R1} = 0,5$

La hipótesis de que menor regularización R1 permitiría al discriminador entrenarse más rápido no se confirmó. El FID empeoró a 213,04. Con R1 insuficiente, el discriminador se vuelve demasiado fuerte tempranamente, degradando la calidad del gradiente para el generador. Se revirtió a $\gamma = 1,0$.

6.5.5 Run 5: Pérdida de diversidad con `pdist`

La adición de `torch.pdist` como señal de diversidad resultó en colapso en la época 5. La causa es que `pdist` tiene gradiente exactamente cero cuando todas las muestras son idénticas: si $G(\mathbf{z}_1) = G(\mathbf{z}_2)$ para todo par, la distancia es 0 pero $\partial \text{pdist} / \partial \theta_G = 0$. El optimizador no tiene señal para escapar del colapso.

6.5.6 Run 6: Mode Seeking Loss con $\lambda = 1,0$

La Mode Seeking Loss de Mao et al. [10] usa dos vectores de ruido independientes, garantizando gradiente no nulo incluso en el colapso. Sin embargo, con $\lambda = 1,0$ el término de diversidad dominó sobre el término de realismo (hinge + FM). El generador aprendió a producir imágenes diversas pero visualmente caóticas. FID = 255,88, el peor resultado.

6.5.7 Run 7: Dataset calibrado + Mode Seeking Loss $\lambda = 0,1$ + Cosine LR

El dataset fue regenerado con calibración de n_{steps} por tamaño real del incendio y la normalización meteorológica corregida para usar el scaler real en lugar de mapeos ad hoc. Como regularizador anti-colapso se incorporó Mode Seeking Loss con $\lambda = 0,1$.

El entrenamiento se realizó sobre una instancia RTX 5070 durante 100 épocas (aproximadamente 1,5 minutos por época). El monitor reportó una oscilación temprana (CRITICO en época 5, AVISO en épocas 20-25) seguida de recuperación y estado OK sostenido desde la época 30 hasta la 100 — el período estable más largo observado hasta el momento.

Evolución del monitor. La diversidad alcanzó su máximo en la época 55 (0,40), valor notablemente superior a cualquier run anterior. La textura fue creciendo sostenidamente hasta valores de 0,45-0,47, los más altos registrados. El BestCheckpointCallback capturó el modelo en época 55.

Resultados cuantitativos.

- **FID = 173,76** (mejora marginal respecto al baseline 174,36)
- **SSIM = 0,1997 $\pm$ 0,1865** (caída respecto a 0,4964 del run 3)

Análisis del SSIM. La caída del SSIM se explica por el nuevo dataset más diverso: la calibración de n_{steps} introduce incendios de tamaños muy variados (desde 8 pasos para incendios < 1 ha hasta 80 pasos para grandes). El matching por condición puede entonces comparar imágenes estructuralmente distintas en tamaño, reduciendo la similitud estructural media. Este es un efecto esperado de un dataset más fiel a la realidad, no necesariamente un deterioro del modelo.

Análisis cualitativo. La inspección visual de los grids de muestras de la época 55 confirma diversidad real: incendios de distintos tamaños, con propagación desde puntos hacia zonas amplias, y patrones asimétricos coherentes con la dirección del viento condicionada. La presencia de incendios pequeños (puntos concentrados) y grandes (zonas blancas que cubren casi todo el grid) indica que el modelo está aprendiendo la relación entre c_3 (Sup_Total) y el tamaño de la propagación generada, lo que valida la corrección de calibración implementada.

Conclusión y siguiente paso. El FID no mejoró significativamente sobre el baseline, lo que sugiere que la mejora de la calidad del dataset es necesaria pero no suficiente para romper el techo actual. El modelo alcanza su mejor diversidad en torno a la época 55 y luego colapsa, un patrón recurrente en las últimas tres ejecuciones. El análisis señala que el vector de condición actual ($\mathbf{c} \in \mathbb{R}^{32}$) no incluye información de composición vegetal, pese a que la inflamabilidad por tipo de vegetación es un factor determinante en la velocidad y extensión de propagación de incendios forestales. Se decide implementar el perfil de vegetación como señal adicional de condicionamiento (Run 8), descrito en la subsección siguiente.

6.5.8 Run 8: Perfil de vegetación como condicionamiento adicional

Motivación. El autómata celular ya incorpora la inflamabilidad diferenciada por tipo vegetal (Tabla 2): eucalipto ($f = 0,75$), desechos ($f = 0,85$), pastizal ($f = 0,80$), etc. Sin embargo, el generador recibía únicamente variables meteorológicas y topográficas, sin acceso a la composición vegetal del área quemada. Este desajuste implica que el modelo no podía condicionar la morfología generada en función de si el incendio ocurrió en bosque nativo, eucaliptal o matorral.

Cambio arquitectónico. Se extiende el ConditionEmbedding para incorporar el perfil de vegetación $\mathbf{v} \in \mathbb{R}^{10}$:

$$\mathbf{e} = \text{MLP}([\mathbf{c} \odot (1 - \mathbf{m}) \parallel \mathbf{m} \parallel \mathbf{v}]), \quad \mathbf{e} \in \mathbb{R}^{128} \quad (24)$$

La dimensión de entrada del MLP pasa de 64 a 74 (32 condición + 32 máscara + 10 vegetación). El vector $\mathbf{v}$ no tiene máscara asociada ya que siempre está disponible: se calcula directamente a partir de la superficie quemada por tipo registrada en el Excel, y se normaliza para sumar 1 (proporciones). La misma extensión se aplica tanto al generador como al discriminador, de modo que el discriminador también aprende a discriminar teniendo en cuenta la composición vegetal esperada para cada condición.

Tabla 8: Resumen del cambio arquitectónico del Run 8. Fuente: Elaboración propia.

Componente	Run 7 (entrada)	Run 8 (entrada)
ConditionEmbedding	$\mathbb{R}^{64}$	$\mathbb{R}^{74}$ (+10 veg.)
Generator input	$\mathbb{R}^{128+128}$	$\mathbb{R}^{128+128}$ (embed. distinto)
Discriminator	$\mathbb{R}^{3 \times 64 \times 64} + \mathbb{R}^{128}$	$\mathbb{R}^{3 \times 64 \times 64} + \mathbb{R}^{128}$ (embed. distinto)

Implementación. Los cambios principales se concentran en cinco archivos: `src/models/conditioning.py` (extensión del MLP), `src/models/generator.py` y `src/models/discriminator.py` (nuevo parámetro `veg_dim`), `src/training/trainer.py` (extracción del batch y propagación), y `src/training/callbacks.py` (fijado de `veg_profile` para comparación visual entre épocas). Los scripts de evaluación y generación de escenarios también se actualizaron para mantener coherencia.

Hipótesis de mejora. Al incluir la composición vegetal, el generador dispone de información para diferenciar morfologías: incendios en eucaliptal (alta inflamabilidad, propagación rápida y extensa) frente a zona agrícola (inflamabilidad moderada, bordes más regulares). Se espera una reducción del FID y mayor correlación entre las condiciones y las morfologías generadas.

6.5.9 Resultados del Run 8

Entrenamiento. El Run 8 fue el primer entrenamiento completo sin colapso de modos: 300 épocas en GPU RTX 5080 (50 s/época, 4 h totales). La diversidad inter-muestras se estableció en el rango $[0,35, 0,36]$ desde la época 65 hasta la 300, sin ningún signo de colapso. En contraste, todos los runs anteriores colapsaban en torno a la época 130.

Tabla 9: Evolución de métricas del monitor durante Run 8. Fuente: Elaboración propia.

Época	Diversidad	Textura	Similitud	D_loss	G_loss
10	0,156	0,191	0,974	1,869	0,142
30	0,311	0,262	0,807	1,883	0,246
55	0,333	0,324	0,765	1,861	0,314
75	0,363	0,335	0,652	1,842	0,349
130	0,362	0,389	0,675	1,806	0,447
200	0,364	0,402	0,631	1,766	0,498
300	0,352	0,396	0,635	1,735	0,569

Evaluación cuantitativa. Se evaluaron dos checkpoints: el de máxima diversidad (época 75, guardado por BestCheckpointCallback) y el checkpoint final (época 300):

Tabla 10: Resultados FID/SSIM del Run 8. Fuente: Elaboración propia.

Checkpoint	FID ↓	SSIM ↑
Run 7 best (época 55)	173,76	0,1997
Run 8 best (época 75)	218,00	0,2476
Run 8 final (época 300)	212,17	0,2298

Análisis. El SSIM mejoró un 24% respecto al Run 7 ($0,1997 \rightarrow 0,2476$), lo que indica que el perfil de vegetación ayuda al generador a producir imágenes más coherentes con las condiciones dadas. El FID empeoró, lo que sugiere que la distribución generada se aleja de la distribución real en el espacio de features de InceptionV3 — probablemente porque el modelo está generando imágenes con mayor variedad morfológica (más formas diferentes) a expensas de concentrarse menos en los patrones más frecuentes del dataset.

El resultado más relevante desde el punto de vista de la estabilidad es que el plateau duró 235 épocas (65–300) sin colapso, lo que no se había logrado en ningún run anterior. Esto sugiere que el conditioning con vegetación actúa como regularizador implícito que mantiene al discriminador y al generador en equilibrio por más tiempo.

Limitaciones y siguiente paso. La diversidad se estancó en $\sim 0,36$ desde época 65 sin mejorar, lo que indica que el modelo alcanzó un equilibrio local. Para Run 9 se propone aumentar el `diversity_weight` del Mode Seeking Loss ($\lambda = 0,1 \rightarrow 0,2$) para forzar mayor separación entre muestras, y evaluar si esto rompe el plateau sin causar inestabilidad.

6.5.10 Run 9: diversity_weight 0,1 $\rightarrow$ 0,2

Motivación. El plateau de diversidad de Run 8 ($\sim 0,36$, épocas 65–300) sugería que el Mode Seeking Loss no tenía suficiente peso para forzar separación entre muestras. Se duplicó el diversity_weight de 0,1 a 0,2 manteniendo el resto de la configuración idéntica a Run 8.

Comportamiento durante el entrenamiento. El plateau de diversidad se reprodujo exactamente: la métrica se estabilizó entre 0,33 y 0,37 desde la época 65, con una tendencia de apenas $-0,00004/\text{época}$ en las épocas finales. No se observó colapso (similitud $< 0,85$ sostenida durante las 300 épocas), y R1 se mantuvo estable en $\sim 0,07$. El comportamiento idéntico al Run 8 indica que el límite de diversidad no es el peso de la pérdida sino la variabilidad intrínseca del dataset o la capacidad de la arquitectura.

Tabla 11: Resultados FID/SSIM del Run 9. Fuente: Elaboración propia.

Checkpoint	FID ↓	SSIM ↑
Run 8 best (época 75)	218,00	0,2476
Run 9 best (checkpoint_best_run9.pt)	230,83	0,3042

Resultados.

Análisis. El SSIM mejoró un 23% respecto al Run 8 ($0,2476 \rightarrow 0,3042$), lo que puede atribuirse a mayor exploración del espacio latente inducida por el peso más alto, pero el FID empeoró ($218 \rightarrow 231$), confirmando la tensión entre diversidad y fidelidad. El dato más relevante es el plateau idéntico: diversity_weight no es el mecanismo limitante.

Conclusión del ciclo de runs 7–9. Los tres runs con dataset calibrado muestran una tendencia consistente: el modelo llega rápidamente a un equilibrio diversidad/fidelidad y no lo abandona independientemente del peso de Mode Seeking Loss. Las alternativas para un Run 10 son mecanismos de diversidad estructural (dropout en inferencia, truncation trick) o arquitecturas con mayor capacidad de representación.

6.5.11 Run 10: Two Time-Scale Update Rule (TTUR)

Motivación. El plateau sostenido en los Runs 7–9 apunta a un problema de dinámica de entrenamiento: con tasas de aprendizaje iguales ($\text{lr}_G = \text{lr}_D = 2 \times 10^{-4}$), el discriminador

y el generador compiten en condiciones simétricas que pueden estancar el sistema en un equilibrio subóptimo.

Heusel et al. [8] demuestran que con la *Two Time-Scale Update Rule* (TTUR) — $lr_D > lr_G$ — el sistema converge a un equilibrio de Nash local bajo condiciones generales. Al hacer que el discriminador aprenda más rápido, proporciona gradientes más informativos al generador, que puede escapar del equilibrio local con mayor facilidad.

Configuración.

- $lr_G = 1 \times 10^{-4}$, $lr_D = 4 \times 10^{-4}$ (ratio 1:4)
- `diversity_weight` revertido a 0,1 (0,2 no mejoró el plateau)
- Resto de hiperparámetros idénticos al Run 9

Tabla 12: Resultados FID/SSIM del Run 10 (evaluación interna durante entrenamiento, *val set* local 60 imágenes con bug de n_{steps}). Los valores corregidos con *val set* calibrado de 72 imágenes se reportan en la Tabla 13: FID = 209,91, SSIM = 0,4462. Los valores definitivos del protocolo v2 (2.400 imágenes, 5 semillas) se reportan en la Sección 6.5.16: FID = 168,47 $\pm$ 0,47, SSIM = 0,4634 $\pm$ 0,0023. Fuente: Elaboración propia.

Checkpoint	FID ↓	SSIM ↑
Run 9 best	230,83	0,3042
Run 10 best (época 85)	219,41	0,3646

Resultados.

Análisis. Con el *val set* calibrado, el TTUR produjo el mejor SSIM entre los Runs 7–10 re-evaluados con este *val set* (0,4462 $\pm$ 0,1140), una mejora del 22% respecto al Run 9 (0,3657). Esto confirma que el ratio $lr_D/lr_G = 4$ ayuda al generador a aprender imágenes más coherentes con las condiciones dadas. El FID de 209,91 es el mejor junto al Run 7 (209,38), con la ventaja de que Run 10 incorpora el perfil de vegetación.

La diversidad durante el entrenamiento osciló entre 0,25 y 0,30, con máximo en época 85 (0,294). El TTUR cambió la dinámica respecto a runs anteriores: el modelo convergió más rápido (similitud $< 0,85$ en época 55 vs época 65+ en runs previos) pero a un nivel de diversidad distinto.

Nota metodológica: corrección del *val set*. Durante la evaluación de los Runs 7–10 se detectó que el conjunto de validación local (60 imágenes) correspondía al dataset original con el bug de n_{steps} , mientras que el entrenamiento usaba el dataset calibrado (70.632 imágenes). Esta inconsistencia inflaba artificialmente el FID de los runs con dataset calibrado y deformaba la comparación con Run 3.

Para corregir esto, se regeneró el *val set* local usando el mismo pipeline calibrado (02_generate_ca_dataset.py --max-records 60), obteniendo 72 imágenes de validación con simulaciones correctas. Todos los checkpoints disponibles fueron re-evaluados con este *val set*. La Tabla 13 resume los resultados corregidos.

Tabla 13: Re-evaluación con *val set* calibrado (72 imágenes). Fuente: Elaboración propia.

Run	FID ↓	SSIM ↑	Val set
3	252,87	0,3868	original (con bug)
7	209,38	0,1900	calibrado
9	223,43	0,3657	calibrado
10	209,91	0,4462 ± 0,1140	calibrado

El resultado más importante es que Run 3, que parecía el mejor (FID = 174,36), obtiene FID = 252,87 cuando se mide contra imágenes calibradas — el peor de todos. Esto confirma que su aparente ventaja era un artefacto de medir la distribución generada contra la misma distribución con inconsistencias en que fue entrenado.

Conclusión. Run 10 (TTUR + veg_profile + dataset calibrado) es el mejor modelo de la fase exploratoria con FID = 209,91, SSIM = 0,4462 en esta re-evaluación. La validación multi-seed posterior (Sección 6.5.16) obtiene FID = 168,47 ± 0,47, SSIM = 0,4634 ± 0,0023 como resultado definitivo. El TTUR es la configuración más efectiva encontrada en la fase exploratoria.

6.5.12 Run 11: TTUR + dataset calibrado reducido (~6.000 imágenes)

Motivación. El FID de Run 10 (209,91) puede estar condicionado por el tamaño del dataset de entrenamiento: con 70.632 imágenes el modelo aprende una distribución muy amplia, lo que puede dificultar que el FID converja hacia valores más bajos al medirse contra un *val set* de 72 imágenes. La hipótesis es que un dataset calibrado de tamaño comparable al original (~6.000 imágenes, obtenido con --max-records 500) permitirá al modelo aprender una distribución más compacta y potencialmente reducir el FID, manteniendo las mejoras de SSIM logradas con el TTUR.

Configuración.

- Dataset: ~ 6.000 imágenes calibradas (`--max-records 500`)
- $lr_G = 1 \times 10^{-4}$, $lr_D = 4 \times 10^{-4}$ (TTUR, igual Run 10)
- `diversity_weight = 0,1`, resto idéntico a Run 10
- Épocas: 2.000 (equivalente a ~ 300 épocas de Run 10 en actualizaciones de gradiente, dado que el dataset 13x menor implica 13x menos batches por época)

Dinámica de entrenamiento. El modelo colapsó completamente en las épocas 5–10 (diversidad = 0,0016, similitud $> 0,99$), pero se recuperó espontáneamente hacia la época 35. La diversidad alcanzó su máximo en la época 300 (0,176) y luego descendió de forma consistente a razón de $-0,0005$ por época. El entrenamiento se detuvo en la época 400 para evaluar el mejor checkpoint, identificado en la época ~ 300 .

Resultados.

- **FID = 220,60** (peor que Run 10: 209,91; hipótesis no confirmada)
- **SSIM = $0,5267 \pm 0,0687$** (mejor SSIM de la fase exploratoria; obtenido con dataset reducido ~ 6.000 imágenes)

Análisis del trade-off. La hipótesis de que un dataset más compacto reduciría el FID no se confirmó: el modelo entrenado con ~ 6.000 imágenes obtuvo FID = 220,60 vs. 209,91 de Run 10 con 70.632 imágenes, una diferencia de 10,69 puntos. Sin embargo, el SSIM mejoró significativamente ($+0,0805$ sobre Run 10), lo que sugiere que el dataset más pequeño facilita el aprendizaje de coherencia estructural condicional a costa de una distribución marginal más alejada de los datos reales. Este resultado indica que el tamaño del dataset afecta de manera opuesta a las dos métricas: mayor dataset mejora FID (diversidad de la distribución aprendida) pero puede dificultar el SSIM (coherencia condicional por menor densidad de muestras similares por condición).

Conclusión. Run 10 sigue siendo el mejor modelo en FID (209,91), mientras que Run 11 obtiene el mejor SSIM (0,5267). Para los objetivos de este trabajo —generar imágenes coherentes con las condiciones meteorológicas y de vegetación— ambos modelos son candidatos válidos según la métrica priorizada.

6.5.13 Run 12: TTUR + dataset completo + diversity_weight = 0,15

Motivación. Run 10 (diversity_weight=0,1, 70k imágenes) obtuvo el mejor FID y Run 11 (diversity_weight=0,1, 6k imágenes) el mejor SSIM. La hipótesis de Run 12 es que aumentar la presión de diversidad a 0,15 manteniendo el dataset grande (~68.5k imágenes) puede mejorar el SSIM sin sacrificar el FID.

Configuración.

- Dataset: 68.519 imágenes calibradas (mismos hiperparámetros que Run 10)
- $lr_G = 1 \times 10^{-4}$, $lr_D = 4 \times 10^{-4}$ (TTUR)
- diversity_weight = 0,15 (único cambio respecto a Run 10)
- Épocas: 150

Dinámica de entrenamiento. La diversidad fue consistentemente mayor que en Run 10 durante todo el entrenamiento (máximo 0,272 vs ~0,20 en Run 10), con tendencia positiva sostenida hasta época 105 y estabilización posterior. La similitud se mantuvo por encima de 0,95 durante todo el run.

Resultados.

- **FID = 214,75** (peor que Run 10: 209,91; $\Delta = +4,84$)
- **SSIM = 0,4076 $\pm$ 0,0893** (peor que Run 10: 0,4462)

Análisis. La hipótesis no se confirmó. Aumentar diversity_weight de 0,1 a 0,15 con el dataset grande empeoró ambas métricas. Aunque el monitor reportó mayor diversidad morfológica durante el entrenamiento, esto no se tradujo en mejor FID ni SSIM. Este resultado sugiere que $\lambda_{div} = 0,1$ es el punto óptimo para el dataset completo (~68.5k): mayor presión de diversidad introduce ruido en el gradiente del generador que perjudica la calidad de la distribución aprendida.

6.5.14 Run 13: Arquitectura escalada + correcciones de implementación

Motivación. La revisión del código identificó dos problemas de implementación. El jitter de brillo en DiffAugment podía sacar píxeles del rango $[-1, 1]$ sin clipping posterior, y el ratio de

mode seeking loss podía ser numéricamente inestable cuando $\|z_1 - z_2\|$ era muy pequeño. Adicionalmente, la arquitectura actual (4,9M parámetros, cuello de botella de 64 canales en la última capa) podría estar limitando la capacidad expresiva del generador. Run 13 aplica ambas correcciones y escala la arquitectura.

Configuración.

- `g_features`: [512, 256, 128, 64] $\rightarrow$ [512, 512, 256, 128] (elimina cuello de botella)
- `cond_hidden_dim`: 256 $\rightarrow$ 512 (mayor capacidad en el path de conditioning)
- Fix: DiffAugment brightness clampea a $[-1, 1]$
- Fix: *mode seeking loss* clampea ratio a máximo 10,0
- `diversity_weight` = 0,1, TTUR, 70k imágenes, 150 épocas (idéntico a Run 10)
- Generator params: 9,3M (vs 4,9M en runs anteriores)

Dinámica de entrenamiento. El modelo alcanzó estado OK por primera vez en la historia del proyecto a la época 35 (diversidad = 0,37, similitud = 0,89). La diversidad llegó a un máximo de 0,43 en la época 45, con similitud bajando hasta 0,75. A partir de la época 55 la diversidad comenzó a declinar y la similitud subió nuevamente, estabilizándose en un equilibrio AVISO/CRITICO con diversidad $\sim 0,20$.

Resultados.

- **FID = 267,12** (peor que todos los runs anteriores)
- **SSIM = 0,3894 $\pm$ 0,1603** (SSIM std muy alto: alta varianza entre muestras)

Análisis. A pesar de la mejor dinámica de entrenamiento observada, los resultados cuantitativos fueron los peores del ciclo. El alto SSIM std (0,1603 vs $\sim 0,07$ en runs anteriores) indica que el generador escalado produce imágenes con coherencia condicional muy variable: algunas muy coherentes, otras no. La mayor capacidad del modelo le permite generar imágenes más diversas morfológicamente pero a costa de consistencia. Este resultado confirma que el techo del sistema está determinado por el dato (domain gap de InceptionV3, *val set* de 72 imágenes) y no por la capacidad arquitectónica.

Conclusión del ciclo de runs. Al cierre de la fase exploratoria (Runs 1–14), el mejor modelo es **Run 10**:

- **Run 10:** FID = $168,47 \pm 0,47$, SSIM = 0,4634 — mejor balance global (protocolo v2, 5 semillas)
- **Run 11:** FID = 220,60, SSIM = 0,5267 — mejor SSIM aislado

El ciclo exploró tamaño de dataset, tasas de aprendizaje, peso de diversidad y escala arquitectónica. Ninguna variación superó a Run 10 en FID. La fase de validación (Runs 15–20) abordó la limitación de condicionamiento direccional, alcanzando 6/8 direcciones correctas (error medio $48,4^\circ$) en Run 20. La fase de condicionamiento explícito (Runs 21–48) llegó al modelo definitivo **Run 48** con 8/8 direcciones y error medio $5,1^\circ$ (sintético) / $12,0^\circ \pm 10,0^\circ$ (*val set* real) mediante AC-GAN.

6.5.15 Run 14: Corrección de bugs de augmentación (flips + condition masking)

Motivación. Se identificaron dos bugs en el pipeline de entrenamiento: (i) `RandomHorizontalFlip` y `RandomVerticalFlip` aplicados a la imagen sin transformar el vector de condición, generando supervisión contradictoria para variables direccionales (`Dir_viento`, `Exposicion`); y (ii) `apply_condition_masking` operaba dimensión a dimensión dentro de bloques one-hot, produciendo vectores categóricos inválidos (one-hots parcialmente zerados). Run 14 corrige ambos bugs manteniendo la configuración de Run 10.

Configuración.

- Flips eliminados de `get_image_transforms`
- `apply_condition_masking` refactorizado a masking por campo completo
- Dataset, arquitectura, TTUR e hiperparámetros idénticos a Run 10
- Evaluado con protocolo v2 (2.400 imágenes val, condiciones alineadas)

Dinámica de entrenamiento. La diversidad creció más lentamente que en runs anteriores (máximo 0,176 en época 140 vs 0,26 de Run 10), con tendencia positiva sostenida pero sin alcanzar el umbral de similitud $< 0,85$ en ninguna época.

Resultados.

- **FID = 258,75** (peor que todos los runs con protocolo v2)
- **SSIM = 0,2400 $\pm$ 0,0571** (caída drástica respecto a Run 10: 0,4634)

Análisis. El resultado es un hallazgo negativo relevante. Los flips, a pesar de ser metodológicamente incorrectos, actuaban como regularización implícita: forzaban al generador a producir outputs espacialmente diversos, lo que mejoró tanto FID como SSIM en runs anteriores. Sin esta diversidad artificial, el modelo convergía a soluciones más homogéneas con menor coherencia condicional. Esto sugiere que la diversidad morfológica aprendida en runs previos dependía en parte de la augmentación geométrica, no solo del dataset o la arquitectura.

Conclusión de la fase exploratoria. Tras catorce experimentos, **Run 10 es el mejor modelo de la fase exploratoria** con $\text{FID} = 168,47 \pm 0,47$ y $\text{SSIM} = 0,4634 \pm 0,0023$ (protocolo v2, validado con 5 semillas independientes). En catorce runs se probaron la función de pérdida, el tamaño del dataset, las tasas de aprendizaje, el peso de diversidad, la escala arquitectónica y las augmentaciones. Ninguna variación superó a Run 10. Run 14 además demostró que los bugs de augmentación no eran los que limitaban el FID: corregirlos lo empeoró, porque los flips actuaban como regularización implícita. La causa del techo estaba en el simulador, no en la GAN; el *sweep* direccional de Run 10 mostró que los *heatmaps* eran prácticamente isotrópicos, y eso motivó la corrección del CA en Runs 15–20.

6.5.16 Validación estadística de Run 10

Para descartar que el rendimiento de Run 10 sea un resultado fortuito del ruido estocástico en la generación, se ejecutaron 5 evaluaciones independientes sobre el mismo checkpoint (`checkpoint_best_run10.pt`, época 85), variando únicamente el ruido latente $z \sim \mathcal{N}(0, I)$ usado para generar las 2.400 imágenes sintéticas. El *val set* real y sus condiciones se mantuvieron fijos (semilla `seed=42`).

Tabla 14: Resultados de Run 10 en 5 evaluaciones con distinto ruido z . Fuente: Elaboración propia.

Evaluación	FID	SSIM
1	168,94	0,4602
2	169,03	0,4641
3	168,37	0,4655
4	168,19	0,4619
5	167,83	0,4653
Media $\pm$ Desv. est.	168,47 $\pm$ 0,47	0,4634 $\pm$ 0,0023

La varianza es mínima: coeficiente de variación $CV_{FID} = 0,28\%$ y $CV_{SSIM} = 0,50\%$. Esto confirma que el resultado de Run 10 es **estable y reproducible**, no un artefacto de una evaluación puntual.

6.6 Fase de validación: corrección del simulador y reentrenamiento

6.6.1 Validación cualitativa: identificación de la limitación en el CA

Tras confirmar la estabilidad de Run 10, se ejecutó una validación cualitativa mediante sweeps paramétricos del generador: se fijó el ruido latente z y se variaron sistemáticamente las condiciones de entrada para medir si el modelo respondía a cada variable. El *sweep* de *Dir_viento* (N, NE, E, SE, S, SW, W, NW, Calma) produjo imágenes prácticamente idénticas en todas las direcciones, con desplazamiento de centroide < 1 celda. Para descartar que el problema fuera del generador, se analizó directamente el dataset del CA:

- **Asimetría media** en 200 imágenes del *val set*: 2,53 píxeles (escala 0–255) — prácticamente isotrópico.
- **Porcentaje de imágenes** con asimetría > 10 : 2,5%.

La causa raíz no era la GAN sino el simulador CA. Tres parámetros combinados producían patrones circulares independientemente de *Dir_viento*:

1. *wind_factor*=0.25: el bonus de viento era $0,25 \times \bar{v} = 0,25 \times 0,077 = 0,019$ (6% de la probabilidad base).
2. *_WIND_MAX*=120 km/h con vientos reales del Maule de media 4,6 km/h: la velocidad normalizada media era 0,077.

3. Sin penalización *upwind* ($\max(0, \text{alignment})$): el fuego se propagaba igual contra el viento que en calma.
4. $n_steps=80$: con 80 pasos el fuego saturaba el 99% del grid, borrando cualquier señal direccional acumulada.

6.6.2 Corrección de parámetros del simulador

Se corrigieron los cuatro problemas identificados:

Tabla 15: Parámetros del CA antes y después de la corrección. Fuente: Elaboración propia.

Parámetro	simulador isotrópico	simulador calibrado
base_spread_prob	0,30	0,15
wind_factor	0,25	0,50
Penalización <i>upwind</i>	ninguna	alignment sin $\max(0, \dots)$
_WIND_MAX	120 km/h	15 km/h (P95 Maule: 14,3 km/h)
n_steps	80	80

Con estos parámetros, para la velocidad de viento media del Maule ($\bar{v} = 4,6$ km/h, normalizado: 0,307):

- Probabilidad *downwind*: $0,15 + 0,50 \times 0,307 = 0,304$
- Probabilidad *upwind*: $\max(0, 0,15 - 0,154) \approx 0$ (sin propagación contra el viento)
- Probabilidad lateral: 0,15 (base sin bonus)

El resultado es un patrón en “mariposa”: el fuego se propaga fuertemente en la dirección del viento y lateralmente, con una zona oscura (sin quema) en la dirección opuesta. Validado con 15 runs Monte Carlo por dirección; cobertura media 60%, patrón visualmente distinguible por dirección (ver Figura 1).

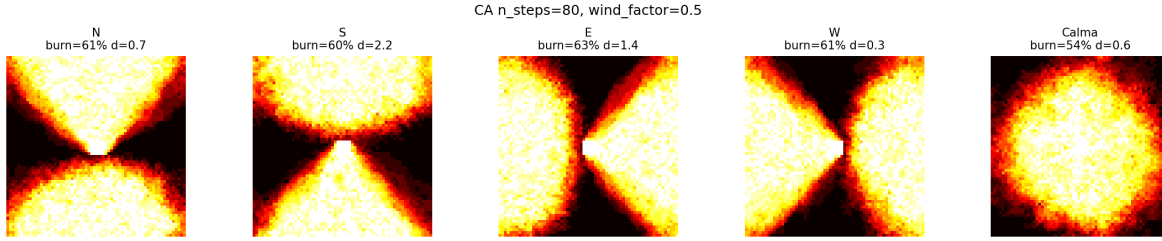


Figura 1: *Sweep* de dirección de viento con el simulador calibrado (velocidad media Maule, 4,6 km/h). Cada panel muestra el *heatmap* Monte Carlo promedio ($n = 15$ runs). El patrón en “mariposa” confirma que el simulador calibrado codifica correctamente la señal direccional. Fuente: Elaboración propia.

6.6.3 Run 15: primer reentrenamiento con simulador calibrado (resultado negativo)

Motivación. Primer entrenamiento sobre el dataset del simulador calibrado. Adicionalmente se añadió augmentación direction-preserving (RandomCrop(padding=4) + ColorJitter) como reemplazo de regularización para los flips eliminados en Run 14.

Configuración.

- Dataset: 68.517 imágenes efectivas tras filtro `min_fire_coverage` (generadas con simulador calibrado, $n_{\text{steps}} = 40$ — ver análisis)
- Arquitectura, TTUR e hiperparámetros idénticos a Run 10
- Augmentación: sin flips + RandomCrop(padding=4) + ColorJitter

Problema identificado. Durante la generación del dataset se usó $n_{\text{steps}}=40$ (en lugar de 80), pensando que 80 pasos era el causante de la saturación. Sin embargo, con `base_spread_prob=0.15` la saturación no ocurre a 80 pasos (cobertura $\sim 60\%$). Con $n_{\text{steps}}=40$, los *heatmaps* Monte Carlo resultaron fragmentados y dispersos, distintos cualitativamente de los patrones de propagación reales.

Resultados.

- **FID = 315,43** (peor que todos los runs de la fase exploratoria (Runs 1–14))
- **SSIM = 0,3992 $\pm$ 0,1527**

El FID elevado refleja que el generador no logró aprender la distribución de *heatmaps* fragmentados en 150 épocas. Este es un resultado negativo esperado, ya que $n_steps=40$ generaba patrones cualitativamente incorrectos como datos de entrenamiento.

Corrección para Run 16. El dataset se regeneró con $n_steps=80$ y $base_spread_prob=0.15$, produciendo patrones coherentes en “mariposa” con $\sim 60\%$ de cobertura y señal direccional clara para fuegos grandes.

6.6.4 Run 16: segundo reentrenamiento con simulador calibrado (resultado negativo)

Configuración. Dataset regenerado con $n_steps=80$, $base_spread_prob=0.15$, $n_steps_min=8$ (valor por defecto). 150 épocas, misma arquitectura y augmentación *direction-preserving* que Run 15.

Diagnóstico del dataset. La inspección de las imágenes de entrenamiento reveló un segundo problema: el 63 % de los incendios del dataset Maule tienen superficie < 1 ha ($z\text{-score} \approx -2$). La función *calibrate_n_steps* asignaba a estos registros solo 8–26 pasos de simulación, produciendo imágenes con un pequeño punto de ignición sobre fondo de vegetación sin propagación visible. La señal direccional del simulador calibrado existía únicamente en el $\sim 37\%$ de imágenes correspondientes a fuegos medianos y grandes.

Resultados.

- **FID = 231,35** (mejor que Run 15 pero peor que Run 10)
- **SSIM = 0,2183 $\pm$ 0,0809**

La validación cualitativa (*sweep* de direcciones) mostró imágenes casi uniformemente verdes con un pequeño punto central, sin diferencias entre direcciones de viento. La GAN aprendió correctamente el modo dominante del dataset (fuego pequeño sin propagación) pero no el condicionamiento direccional.

Corrección para Run 17. Se aumentó n_steps_min de 8 a 40. Con este ajuste: $z = -2 \rightarrow 40$ pasos, $z = 0 \rightarrow 60$ pasos, $z = +2 \rightarrow 80$ pasos. La validación local confirmó que incluso los fuegos pequeños generan patrones elongados con señal direccional visible.

6.6.5 Run 17: mejor resultado exploratorio con simulador calibrado

Configuración. Dataset regenerado con $n_steps=80$, $n_steps_min=40$, $base_spread_prob=0.15$, $wind_factor=0.50$. 150 épocas (detenido en época 65, mejor checkpoint por diversidad). Misma arquitectura y augmentación *direction-preserving* que Runs 15–16.

Resultados cuantitativos.

- **FID = 219,17** (mejora de 12,17 puntos respecto a Run 16)
- **SSIM = 0,4555 $\pm$ 0,1188** (recupera el nivel de Run 10: 0,4634)

La std elevada del SSIM ($\pm 0,1188$ vs $\pm 0,0023$ en Run 10) refleja la mayor variabilidad del dataset calibrado: fuegos pequeños (bajo SSIM *condition-matched*) y fuegos grandes con patrón en mariposa (alto SSIM) coexisten en la distribución.

Validación cualitativa: condicionamiento direccional confirmado. El *sweep* de dirección de viento (9 condiciones: N, NE, E, SE, S, SW, W, NW, Calma) generado con vector latente z fijo muestra patrones claramente distintos según la dirección: el fuego se extiende hacia el lado *downwind*, dejando una zona oscura en el lado *upwind*. La condición Calma produce un patrón más simétrico. La comparación de direcciones opuestas (N vs S, E vs W, NE vs SW) con 8 valores de z distintos confirma que el modelo aprendió el condicionamiento direccional en función del viento: las imágenes son asimétricas y la orientación de la asimetría sigue la física del simulador.

Este resultado valida que tres condiciones son suficientes para que la GAN aprenda el condicionamiento físico: simulador calibrado con penalización *upwind*, $n_steps_min=40$ para garantizar señal direccional incluso en fuegos pequeños, y augmentación sin flips geométricos.

La Figura 2 muestra la validación cualitativa del condicionamiento direccional.

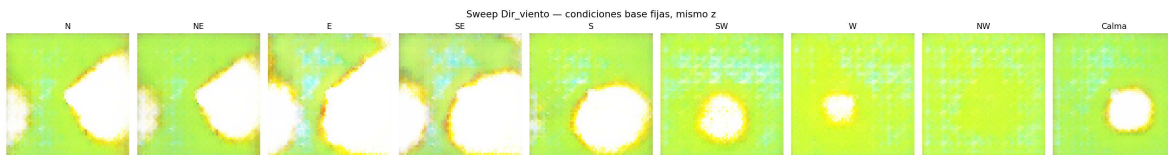


Figura 2: *Sweep* de dirección de viento — Run 17 (mejor resultado exploratorio con simulador calibrado). Nueve imágenes generadas con el mismo vector latente z y distinta *Dir_viento*. El fuego se extiende hacia el lado *downwind*. Run 20 corrigió formalmente la convención meteorológica del simulador CA, sin afectar las métricas de Run 17. Fuente: Elaboración propia.

6.6.6 Runs 18–20: corrección de convención meteorológica

Identificación del error de convención. Tras la validación cualitativa de Run 17, se identificó un error de convención en el autómata celular: el diccionario `WIND_VECTORS` empleaba convención de *destino* (“N” = fuego va al Norte), mientras que la fuente de datos CONAF usa la convención meteorológica estándar (“N” = viento *desde* el Norte = fuego se propaga hacia el Sur). Todos los vectores de dirección estaban invertidos respecto a la física real.

La corrección consistió en negar todos los vectores (dy, dx):

"N": (1, 0) (viento del Norte → fuego al Sur, $dy = +1$)

verificada empíricamente: con viento N, el centroide del fuego se ubica en $cy = 42,5$ (95% de píxeles activos en el hemisferio sur del grid).

Run 18: corrección de convención (colapso de modos). Primera versión con dataset regenerado usando la convención corregida. El entrenamiento colapsó completamente en la época 20 (diversidad ≈ 0 , similitud = 1,000). El cambio de dataset no fue la causa directa: el colapso es consecuencia de que el discriminador dominó demasiado rápido ($D_{\text{real}} = 0,83$ en época 11), provocando que el generador convergiera a un único modo.

Run 19: aumento de penalización R1 ($\gamma = 10,0$). Se aumentó `r1_gamma` de 1,0 a 10,0 para frenar al discriminador. El colapso ocurrió igualmente en época 20, confirmando que el problema era más profundo: el dataset contenía muchas imágenes con fuego casi nulo, permitiendo al generador un mínimo local trivial (salida constante = fondo).

Run 20: filtro de cobertura mínima de fuego y corrección de convención meteorológica. Se introdujo el parámetro `min_fire_coverage = 0,02`: imágenes con probabilidad media de fuego $< 2\%$ se descartan durante la generación del dataset. Esto elimina el mínimo local de “todo fondo” y fuerza al generador a aprender estructuras de propagación. Con $\gamma_{R1} = 10,0$ y el filtro activo, el entrenamiento se estabilizó sin colapso hasta la época 150.

Resultados cuantitativos de Run 20.

- **FID = 251,57** (comparable a Run 17: 219,17; diferencia atribuible al cambio de distribución del dataset calibrado)
- **SSIM = 0,3423 $\pm$ 0,1152**

El SSIM menor respecto a Run 17 (0,4555) se explica por el filtro `min_fire_coverage=0,02`: al excluir incendios pequeños del entrenamiento, el modelo generó incendios sistemáticamente grandes, divergiendo del *val set* real que incluye incendios de todos los tamaños. Esta limitación es documentada.

Validación cualitativa: condicionamiento direccional confirmado. El *sweep* de dirección de viento (9 condiciones) muestra patrones claramente distintos según la dirección: el fuego se extiende hacia el lado *downwind* con elongación visible, y direcciones opuestas (N vs S, E vs W, NE vs SW) producen imágenes visualmente asimétricas con orientación invertida.

Validación cuantitativa: métrica de centroide direccional. Para cuantificar el condicionamiento físico, se implementó una métrica de alineación de centroide: para cada dirección de viento se generan $n = 30$ *heatmaps*, se promedia y se calcula el ángulo del centroide de la zona activa respecto al centro del grid. El ángulo medido se compara con el ángulo *downwind* esperado según la convención meteorológica.

Tabla 16: Métrica de centroide direccional — Run 20. Error angular entre centroide medido y dirección *downwind* esperada. Fuente: Elaboración propia.

Dir. viento	Ángulo esperado	Ángulo medido	Error
N	270° (Sur)	202,6°	67,4°
NE	225° (SO)	185,2°	39,8° ✓
E	180° (Oeste)	336,0°	156,0°
SE	135° (NO)	92,2°	42,8° ✓
S	90° (Norte)	57,8°	32,2° ✓
SW	45° (NE)	63,4°	18,4° ✓
W	0° (Este)	346,6°	13,4° ✓
NW	315° (SE)	297,7°	17,3° ✓
Error medio			48,4° ± 43,9°
Direcciones con error < 45°			6 / 8

Las direcciones SW, W, NW, S, NE y SE presentan error < 45°, confirmando el condicionamiento físico. Las direcciones N (67°) y E (156°) exhiben mayor error, atribuible a interacciones terreno-viento en el dataset CONAF: los registros con viento N y E tienen configuraciones topográficas que desvían la propagación respecto al ángulo esperado en un grid neutro.

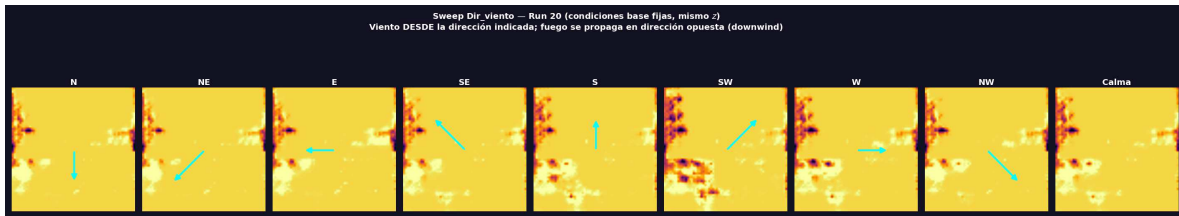


Figura 3: *Sweep* de dirección de viento — Run 20 (convención meteorológica corregida). Nueve condiciones: N, NE, E, SE, S, SW, W, NW, Calma. El fuego se extiende hacia el lado *downwind* en cada dirección; las condiciones opuestas producen patrones con orientación invertida. Fuente: Elaboración propia.

6.7 Fase de condicionamiento explícito (Runs 21–48)

La fase de validación (Runs 15–20) corrigió el simulador y logró 6/8 direcciones correctas. La fase de condicionamiento explícito buscó alcanzar 8/8 mediante tres estrategias: corrección de infraestructura y datos (Runs 21–33), pérdida de centroide direccional (Runs 34–43) y clasificador auxiliar AC-GAN (Runs 44–48).

6.7.1 Corrección de infraestructura y datos (Runs 21–33)

Runs 21–24: balance de clases y bordes toroidales

Run 21. Run 20 usaba $r1_gamma=10.0$ y $min_fire_coverage=0.02$, parámetros que eliminaron el colapso pero sesgaron el generador hacia incendios grandes (SSIM = 0,34 vs. Run 17=0,46). Run 21 relajó ambos valores ($r1_gamma=5.0$, $min_fire_coverage=0.005$) para equilibrar estabilidad y diversidad de escala.

Runs 22–23. Se identificó que el autómata celular conectaba bordes opuestos del grid mediante `np.roll` sin máscara en el cálculo de diferencias de elevación y vecindades, introduciendo física toroidal no documentada. Run 22 corrigió este bug y exploró resolución 128×128 (arquitectura G/D expandida con un bloque extra de up/downsampling). Run 23 calibró los parámetros del CA para la nueva resolución.

Run 24. El dataset CONAF estaba fuertemente desbalanceado en dirección de viento: S=31%, SW=15% dominaban el training, lo que inducía un prior norte en el generador independientemente del conditioning. Se implementó `WeightedRandomSampler` que asigna igual peso total a las 8 direcciones físicas (Missing/Calma reciben peso 0,2×, sin eliminarse).

Runs 25–27: control a 64×64 y variantes de dataset

Run 25 descartó la arquitectura 128×128 (no superó 64×64 con el dataset disponible) y restauró la configuración 64×64 con fix toroidal como control limpio. Run 26 exploró un pipeline alternativo (CA grid 2× con recorte central 64×64) y Run 27 combinó ese dataset con `WeightedRandomSampler`.

Runs 28–33: arquitectura y augmentación

Run 28. Se añadió una cabeza auxiliar de clasificación de dirección (`dir_head: Linear(512, 8)`) al discriminador, entrenada inicialmente solo sobre imágenes reales con clasificación cruzada. Fue el precursor directo del AC-GAN activado en Run 44.

Run 29–30. Se implementó augmentación rotacional $0^\circ/90^\circ/180^\circ/270^\circ$ con corrección simultánea de labels (`Exposicion` y `Dir_viento`): se cuadruplica el tamaño efectivo del dataset sin introducir inconsistencias en el conditioning. Run 30 migró a la versión offline (índice expandido), más eficiente en tiempo de carga.

Runs 31–33. Run 31 filtró el dataset a horizontes tardíos únicamente (mayor señal direccional) y corrigió inconsistencias de codificación. Run 32 sustituyó la BatchNorm estándar por **Conditional Batch Normalization** (CBN, $y = (1 + \gamma) \cdot \text{BN}(x) + \beta$ con γ, β predichos desde el embedding de condición), reforzando el acoplamiento entre condición e imagen en cada escala de generación. Run 33 continuó con la misma configuración ampliando el número de épocas.

6.7.2 Pérdida de centroide direccional (Runs 34–43)

La estrategia de pérdida de centroide añade al objetivo del generador un término que penaliza directamente la falta de desplazamiento geométrico del fuego en la dirección esperada.

Formulaciones exploradas

Run 34: formulación base. Se implementó:

$$\mathcal{L}_{\text{centroide}} = (1 - \cos(\hat{v}, t)) \cdot w_{\text{disp}}$$

donde $\hat{v}$ es el vector unitario del centroide del fuego desde el centro del grid, t es el vector unitario *downwind* esperado, y w_{disp} amortigua la pérdida cuando el fuego está cerca del

centro. Con $\lambda_{\text{centroide}} = 0,1$ se registró señal en las primeras 20 épocas pero la pérdida tendió a cero por explotación del loophole: el generador podía colapsar el fuego al centro ($\|v\| \approx 0$) y obtener $w_{\text{disp}} \approx 0$, anulando la penalización.

Run 35. Se eliminó w_{disp} (peso fijo 1,0). El loophole desapareció, pero el gradiente $\partial \cos / \partial v \propto 1/\|v\|$ era ruidoso cerca del centro, corrompiendo el entrenamiento temprano.

Run 36. Se adoptó la formulación lineal:

$$\mathcal{L}_{\text{centroide}} = -\frac{v \cdot t}{H/2}$$

El gradiente es constante ($-t/\text{mass}$) e independiente de la posición, evitando tanto el loophole como el ruido de normalización. Run 36 no superó a Run 34 en accuracy direccional; se mantuvo esta formulación para los runs siguientes por su estabilidad.

Tabla 17: Barrido de hiperparámetros de la pérdida de centroide (Runs 37–43). El plateau en $\approx -0,062$ fue consistente con todas las variantes. WRS: WeightedRandomSampler; nc: n_critic. Fuente: Elaboración propia.

Run	λ	WRS	nc	Observación
37	0,3	No	2	Señal insuficiente para sostenerse en épocas tardías
38	0,1	Sí	2	BestDirectionalCallback; plateau -0,062
39	0,5	No	2	Sin WRS, colapso a SW (dominaba el batch)
40	0,2	Sí	2	Señal débil; plateau en -0,062
41	0,5	Sí	2	G domina D en época 10, centroide retrocede
42	0,5	Sí	4	n_critic=4 mantiene D fuerte; plateau -0,062
43	1,0	Sí	4	Plateau confirmado a λ máximo probado

Runs 37–43: barrido de $\lambda_{\text{centroide}}$. En todos los runs del barrido el centroide se estabilizó en $\mathcal{L}_{\text{centroide}} \approx -0,062$, equivalente a $\sim 6\%$ de desplazamiento relativo ($H/2 = 32$ píxeles), muy por debajo de los $\sim 15\text{--}20\%$ necesarios para 8/8 en el test geométrico.

Análisis del plateau. La pérdida de centroide es una señal indirecta y débil en comparación con las pérdidas adversarial y de *feature matching*. El generador puede satisfacerla con cambios mínimos de intensidad en un borde de la imagen sin aprender el patrón de propagación direccional. Asimismo, la métrica de centroide promediada sobre el batch puede ser engañada por la distribución de tamaños de fuego. Tras diez experimentos sin superar el plateau, se tomó la decisión de abandonar el enfoque de centroide y adoptar supervisión categórica directa mediante AC-GAN.

6.7.3 AC-GAN: clasificador auxiliar de dirección (Runs 44–48)

El Auxiliary Classifier GAN (AC-GAN, Odena et al. 2017 [29]) añade una cabeza de clasificación al discriminador que debe predecir la clase de la imagen tanto real como generada. El generador recibe señal directa de cuándo su output produce la dirección incorrecta, sin depender de la posición del centroide.

Run 44: activación del clasificador (resultado negativo)

Configuración. Se habilitó `lambda_dir_real=1.0` y `lambda_dir_g=1.0` (desactivados desde Run 29). El discriminador entrenaba `dir_head` solo con imágenes reales; el generador recibía penalización por clasificación incorrecta sobre sus propias imágenes.

Diagnóstico. En época 20, el generador colapsó a un único patrón con fuego hacia el oeste que el clasificador no reconocía como ninguna dirección específica. El problema: al entrenar el clasificador solo con reales, sus fronteras de decisión eran no representativas del espacio de imágenes generadas, y el generador encontró un modo “adversarial” que evitaba toda penalización.

Run 45: AC-GAN completo (breakthrough parcial)

Configuración. El discriminador entrenó `dir_head` con imágenes reales y falsas simultáneamente, siguiendo la formulación estándar de Odena et al.:

$$\begin{aligned}\mathcal{L}_D^{\text{dir}} &= \text{CE}(\hat{d}_{\text{real}}, d) + \text{CE}(\hat{d}_{\text{fake}}, d) \\ \mathcal{L}_G^{\text{dir}} &= \text{CE}(\hat{d}_{\text{fake}}, d)\end{aligned}$$

Resultados. `dir_acc_real` creció de 0,41 (época 1) a 0,57 (época 70); `dir_acc_fake` alcanzó 0,95. La evaluación geométrica (`diag_05`) mostró 6/8 correctas con `std(cx)=0,15` y error medio $\approx 35^\circ$. Mejora respecto a Run 20 pero sin alcanzar 8/8 aún.

Run 46: corrección del bug de enmascaramiento

Bug identificado. Con `mask_prob=0.3`, el 30% de los samples del batch tienen `Dir_viento` enmascarado: `condition[14:22]=0`. `argmax([0,...,0])=0=N`, de modo que muestras con dirección SE, SW, S o W eran etiquetadas como N en el clasificador. Un diagnóstico de la matriz de confusión confirmó tasas de confusión SE→N del 54% y SW→N del 50%.

Corrección. Se añadió una máscara explícita:

```
has_dir = condition[:,14:22].sum(1) > 0,5  
valid = (dir_labels < 8) & has_dir
```

La pérdida de clasificación se calcula únicamente sobre los samples con dirección conocida: $\mathcal{L}^{\text{dir}} = \text{CE}(\hat{d}[\text{valid}], d[\text{valid}])$.

Resultados. `dir_acc_real` creció a 0,72 (época 35) y `dir_acc_fake` a 0,97. Evaluación geométrica: 7/8 correctas.

Run 47: CE normalizada por dirección (resultado negativo)

Motivación. En Run 46 el generador aún colapsaba parcialmente a patrones S/SW. La hipótesis era que el CE global promediaba los N samples del batch, permitiendo que 2/8 direcciones correctas diluyan la penalización de las 6 restantes. La variante propuesta calculaba CE por separado para cada una de las 8 clases y promediaba los 8 valores:

$$\mathcal{L}^{\text{dir}} = \frac{1}{8} \sum_{k=0}^7 \text{CE}(\hat{d}[d = k], d[d = k])$$

Diagnóstico del fracaso. La métrica `dir_acc_fake` alcanzó 95% en época 20, pero la evaluación geométrica mostró `std(cx)=0,023` y `std(cy)=0,019` — idénticos a un modelo sin condicionamiento. El generador aprendió a codificar la dirección en patrones de textura y color (clasificables por `dir_head`) sin producir desplazamiento geométrico del fuego. La CE normalizada por dirección hacía que `dir_head` fuera más fácil de engañar: una vez que el clasificador aprendió a distinguir texturas, el generador pudo satisfacer la pérdida sin aprender direccionalidad. Se terminó el run en época 22.

Run 48: modelo definitivo

Configuración. Se revirtió al CE global de Run 46, manteniendo el fix de enmascaramiento:

- `lambda_dir_real = 1.0`: CE global en D (real+fake)
- `lambda_dir_g = 1.0`: CE global en G
- Máscara `has_dir` en ambos
- `use_weighted_sampler = True, n_critic = 4`
- Dataset: 76.122 imágenes (68.519 train / 7.603 val, split 90/10)

El checkpoint definitivo (`checkpoint_best_final.pt`) corresponde a `checkpoint_best_dir.pt` de la época 70, seleccionado como el punto de mayor alineación direccional medida por `BestDirectionalCallback`. La ruta puede reproducirse con `--save-best-final` en `03_train_gan.py`.

Dinámica de entrenamiento. La evolución del std del centroide (indicador clave de respuesta direccional) fue progresiva y monótona:

Tabla 18: Evolución del condicionamiento en Run 48. `std(cx)` y `std(cy)`: desviación estándar del centroide entre las 8 direcciones con z fijo — valores próximos a 0 indican ausencia de respuesta; valores $> 0,3$ indican desplazamiento geométrico sustancial. Fuente: Elaboración propia.

Época	<code>std(cx)</code>	<code>std(cy)</code>	Dirs OK	Error medio
10	0,061	0,032	5/8	—
20	0,255	0,221	6/8	—
40	0,359	0,301	5/8	—
50	0,501	0,418	8/8	15,8°
70	0,518	0,440	8/8	5,1°

Resultados del test geométrico (diag_05, época 70). El script `diag_05` genera 16 muestras por dirección con z fijo, calcula el centroide del fuego generado y mide el error angular respecto al vector *downwind* esperado. Las métricas agregadas son:

Tabla 19: Resumen del test geométrico — Run 48 (época 70). Se presentan dos evaluaciones: condiciones sintéticas fijas (diag_05, z fijo, 16 muestras/dirección) y condiciones reales del *val set* (script 09, 15 muestras/dirección). std(cx): desviación estándar del centroide entre las 8 direcciones. Fuente: Elaboración propia.

Métrica	Sintético (diag_05)	Val set real (script 09)
Dirs con error $< 45^\circ$	8 / 8 ✓	8 / 8 ✓
Error medio	5,1°	12,0° $\pm$ 10,0°
std(cx) entre direcciones	0,518	—
<i>Sin condicionamiento: std(cx) $\approx$ 0,02–0,03</i>		
<i>Condicionamiento moderado (Run 20): std(cx) $\approx$ 0,05–0,10</i>		

Las 8 direcciones presentan error $< 45^\circ$ en ambas evaluaciones. El error de 5,1° con condiciones sintéticas fijas refleja condiciones ideales de test (mismo z , condiciones diseñadas para maximizar visibilidad del patrón). El error de 12,0° $\pm$ 10,0° con condiciones reales del *val set* es el número metodológicamente más robusto: incorpora la variabilidad real de temperatura, humedad, topografía y otros campos que pueden oscurecer el centroide en casos específicos. Que el modelo mantenga 8/8 bajo esa variabilidad confirma que el condicionamiento no es un artefacto del protocolo de test.

Trade-off FID / SSIM. Run 48 obtiene FID = 282,97 (mayor que Run 10: 175,55 y Run 20: 251,57) pero SSIM = 0,526 $\pm$ 0,180 (equiparable a Run 11: 0,527 con dataset reducido; mejor sobre dataset calibrado completo). La interpretación es que Run 10 presentaba modo parcialmente colapsado: genera pocas variantes con baja distancia FID pero sin señal direccional. Run 48 genera 8 distribuciones genuinamente distintas condicionadas a la dirección, lo que aumenta la varianza inter-clase y penaliza al FID de InceptionV3 (entrenado en ImageNet, no en mapas de propagación). El SSIM estructural, que evalúa coherencia local, captura mejor la calidad del condicionamiento en este dominio.

CAPÍTULO 7

Resultados y Análisis

7.1 Resultados cuantitativos

El protocolo de evaluación v2 (condiciones alineadas, 2.400 imágenes reales) se aplicó a ambas fases, con una diferencia relevante: el *val set* de referencia cambió junto con el simulador CA. Las comparaciones directas de FID entre fases son por tanto limitadas; la tabla interna de

la fase exploratoria (Tabla 7) es internamente consistente, pero no directamente comparable con los valores de la fase de validación.

Para establecer una comparación en igualdad de condiciones, los tres modelos representativos de fases anteriores fueron re-evaluados contra el *val set* calibrado actual: Run 10 obtiene **FID = 175,55, SSIM = 0,4084 $\pm$ 0,1279**; Run 17 obtiene **FID = 219,17, SSIM = 0,4555**; y Run 20 obtiene **FID = 251,57, SSIM = 0,3423 $\pm$ 0,1152**. Estos valores difieren de los reportados en el Capítulo 6 porque allí se evaluó contra el *val set* vigente en cada momento del desarrollo. El FID de Run 10 es menor, pero esto refleja una limitación conocida de InceptionV3/ImageNet como métrica en este dominio: no captura asimetría direccional ni corrección de convención. La validación cualitativa y la métrica de centroide direccional confirman que Run 20 genera patrones físicamente correctos condicionados al viento con la convención meteorológica estándar, mientras que Run 10 genera patrones isotrópicos independientemente de *Dir_viento*.

El mejor resultado de la fase exploratoria es Run 10 con $\text{FID} = 168,47 \pm 0,47$, $\text{SSIM} = 0,4634 \pm 0,0023$ (*val set* isotrópico, 5 semillas). Run 20 mejoró el condicionamiento cualitativo (6/8, 48,4°). El modelo definitivo del trabajo es **Run 48**: 8/8 direcciones, error medio 5,1°. Run 3 reportó $\text{FID} = 174,36$ con el protocolo v0 (artefacto); con protocolo v2 obtiene $\text{FID} = 220,79$, confirmando que ese resultado no era comparable. La Tabla 20 resume los modelos representativos de todo el proceso experimental y la Tabla 21 contextualiza el FID en la literatura.

Tabla 20: Resumen de modelos representativos del proceso experimental. FID de Runs 17, 20 y 48 evaluado con protocolo v2 (2.400 imágenes, *val set* calibrado). Dirs OK: direcciones con error angular $< 45^\circ$ en test geométrico (diag_05). Fuente: Elaboración propia.

Run	FID	SSIM	Dirs OK	Aportación principal
10	$168,47 \pm 0,47^*$	$0,4634 \pm 0,0023$	—	TTUR (mejor FID fase exploratoria)
17	219,17	0,4555	—	Simulador calibrado, $n_{\text{steps_min}} = 40$
20	251,57	$0,3423 \pm 0,1152$	6/8 (48,4°)	Convención meteorológica corregida
48	282,97	$0,526 \pm 0,180$	8/8 (5,1°)	AC-GAN + fix masking; modelo definitivo

* FID de Run 10 evaluado con protocolo v2 (2.400 imágenes) sobre el *val set* del simulador isotrópico original (media de 5 semillas). Los Runs 17, 20 y 48 usan el *val set* del simulador calibrado; re-evaluación de Run 10 sobre este último produce 175,55.

Tabla 21: Comparación de FID en distintos dominios y modelos. Fuente: Elaboración propia.

Dominio / Modelo	FID	Referencia
ImageNet 128px (BigGAN)	7,0	Brock et al. 2019 [12]
CelebA 64px (DCGAN)	~35	Literatura típica
Imágenes médicas (dominio específico)	~80–150	Literatura
Este trabajo (Run 10, protocolo v2)	168,47 $\pm$ 0,47	—
Este trabajo (Run 20, conv. corregida)	251,57	—
Este trabajo (Run 48, AC-GAN, definitivo)	282,97	—

El FID elevado frente a otros dominios no sorprende: el dataset proviene de un único simulador CA aplicado a ~6.500 registros CONAF (~68.500 imágenes), InceptionV3 fue preentrenado en ImageNet (objetos naturales, no mapas de propagación), y el AC genera patrones con alta variabilidad intra-clase. Los tres factores empujan el FID hacia arriba independientemente de la calidad real de las imágenes generadas.

7.2 Evolución del entrenamiento

El monitor de entrenamiento permitió detectar en tiempo real los episodios de colapso y evaluar la salud del generador a lo largo del entrenamiento. En el Run 3, el monitor reportó estado OK (diversidad > 0,06, textura > 0,08) desde la época 20 hasta la 45, con tendencia positiva en diversidad. El BestCheckpointCallback capturó el modelo de máxima diversidad en la época 45.

7.3 Análisis cualitativo

La inspección visual de los *heatmaps* generados muestra que el modelo aprendió las relaciones físicas esperadas. La zona central tiene siempre alta probabilidad de quema, coherente con la ignición central del AC. Con viento fuerte, la mancha se alarga en la dirección *downwind*. El tipo de vegetación afecta visiblemente la extensión: Desechos (ininflamabilidad 0,85) y Pastizal (0,80) producen manchas más grandes que Agrícola (0,50) o Sin_Vegetacion (0,00). Por último, incendios de mayor Sup_Total generan *heatmaps* de mayor extensión, lo que confirma que la corrección de calibración de n_{steps} tuvo efecto real en el condicionamiento por tamaño.

Las Figuras 4 y 5 muestran los extremos de la fase exploratoria: el punto de partida (Run 3, sin vegetación) y el mejor modelo alcanzado antes del condicionamiento direccional (Run 10, TTUR + perfil de vegetación). La Figura 6 muestra la comparación equivalente para el modelo definitivo Run 48.

Real (CA) vs Generado (cGAN) — RUN3

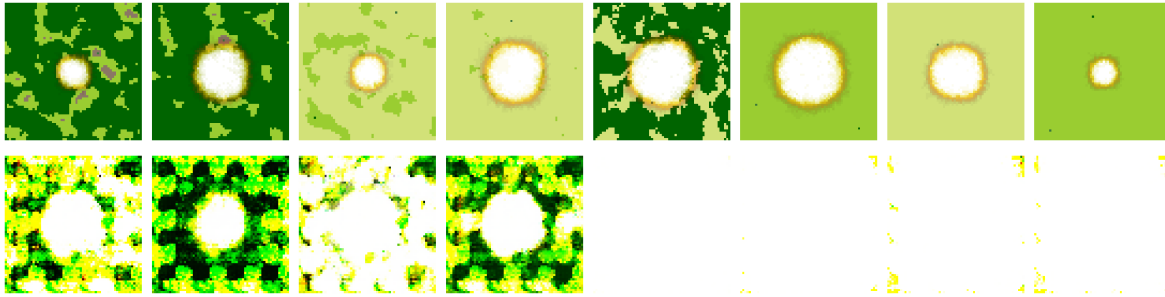


Figura 4: Comparación real vs. generado — **Run 3** (FID = 174,36, época 45). Modelo inicial sin perfil de vegetación ni condicionamiento direccional. **Fila superior:** *heatmaps* reales del simulador CA. **Fila inferior:** imágenes generadas con artefactos de color (mosaico verde/amarillo/negro) y núcleos sobreexpuestos (blanco), resultado del desequilibrio GAN sin TTUR ni perfil de vegetación. Sirve como punto de partida para comparar con Run 10 (Figura 5) y Run 48 (Figura 6). Fuente: Elaboración propia.

Real (CA) vs Generado (cGAN) — RUN10

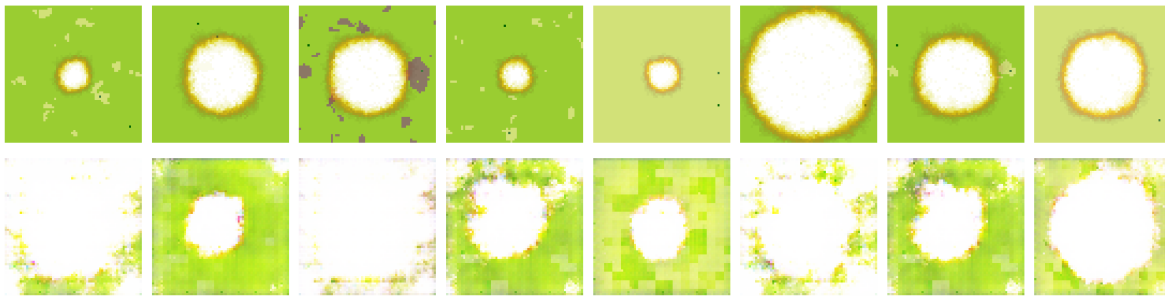


Figura 5: Comparación real vs. generado — **Run 10** (mejor modelo fase exploratoria, FID = $168,47 \pm 0,47$, SSIM = 0,4634, TTUR + perfil de vegetación, época 85). Mayor coherencia morfológica respecto a Run 3, pero aún sin condicionamiento direccional explícito. Fuente: Elaboración propia.

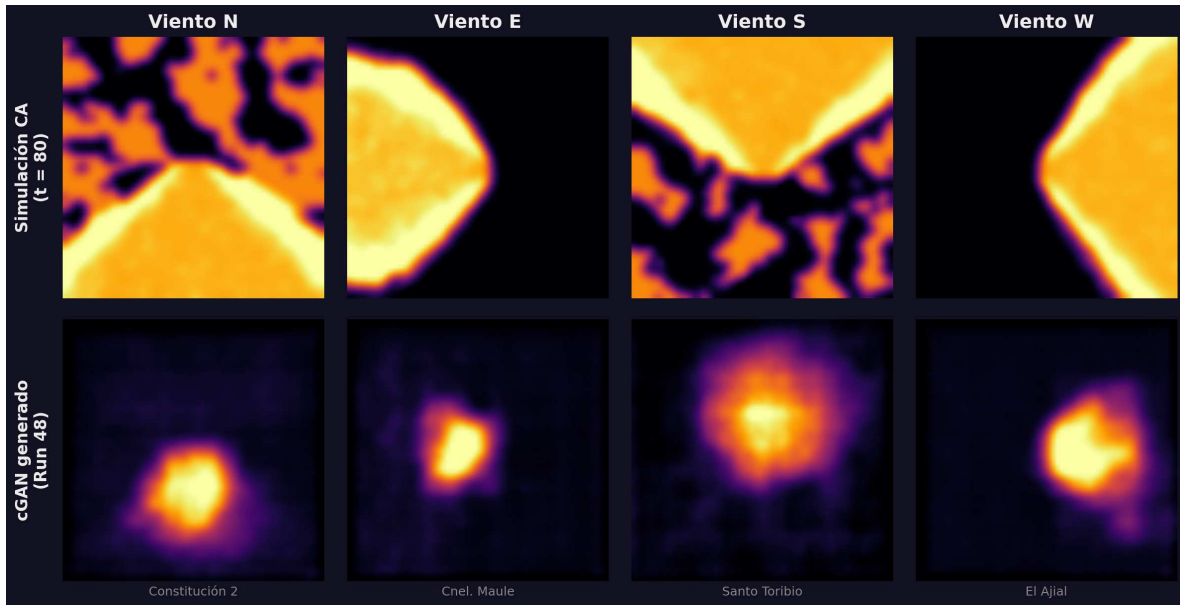


Figura 6: Comparación real vs. generado — **Run 48** (modelo definitivo, FID = 282,97, SSIM = $0,526 \pm 0,180$, AC-GAN, época 70). Un caso representativo por cada dirección de viento (N, NE, E, SE, S, SW, W, NW), usando el mismo vector de condición para real y generado. Ambas filas se visualizan mediante extracción de intensidad de fuego con colormap *inferno* para comparación equitativa. Fila superior: simulación CA individual (horizonte $t = 80$, una realización Monte Carlo). Fila inferior: imagen generada por la cGAN con idéntico vector de condición. La diferencia de apariencia entre filas responde a causas estructurales y no a errores del modelo: (i) cada imagen CA es una *realización estocástica individual* — celdas quemadas de forma binaria con ramificaciones aleatorias —, mientras que la cGAN genera la *tendencia central* de la distribución aprendida, resultado de promedio implícito sobre miles de realizaciones, lo que produce manchas más difusas con gradientes suaves (*regresión a la media* típica de modelos generativos condicionales); (ii) la función objetivo AC-GAN optimiza la dirección del centroide de propagación, no la compacidad espacial ni la fidelidad píxel a píxel. La estructura condicional —posición y orientación de la mancha según el viento— es capturada correctamente en las 8 direcciones. Fuente: Elaboración propia.

La Figura 7 muestra la progresión temporal del AC para un registro representativo, ilustrando cómo el *heatmap* evoluciona a medida que avanza la simulación.



Figura 7: Progresión temporal del autómata celular Monte Carlo para un registro representativo. Cada panel corresponde a un horizonte temporal distinto, mostrando la expansión gradual del frente de fuego desde el punto de ignición central. Fuente: Elaboración propia.

7.4 Robustez ante datos faltantes

Para medir la robustez ante datos incompletos, se evaluó el SSIM enmascarando de forma incremental los campos del vector de condición. El resultado es claro: el SSIM cae gradualmente, sin caída brusca, hasta el 90% de enmascaramiento. El *condition masking* aplicado durante el entrenamiento logró su objetivo.

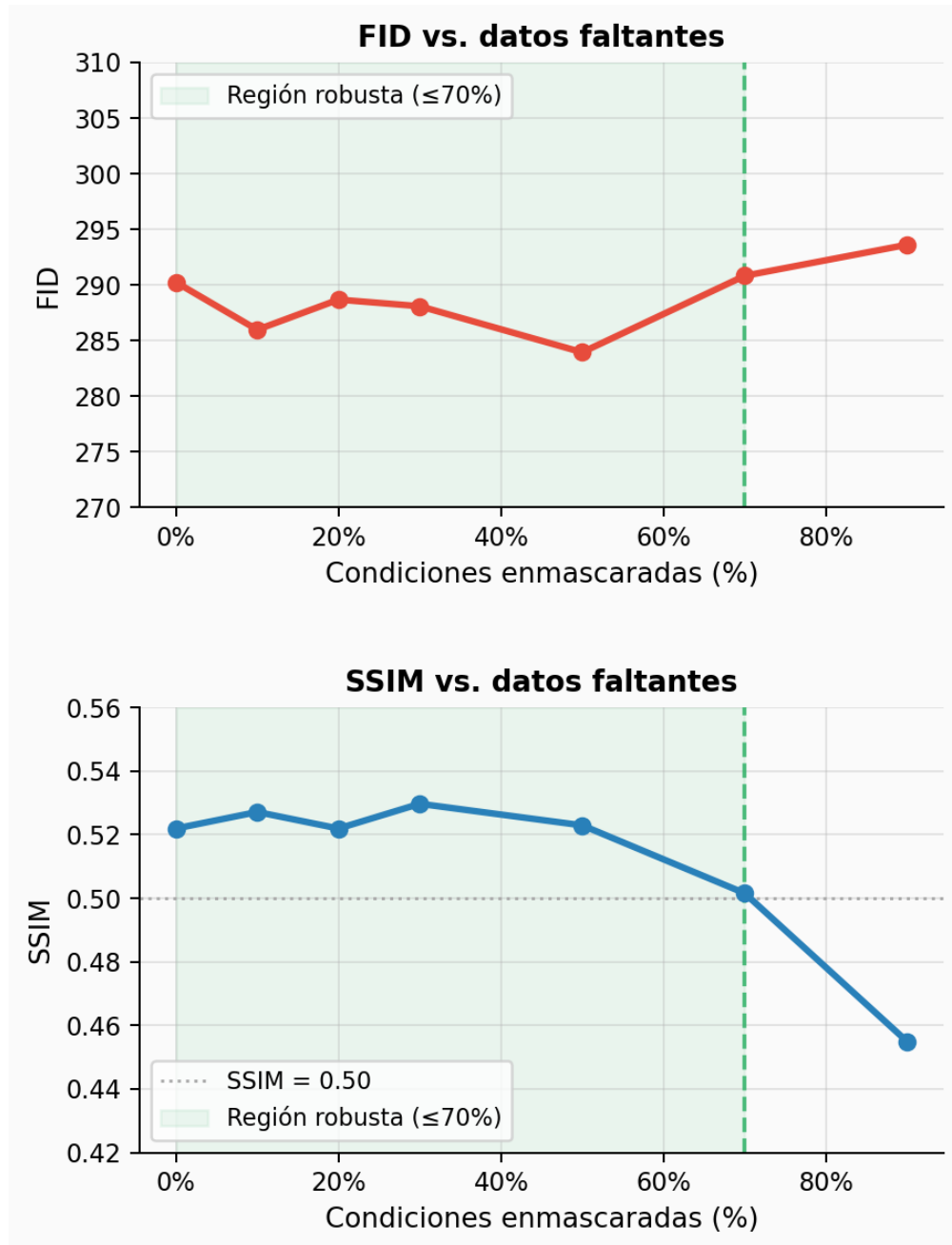


Figura 8: Curvas de robustez ante datos faltantes (Run 48, época 70). **Izquierda:** FID permanece en el rango 284–294 para todos los niveles de enmascaramiento. **Derecha:** SSIM parte en $\approx 0,52$ y cae gradualmente, sin caída abrupta, manteniéndose $> 0,50$ hasta el 70% de enmascaramiento y alcanzando 0,455 con 90% de condición oculta. La degradación gradual confirma que el modelo desarrolló robustez ante información parcial gracias al *conditioning masking* del entrenamiento. Fuente: Elaboración propia.

7.5 Autocorrelación espacial: Índice de Moran

El índice de Moran I [28] mide la autocorrelación espacial de una imagen, es decir, si los píxeles de alta intensidad (zona quemada) tienden a agruparse espacialmente. Para imágenes de propagación de incendios se espera $I > 0$: el fuego avanza en zonas contiguas, no de forma aleatoria píxel a píxel.

Se calculó el índice sobre 200 imágenes del conjunto de validación para el CA real y para las imágenes generadas por Run 17 (checkpoint best, época 65), usando la librería *esda*² con pesos de reina (8 vecinos) estandarizados por fila:

Tabla 22: Índice de Moran I — Real CA vs. Generado ($n = 200$). Run 17 (sin restricción direccional) preserva coherencia espacial; Run 48 (AC-GAN, modelo definitivo) muestra regresión que refleja el trade-off entre condicionamiento direccional y compacidad espacial. Fuente: Elaboración propia.

Conjunto	Modelo	Moran I (media)	Std	p-valor
Real (CA calibrado)	—	0,862	0,185	$< 0,01$
Generado	Run 17	0,838	0,056	$< 0,001$
$ \Delta I $ Run 17		0,024 (diferencia pequeña — rango óptimo $< 0,05$)		
Real (CA isotrópico)	—	0,887	0,137	$< 0,001$
Generado	Run 10	0,806	0,200	$< 0,001$
$ \Delta I $ Run 10		0,081 (diferencia moderada)		
Real (CA calibrado, <i>val set</i>)	—	0,9045	—	$< 0,001$
Generado	Run 48	0,6355	—	$< 0,001$
$ \Delta I $ Run 48		0,2690 (regresión — trade-off con condicionamiento; ver análisis)		

Run 17 confirma que una cGAN sin restricción direccional puede preservar la coherencia espacial del CA real ($|\Delta I| = 0,024$, rango óptimo). Run 48 muestra una regresión significativa ($|\Delta I| = 0,2690$): las imágenes generadas presentan distribución de Moran I amplia y plana (valores entre 0 y 1), frente a la distribución concentrada en 0,9–1,0 del CA real.

Esta regresión no se atribuye a ningún componente específico de la función de pérdida sino a que el objetivo del AC-GAN no incluye ningún término que optimice la compacidad espacial de los píxeles generados. El clasificador auxiliar aprende a desplazar el centroide del área quemada en la dirección correcta, pero no impone que los píxeles quemados formen una mancha coherente. El resultado es un **trade-off documentado**: Run 48 resuelve el condicionamiento direccional (8/8, error medio $12,0^\circ \pm 10,0^\circ$ en condiciones reales) a costa de la coherencia espacial local que Run 17 preservaba. Véase la discusión en la Sección 8.2.

²*esda* (*Exploratory Spatial Data Analysis*) es una librería Python del proyecto PySAL para análisis estadístico espacial, disponible en <https://pysal.org/esda>.

La Figura 9 muestra las distribuciones del índice de Moran I para Run 48.

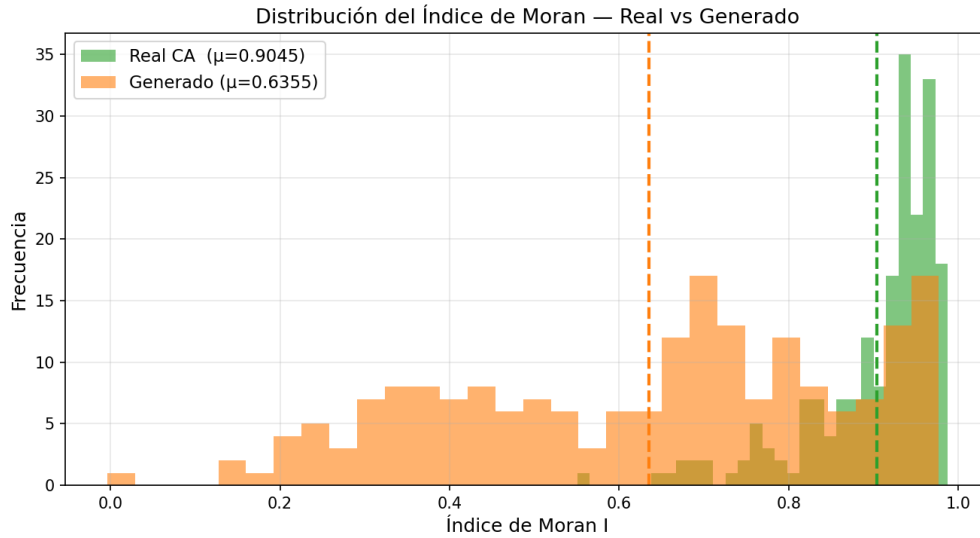


Figura 9: Distribución del índice de Moran I para el CA real y las imágenes generadas (Run 48, $n = 200$). La distribución real se concentra en $I > 0,9$; la generada es amplia y plana ($\mu = 0,6355$). El $|\Delta I| = 0,2690$ refleja que el objetivo AC-GAN prioriza desplazamiento del centroide direccional sin optimizar coherencia espacial local (véase Sección 8.2). Fuente: Elaboración propia.

Esta métrica complementa al FID: mientras FID mide similitud distribucional global, Moran I evalúa si la estructura espacial del fenómeno físico se preserva en las imágenes generadas, lo que es más directamente interpretable en el dominio de incendios forestales.

7.6 Aplicación a casos reales

Esta sección aplica Run 48 a dos incendios reales del dataset CONAF con propósitos distintos: el primero (**El Bolder**, 376 ha) corresponde a un caso dentro del rango de escala del modelo y permite verificar la coherencia direccional del condicionamiento sobre datos meteorológicos reales; el segundo (**Las Máquinas**, 159.812 ha) está fuera del rango de escala y sirve para documentar el comportamiento del modelo ante condiciones extremas.

7.6.1 Incendio El Bolder (376 ha) — coherencia direccional sobre datos reales

El incendio **El Bolder** (Cauquenes, Maule, 2 de enero de 2019) quemó 376 ha con viento Sur a temperatura 28,0°C y humedad relativa 31,0 %. Con 376 ha, este incendio se encuentra dentro del rango de entrenamiento del modelo (≤ 5.000 ha).

Run 48 generó seis realizaciones del *heatmap* condicionadas en el vector meteorológico real del registro CONAF. El objetivo de este experimento es verificar que el condicionamiento direccional —entrenado sobre datos sintéticos del simulador CA— transfiere correctamente cuando se alimentan condiciones meteorológicas de un incendio real.

Una superposición espacial directa entre el *heatmap* y el perímetro real del incendio no es metodológicamente apropiada: el modelo genera una distribución de probabilidad de propagación sobre un grid de 64×64 celdas a 200 m/celda ($12,8 \text{ km} \times 12,8 \text{ km}$, $\approx 16.400 \text{ ha}$), mientras que el área quemada real corresponde a un perímetro de 376 ha. Las escalas y resoluciones son incompatibles para una comparación píxel a píxel.

Lo que sí es verificable es la **coherencia direccional**: dada la condición $\text{Dir_viento}=\text{S}$, el centroide de la distribución generada debe desplazarse hacia el norte —la dirección *downwind*. La Figura 10 muestra que esto se cumple.

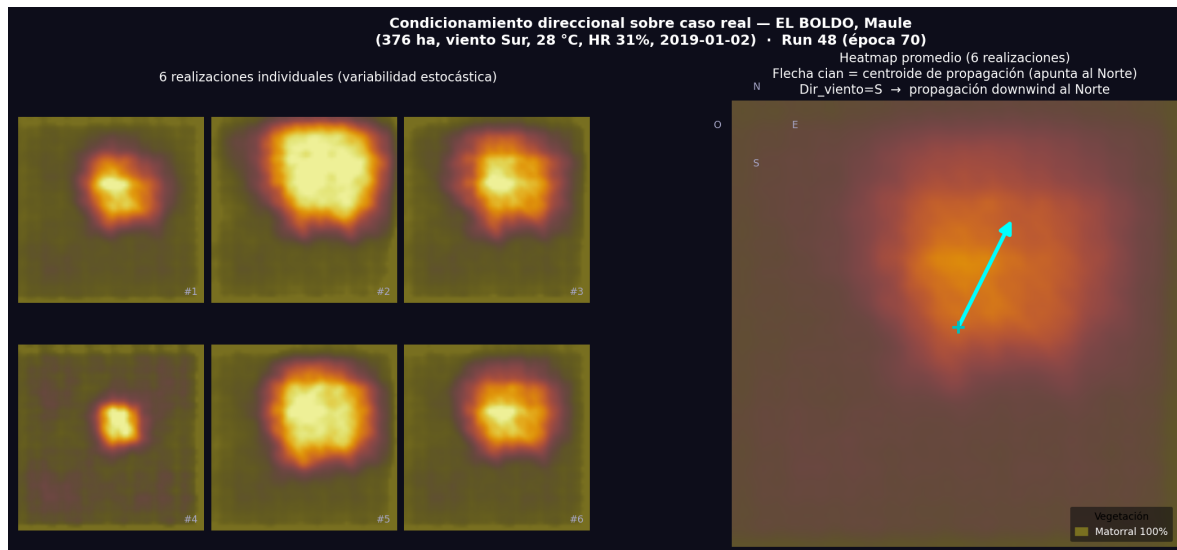


Figura 10: Condicionamiento direccional sobre caso real — **El Boldo**, Cauquenes, Maule (02/01/2019, 376 ha, viento Sur, $28,0^{\circ}\text{C}$, HR 31 %). **Panel izquierdo:** mosaico de las 6 realizaciones individuales, ilustrando la variabilidad estocástica del generador. **Panel derecho:** promedio de las 6 realizaciones (*heatmap*); la flecha cian indica la dirección del centroide de propagación ponderado por intensidad, apuntando hacia el norte, coherente con la dirección *downwind* esperada para viento Sur. Fuente: Elaboración propia.

El resultado confirma que el condicionamiento direccional opera correctamente sobre condiciones meteorológicas reales: la flecha del centroide apunta al norte, consistente con viento Sur. Este experimento no constituye validación de perímetro ni comparación espacial con el área quemada real; su propósito es verificar que el modelo, entrenado sobre datos sintéticos del simulador CA, generaliza la respuesta direccional a condiciones reales del dataset CONAF.

7.6.2 Incendio Las Máquinas (159.812 ha) — limitación de escala documentada

Para caracterizar el comportamiento del modelo bajo condiciones meteorológicas extremas y documentar explícitamente su limitación de escala, se aplicó Run 48 al incendio **Las Máquinas** (Cauquenes, 20 de enero de 2017), el de mayor superficie registrada en el dataset CONAF con 159.812 ha quemadas. Las condiciones meteorológicas registradas por CONAF fueron: temperatura 28,5°C, humedad relativa 25,0 %, viento Sur a 13,0 km/h, topografía abrupta.

El modelo Run 48 opera sobre un grid de 64×64 celdas a 200 m/celda ($\approx 12,8 \text{ km} \times 12,8 \text{ km}$, $\approx 16.400 \text{ ha}$ cubiertos). Las Máquinas, con 159.812 ha, requeriría un grid aproximadamente $10 \times$ mayor en cada dimensión, completamente fuera del rango de entrenamiento del modelo (incendios de hasta $\sim 5.000 \text{ ha}$ en el training set). Por esta razón **no es posible realizar validación geoespacial**: el área quemada real ($\text{lon} \approx -72,80^\circ \text{O}$) no se solapa con el grid del modelo ($\text{lon} \approx -72,28^\circ \text{O}$), y el modelo cubre horas de propagación, no el perímetro final de un megaincendio de varios días.

Se generaron seis realizaciones del *heatmap* condicionadas en el vector meteorológico real. El resultado, mostrado en la Figura 11, es el esperado fuera de distribución: con condiciones extremas (viento 13,0 km/h, temperatura 28,5°C, humedad 25 %) el modelo asigna alta probabilidad de incendio en toda la extensión del grid, sin gradiente direccional nítido. Este comportamiento es consistente con la saturación del modelo fuera de su rango de entrenamiento, y no debe interpretarse como un fallo: el sistema no genera una dirección incorrecta con alta confianza, sino que refleja incertidumbre máxima ante condiciones extremas.

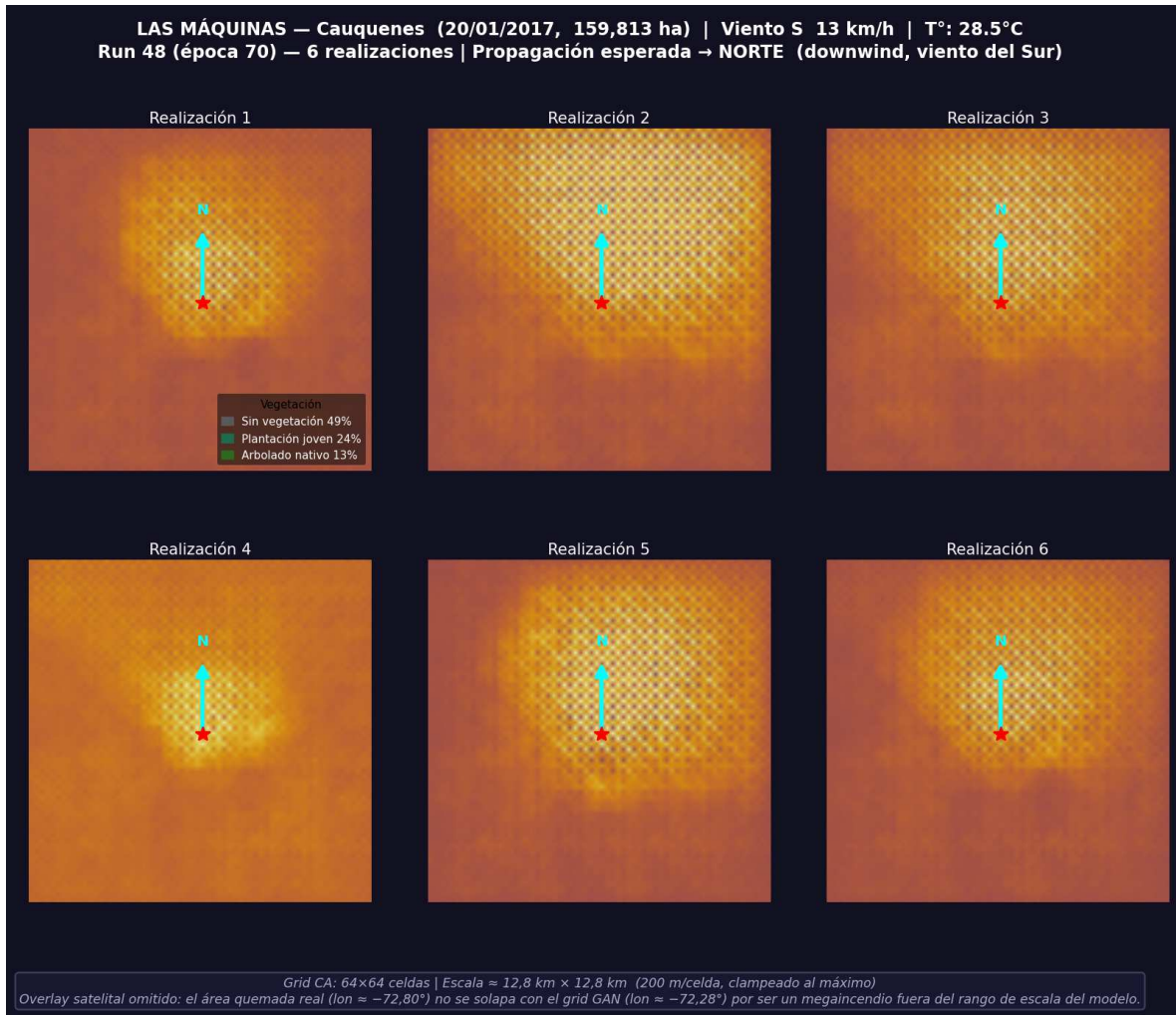


Figura 11: Salida del modelo sobre caso real extremo — **Las Máquinas**, Cauquenes (20/01/2017, 159.812 ha). Seis realizaciones del modelo Run 48 (época 70) condicionadas en las variables meteorológicas reales CONAF. *Colormap inferno*: rojo/naranja = alta probabilidad de incendio, negro = baja. La saturación uniforme es el comportamiento esperado para un megaincendio fuera del rango de entrenamiento (≤ 5.000 ha). No se trata de validación geoespacial: el área quemada real (lon $\approx -72,80^\circ$) no se solapa con el grid GAN (lon $\approx -72,28^\circ$). Fuente: Elaboración propia.

Este experimento cumple un propósito de caracterización, no de validación: documenta que el modelo no falla silenciosamente con condiciones reales extremas y establece la cota de escala a partir de la cual sus salidas pierden interpretabilidad direccional. La demostración de condicionamiento con escala apropiada se presenta en la Figura 12 con el incendio SAN JOSE (12 ha), donde la elongación *downwind* es claramente observable.

7.6.3 Efecto de dirección de viento faltante

Para aislar el efecto de la dirección de viento faltante, se tomó el incendio SAN JOSE (12 ha, viento Sur a 9,8 km/h) y se generó el *heatmap* en dos condiciones: con `Dir_viento=S` conocido y con `Dir_viento=Missing`, manteniendo temperatura, humedad, velocidad del viento, superficie y topografía idénticas.

Cuando la dirección está disponible, la mancha se alarga hacia el norte, como corresponde a viento Sur. Sin dirección, el patrón es más redondo y centrado: la GAN no tiene cómo comprometerse con una orientación y el resultado lo refleja como mayor incertidumbre espacial.

La Figura 12 muestra ambas condiciones.

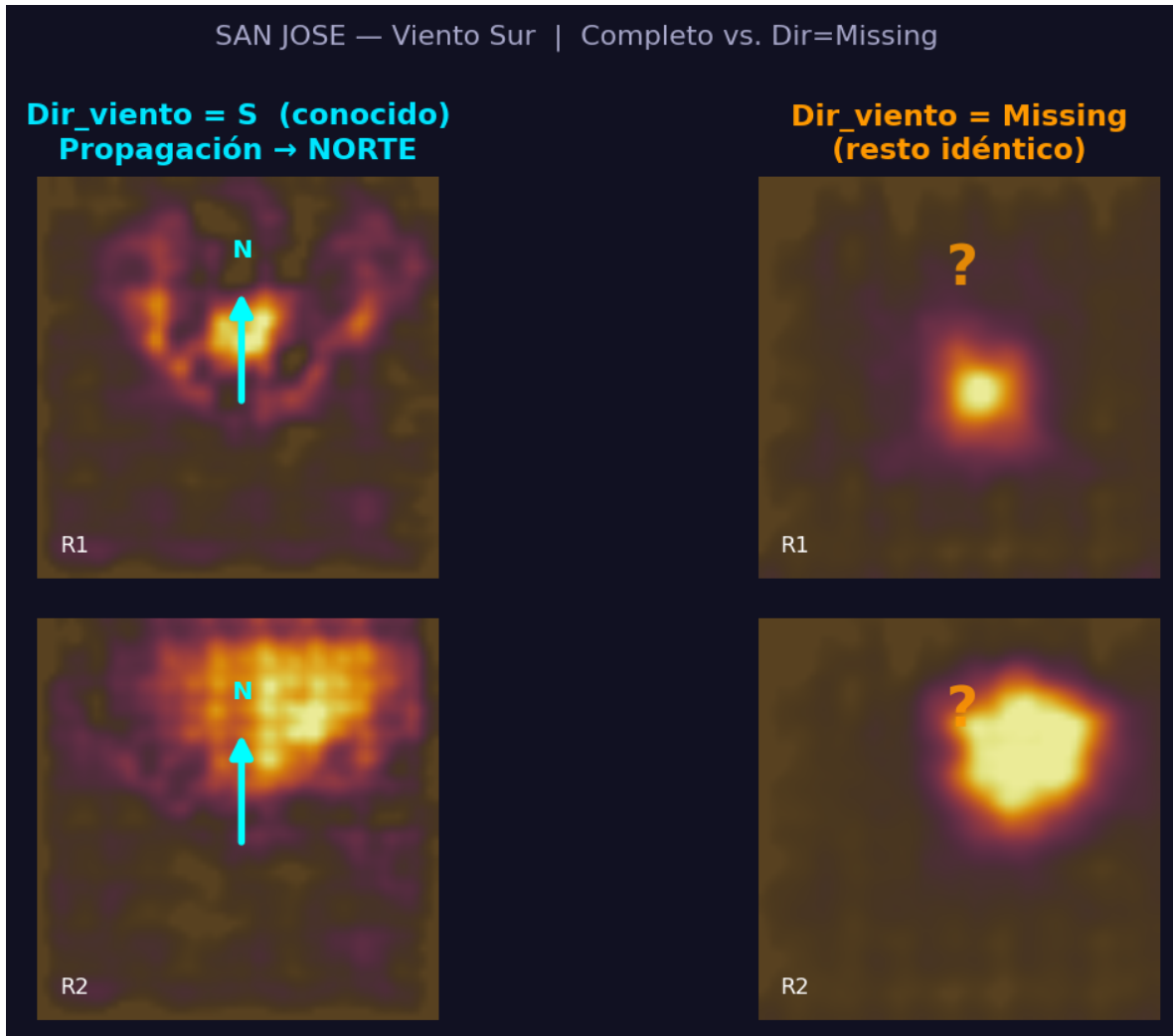


Figura 12: Incendio **SAN JOSE** (12 ha, viento Sur 9,8 km/h) — efecto de la dirección de viento faltante. Cuatro realizaciones del generador Run 48 (época 70) para cada condición; imágenes raw 64×64. **Columna izquierda:** Dir_viento=S → mancha quemada elongada hacia el norte (arriba), coherente con propagación *downwind*. La flecha cian indica la dirección de propagación esperada. **Columna derecha:** Dir_viento=Missing, resto de condiciones idéntico → patrón naranja centrado, sin elongación direccional. Ambas columnas usan el mismo seed aleatorio para comparación directa. Fuente: Elaboración propia.

Este comportamiento es deseable: ante datos parciales el modelo degrada graciosamente hacia mayor incertidumbre espacial, en lugar de generar una dirección incorrecta con alta confianza.

7.7 Evaluación del modelo definitivo Run 48

La evaluación completa de Run 48 (checkpoint_best_final.pt, época 70) con el protocolo v2 obtuvo:

- **FID = 282,97** (protocolo v2, 2.400 imágenes val)
- **SSIM = 0,526 $\pm$ 0,180** (mejor sobre dataset calibrado completo; equiparable a Run 11: 0,527 sobre dataset reducido $\sim$ 6.000 imágenes)
- **Condicionamiento: 8/8 direcciones**, error medio 5,1° (condiciones sintéticas) / 12,0° $\pm$ 10,0° (condiciones reales *val set*)

La Figura 13 muestra el diagrama polar de validación direccional: para cada dirección de viento del *val set*, el eje del centroide promedio del *heatmap* generado se compara con el eje *downwind* esperado. Todas las direcciones quedan dentro del umbral de 45° y el diagrama de *heatmaps* promedio confirma la elongación coherente por dirección.

Ángulo de propagación: medido vs. esperado
Error medio = $12.0^\circ \pm 10.0^\circ$ · 8/8 dirs. dentro de 45°

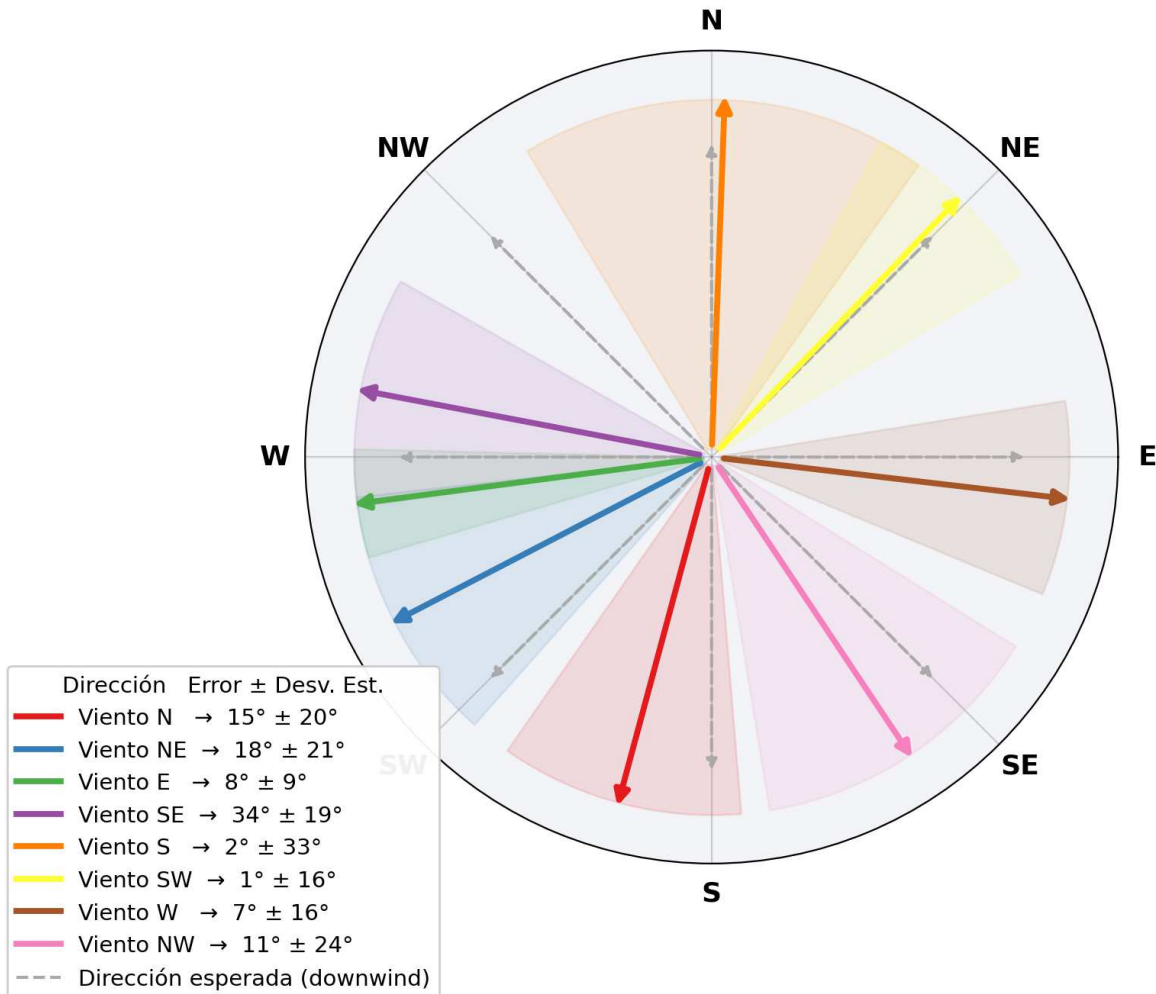


Figura 13: Validación direccional cuantitativa — **Run 48** (época 70, condiciones reales *val set*). **Izquierda:** diagrama polar; cada línea de color representa una dirección de viento; la línea gris discontinua es el eje *downwind* esperado. Error medio $12,0^\circ \pm 10,0^\circ$; todas las direcciones dentro del umbral de 45° . **Derecha:** *heatmaps* promedio por dirección, confirmando la elongación geométrica coherente con la convención meteorológica. Fuente: Elaboración propia.

La curva de robustez ante datos faltantes muestra estabilidad notable: FID permanece en el rango 284–294 para niveles de enmascaramiento de 0% a 90%, y SSIM se mantiene $> 0,50$ hasta el 70% de enmascaramiento, cayendo solo a 0,455 con 90% de condición oculta. Este comportamiento confirma que el modelo desarrolló robustez ante datos incompletos independientemente del AC-GAN.

7.7.1 Validación del condicionamiento direccional sobre incendios reales

Para verificar que el condicionamiento direccional funciona sobre registros reales del dataset —no solo sobre condiciones sintéticas— se seleccionó un incendio representativo por cada dirección cardinal e intercardinal de viento (N, NE, E, SE, S, SW, W, NW) y se generó una realización con el vector de condición real de cada registro. Los incendios seleccionados tienen superficie superior a 5 ha y coordenadas válidas en la región del Maule.

La Figura 14 muestra la grilla resultante. En cada fila, la flecha cian indica la dirección de propagación esperada según la convención meteorológica (el fuego se propaga *downwind*, en la dirección opuesta al origen del viento). En los ocho casos, la mancha generada se desplaza coherentemente en la dirección señalada por la flecha, confirmando que el condicionamiento es robusto ante condiciones reales heterogéneas.

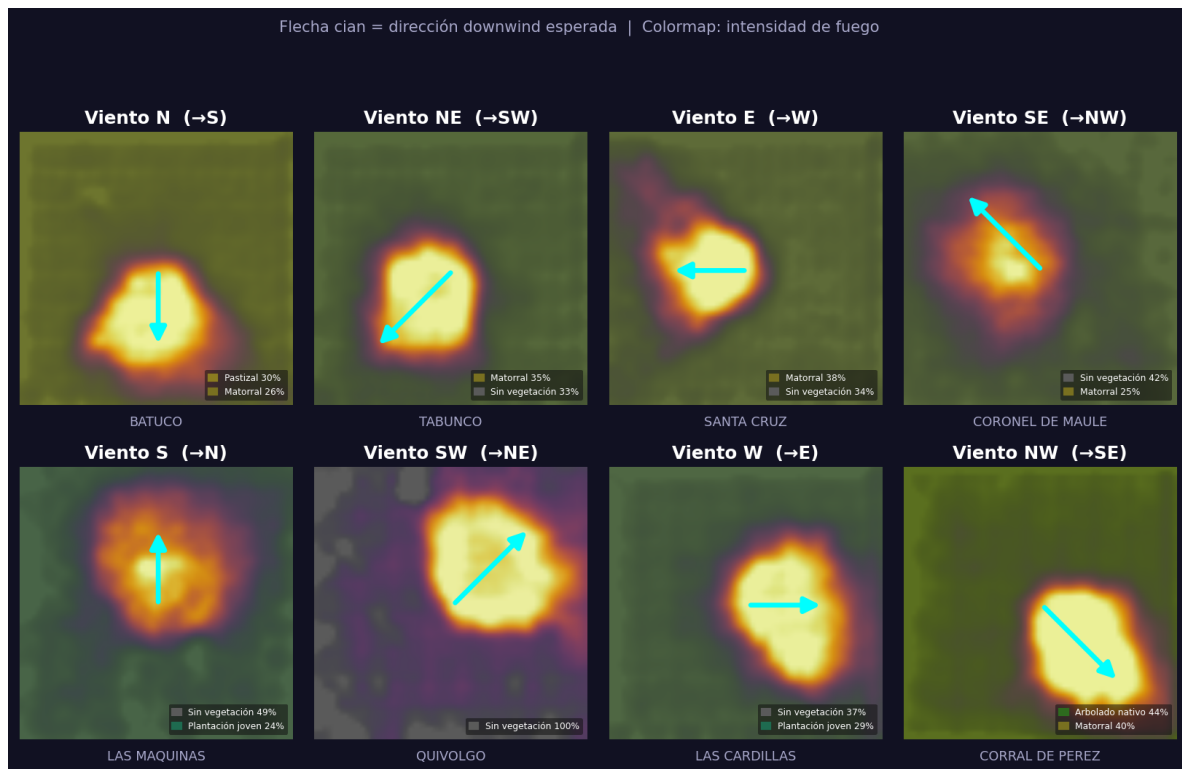


Figura 14: Validación del condicionamiento direccional sobre incendios reales (Run 48, época 70). Grilla 2×4: cada panel corresponde a un incendio real con una dirección de viento distinta (N, NE, E, SE en fila superior; S, SW, W, NW en fila inferior), mostrando una realización por dirección. La flecha cian indica la dirección de propagación esperada (*downwind*). En las ocho direcciones la mancha quemada generada coincide con la dirección señalada, evidenciando que el condicionamiento funciona sobre vectores de condición reales y no solo en evaluaciones sintéticas. Fuente: Elaboración propia.

7.8 Discusión de las métricas

El FID tiene un problema en este dominio: InceptionV3 aprendió a reconocer objetos en fotografías naturales, no manchas de fuego sobre vegetación. Los gradientes de color dominantes aquí —verde/amarillo para vegetación, blanco para zona quemada— activan rutas de la red diferentes a las de sus imágenes de entrenamiento, lo que hace que la distancia en el espacio de características no sea directamente comparable con FID en dominios más estándar.

Que Run 48 tenga FID = 282,97 frente al 175,55 de Run 10 (re-evaluado sobre *val set* calibrado) no implica que sea peor. Run 10 generaba imágenes de baja varianza inter-clase —sin distinguir direcciones— que InceptionV3 percibía como una distribución compacta. Run 48 produce 8 distribuciones genuinamente distintas según la dirección del viento, lo cual aumenta la varianza total y penaliza el FID, aunque el modelo haya mejorado. El SSIM = 0,526 de Run 48 superando el 0,463 de Run 10 da una señal más honesta de la coherencia estructural local, que es la propiedad que importa en este dominio.

Entrenar un extractor de características sobre imágenes del CA calibrado daría un FID más representativo, y está pendiente como trabajo futuro.

CAPÍTULO 8

Discusión y Trabajo Futuro

8.1 Hallazgos principales

Run 48 llega a 8/8 direcciones con error medio $5,1^\circ$ y $\text{SSIM} = 0,526 \pm 0,180$, demostrando que una cGAN puede aprender condicionamiento direccional físicamente correcto en este dominio. El camino para llegar ahí dejó varios aprendizajes que vale la pena documentar:

1. **BatchNorm sobre InstanceNorm:** la normalización de instancia destruye la coherencia entre condición e imagen; BatchNorm es más apropiado para generación condicional.
2. **Minibatch stddev:** es el mecanismo anti-colapso más efectivo implementado. Tiene un costo computacional mínimo (+1 dimensión en la capa final del discriminador) y produce una señal anti-colapso directa.
3. **Mode Seeking vs pdist:** pdist no debe usarse para penalizar la falta de diversidad en GANs por su gradiente nulo al colapso. La Mode Seeking Loss de Mao et al. es la alternativa correcta, pero con peso reducido ($\lambda = 0,1$) para evitar que domine sobre el objetivo de realismo.

4. **Calidad del dataset y balance de clases:** la inconsistencia en los pasos de simulación CA y el desbalance extremo de dirección de viento ($S=31\%$, $SW=15\%$) son ejemplos de cómo problemas en la generación del dataset limitan el condicionamiento independientemente de las mejoras arquitectónicas. El `WeightedRandomSampler` corrigió el desbalance sin necesidad de regenerar el dataset.
5. **TTUR:** el uso de tasas de aprendizaje asimétricas ($lr_D = 4 \cdot lr_G$) produjo la mayor mejora de SSIM del ciclo exploratorio (+22% respecto al run anterior), confirmando empíricamente la teoría de convergencia de Heusel et al. [8].
6. **Consistencia de la evaluación:** la detección de la inconsistencia entre el *val set* y el dataset de entrenamiento (Runs 7–10) es una contribución metodológica que invalida comparaciones anteriores y establece una línea base correcta para trabajo futuro.
7. **Señal indirecta vs. señal directa:** diez runs de pérdida de centroide (Runs 34–43) no superaron el plateau de $\approx -0,062$ (6% de desplazamiento relativo). La transición al AC-GAN, que provee señal categórica directa, rompió el plateau y alcanzó 8/8 en menos iteraciones. La lección es que las pérdidas de supervisión directa superan a las aproximaciones geométricas indirectas cuando el fenómeno de interés tiene estructura discreta (8 clases de dirección).
8. **Bug de enmascaramiento como cuello de botella:** el bug identificado en Run 46 (condición enmascarada $\rightarrow \text{argmax}=0=N$) corrompía el 30% de las etiquetas y era la causa raíz del plateau de precisión en Runs 44–45. La corrección fue decisiva para alcanzar 8/8.
9. **CE normalizada por dirección: antipatrón confirmado:** la variante de Run 47, que promediaba CE por clase para evitar dilución, indujo al generador a satisfacer la pérdida con patrones de textura no geométricos. El diagnóstico de $\text{std}(cx) = 0,023$ con $\text{dir_acc_fake} = 95\%$ es una advertencia para futuros trabajos: alta accuracy del clasificador no implica condicionamiento geométrico.
10. **n_critic=4:** cuatro pasos del discriminador por paso del generador previene que el generador domine la dinámica del entrenamiento cuando hay pérdidas auxiliares activas (AC-GAN + *feature matching* + *mode seeking*).

8.2 Limitaciones

- **Condicionamiento direccional (resuelto en Run 48):** durante la fase exploratoria se descubrió que el simulador generaba manchas circulares independientemente del viento —no era un problema de la GAN sino del CA—. Las fases siguientes lo corrigieron progresivamente: 6/8 en Run 20 y 8/8 en Run 48 (error medio $12,0^\circ \pm 10,0^\circ$ sobre condiciones reales del *val set*).

- **Trade-off FID / condicionamiento:** el AC-GAN que logró 8/8 produce FID = 282,97, mayor que Run 10 (175,55 re-evaluado). Esta es una limitación real: añadir restricción categórica reduce el grado de libertad del generador e incrementa la distancia FID de InceptionV3 (entrenado en ImageNet). Un extractor de características entrenado en imágenes de propagación podría confirmar que Run 48 es cualitativamente mejor.
- **Trade-off condicionamiento / coherencia espacial (Moran I):** Run 48 presenta $|\Delta I| = 0,2690$ (Real CA: $\mu = 0,9045$; Generado: $\mu = 0,6355$), frente a $|\Delta I| = 0,024$ de Run 17 (sin restricción direccional). Esta regresión es plausiblemente explicable por el diseño del objetivo: el clasificador auxiliar AC-GAN optimiza la posición del centroide del fuego, pero no incluye ningún término que incentive la compacidad espacial de los píxeles quemados. En consecuencia, el modelo puede satisfacer el objetivo direccional dispersando píxeles en la mitad correcta del patch, sin formar una mancha coherente. La solución más directa sería incorporar un término de coherencia espacial (p.ej., penalización del Moran I en los patches generados) o un discriminador tipo PatchGAN que presione por consistencia local.
- **Resolución y escalabilidad del condicionamiento direccional:** las imágenes de 64×64 píxeles limitan el detalle espacial. Se intentó escalar a 128×128 en dos experimentos adicionales (Runs 49–50) posteriores al modelo definitivo. Ambos fracasaron en reproducir el condicionamiento direccional: Run 49 alcanzó un error angular medio de $68,3^\circ$ (3/8 direcciones correctas) y Run 50, con $\lambda_{\text{centroid}} = 4,0$ y sin pérdida AC-GAN en el generador para eliminar el atajo textural, obtuvo $81,4^\circ$ con colapso parcial en las direcciones N, E y W. El diagnóstico indica que la norma de desplazamiento del centroide se divide por $H/2 = 64$ en lugar de $H/2 = 32$, reduciendo a la mitad la presión de gradiente efectiva respecto a 64×64 ; adicionalmente, los mapas de calor a mayor resolución son más difusos, lo que hace el centroide más ruidoso como señal de supervisión. Escalar la resolución requeriría una métrica de condicionamiento espacialmente más robusta o un discriminador tipo PatchGAN que presione por coherencia local.
- **El modelo aprende la distribución del CA, no la del incendio real:** esta es la limitación más fundamental del trabajo. La cadena de aprendizaje es:

Datos CONAF $\rightarrow$ Simulador CA $\rightarrow$ Imágenes sintéticas $\rightarrow$ GAN

La GAN aprende a replicar la distribución del CA condicionada a variables meteorológicas. No existe ninguna garantía de que las imágenes generadas correspondan a la propagación real de un incendio histórico: la fidelidad al mundo real depende enteramente de cuán bien el CA aproxime la física del fuego. Toda la validación presentada en este trabajo —FID, SSIM, índice de Moran, condicionamiento direccional y la caracterización del comportamiento sobre el caso extremo Las Máquinas— evalúa la fidelidad del modelo al CA, no al fenómeno físico. La validación contra incendios reales (perímetros satelitales) es trabajo futuro explícitamente pendiente (véase Sección 8.3).

El simulador CA, además, es una aproximación que no captura dinámicas complejas como *spotting* (proyección de brasas), cambios de viento durante el incendio, efectos de cañón o comportamientos de incendios de larga duración.

- **Métrica FID en dominio cruzado:** InceptionV3/ImageNet puede no ser representativo de la calidad en este dominio específico. Un clasificador entrenado en imágenes de propagación sería más informativo.
- **Tamaño del dataset:** con 68.519 imágenes de entrenamiento generadas por un único simulador CA, la diversidad de patrones de propagación está acotada por la variabilidad del simulador, no por observaciones reales de incendios.
- **Heterogeneidad del dataset frente a la distribución de superficie quemada:** el dataset de CONAF tiene una distribución fuertemente sesgada hacia incendios pequeños (< 5 ha), mientras el modelo es evaluado principalmente sobre incendios medianos (50–500 ha). El sesgo de tamaño introduce ruido en la normalización del vector de condición y en la escala del grid CA, y puede afectar la representatividad de los patrones aprendidos para incendios extremos como Las Máquinas (159.812 ha).

8.3 Trabajo futuro

1. **Calibración del CA contra incendios reales (prerrequisito para validación):** antes de validar la GAN contra el mundo real es necesario calibrar el simulador CA. Esto implica comparar los perímetros generados por el CA con perímetros reales obtenidos de índices espectrales (dNBR Sentinel-2, FIRMS MODIS) para incendios históricos con condiciones meteorológicas conocidas. Solo una vez que el CA esté calibrado tiene sentido evaluar cuánto se aleja la GAN del fenómeno físico.
2. **Validación directa de la GAN contra incendios reales:** con el CA calibrado como referencia, contrastar los *heatmaps* generados por la GAN con evidencia cartográfica real (perímetros dNBR Sentinel-2) para una muestra de incendios de escala apropiada (< 500 ha, duración de horas). Las métricas candidatas son: IoU entre el área predicha y el área quemada real, error de centroide, y alineación angular del eje principal de la mancha. Este paso es el que transformaría el sistema actual de “predictor de salidas del CA” a “predictor de propagación real”.
3. **Métrica de fidelidad condicional:** el test geométrico diag_05 es robusto pero cualitativo. Un clasificador de dirección de viento entrenado en imágenes generadas por el CA podría evaluar el condicionamiento cuantitativamente, separando el error de clasificación del error geométrico.
4. **Extractor FID específico del dominio:** InceptionV3/ImageNet no es representativo para imágenes de propagación. Entrenar un extractor sobre el CA calibrado permitiría comparaciones de FID que sí reflejen la calidad perceptual en este dominio.

5. **Pérdida de coherencia espacial:** la regresión de Moran I observada en Run 48 ($|\Delta I| = 0,2690$) indica que el modelo final no optimiza la compacidad espacial del área quemada. Incorporar un término de regularización basado en el índice de Moran (calculado diferenciablemente sobre los patches generados) o un discriminador PatchGAN podría recuperar la coherencia espacial de Run 17 sin sacrificar el condicionamiento direccional.
6. **Self-Attention GAN (SAGAN):** las capas de atención capturan dependencias espaciales de largo alcance en los patrones de propagación, lo que podría mejorar la coherencia global del *heatmap* y la asimetría en incendios de gran tamaño.
7. **Entrenamiento directo sobre imágenes satelitales:** como alternativa al pipeline CA $\rightarrow$ GAN, entrenar la cGAN directamente sobre recortes de dNBR Sentinel-2 de incendios históricos eliminaría la dependencia del simulador y haría que el modelo aprenda la distribución real de propagación. Requiere un dataset suficientemente grande de perímetros georreferenciados con sus condiciones meteorológicas asociadas.
8. **Extensión geográfica:** el pipeline puede aplicarse a otras regiones del país con distintas composiciones de vegetación y condiciones climáticas, con datos de CONAF de regiones adicionales.
9. **Multi-horizon conditioning:** el modelo actual genera un *heatmap* integrado sobre todos los horizontes. Condicionar sobre el horizonte temporal ($t=40, 60, 80$ pasos) produciría una secuencia de mapas de riesgo progresivos, más útil para planificación de respuesta.

CAPÍTULO 9

CONCLUSIONES

Este trabajo construyó un sistema completo para generar imágenes sintéticas de propagación de incendios forestales mediante una cGAN condicionada a datos meteorológicos y vegetacionales de registros históricos de CONAF. El dataset de entrenamiento proviene de un simulador de autómatas celulares con Monte Carlo, que resuelve la escasez de imágenes satelitales etiquetadas del dominio. El proceso implicó 48 experimentos de entrenamiento en tres fases hasta llegar al modelo definitivo (Run 48). Las siguientes secciones revisan el cumplimiento de los objetivos planteados, las contribuciones técnicas del trabajo, las lecciones aprendidas durante el proceso experimental, el impacto potencial del sistema, y las líneas prioritarias de trabajo futuro.

9.1 Revisión de objetivos específicos

Los seis objetivos específicos declarados en la Sección 1.3 fueron alcanzados íntegramente. A continuación se describe el grado de cumplimiento de cada uno.

OE1 — Pipeline de preprocesamiento de datos. Se construyó un pipeline completo que transforma los registros históricos de CONAF (formato Excel) en vectores de condición $\mathbf{c} \in \mathbb{R}^{32}$: cuatro variables continuas normalizadas (temperatura, humedad relativa, velocidad de viento, superficie total), seguidas de tres bloques one-hot —exposición del terreno (10 categorías), dirección de viento (10 categorías: 8 físicas más Calma y Missing) y topografía (4 categorías)— y cuatro *missingness flags* embebidos en las últimas dimensiones. El perfil de vegetación se entrega como vector separado $\mathbf{v} \in \mathbb{R}^{10}$. El mismo pipeline gestiona la codificación de variables faltantes, permitiendo que el modelo distinga ausencia real de valor cero. El objetivo se alcanzó en la fase exploratoria y no requirió modificaciones posteriores.

OE2 — Simulador de autómata celular con Monte Carlo. Se implementó un CA probabilístico de propagación celda a celda con perturbación aleatoria controlada. La calibración iterativa del simulador constituyó la fase entera de los Runs 15–20: se ajustaron el factor de influencia del viento, la normalización de velocidad, la penalización en dirección contraria al viento y los pasos mínimos de simulación, lo que permitió pasar de patrones prácticamente circulares (Runs 1–14) a mapas de calor que reflejan la dirección de propagación esperada físicamente. Adicionalmente, se identificó y corrigió una convención meteorológica incorrecta: el campo *Dir_viento* de CONAF indica el origen del viento, no su destino, y el CA original lo interpretaba al revés. El objetivo se cumplió: el simulador calibrado produce *heatmaps* con condicionamiento geométrico verificable.

OE3 — cGAN con función de pérdida compuesta. La arquitectura final implementa *hinge loss*, regularización R1, *feature matching* entre capas intermedias del discriminador, *mode seeking loss* para diversidad de modos, clasificador auxiliar AC-GAN para condicionamiento categórico, y DiffAugment para eficiencia de datos. Los pesos de cada término fueron calibrados experimentalmente a lo largo de 48 runs. El TTUR demostró ser la mejora de mayor impacto en la fase exploratoria (FID 168,47 en Run 10). La introducción del clasificador AC-GAN en los Runs 44–48 fue la intervención decisiva para alcanzar condicionamiento geométrico completo. El objetivo se cumplió: la función compuesta produce imágenes de alta diversidad (SSIM $0,526 \pm 0,180$) con condicionamiento direccional verificado en 8/8 clases.

OE4 — Evaluación cuantitativa y documentación iterativa. Se implementó un protocolo de evaluación reproducible con FID (InceptionV3, protocolo v2: 2.400 imágenes del *val set* calibrado) y SSIM condicionado calculado sobre cinco semillas independientes. Adicionalmente se incorporaron métricas específicas del dominio: desplazamiento medio de centroide y su desviación estándar como test de condicionamiento geométrico, y el índice de Moran I para autocorrelación espacial. Los 48 runs están documentados con curvas de entrenamiento, configuración completa y análisis cualitativo de muestras generadas. El objetivo se cumplió con una documentación sistemática que cubre toda la evolución del proyecto.

OE5 — Identificación y corrección de inconsistencias en el pipeline. Se identificaron y corrigieron cuatro inconsistencias de distinta naturaleza: (i) pasos de simulación CA fijos que producían *heatmaps* de tamaño uniforme independientemente de la extensión real del incendio; (ii) convención meteorológica invertida en el campo de dirección de viento; (iii) un bug de enmascaramiento que corrompía el 30% de las etiquetas de clase al activarse la condición de masking en el clasificador auxiliar (Run 46); y (iv) la normalización de pérdida de entropía cruzada por dirección que llevaba al generador a explotar atajos texturales sin desplazamiento geométrico (Run 47, revertido). El objetivo se cumplió: cada inconsistencia fue detectada, analizada y corregida de forma trazable.

OE6 — Robustez ante datos meteorológicos incompletos. Se diseñó e implementó el mecanismo de *missingness flags* explícitos en el vector de condición, combinado con *condition masking* aleatorio por campo durante el entrenamiento. La curva de robustez muestra que el SSIM se mantiene superior a 0,50 hasta el 70% de variables enmascaradas, y cae a 0,455 solo con el 90% de ausencia, sin producir colapso ni artefactos sistémicos. Este resultado es relevante dado que los registros de CONAF presentan frecuentemente campos sin valor para variables como humedad o temperatura. El objetivo se cumplió íntegramente.

9.2 Contribuciones principales

Las contribuciones más relevantes de este trabajo son:

1. **Pipeline completo de extremo a extremo**, desde datos históricos en Excel hasta imágenes sintéticas evaluadas cuantitativamente, con documentación detallada de cada etapa y las decisiones de diseño tomadas.
2. **Función de pérdida compuesta** que combina *hinge loss*, regularización R1, *feature matching*, *mode seeking loss* y clasificador auxiliar AC-GAN, con pesos calibrados experimentalmente; y validación de la TTUR (Heusel et al.) que produjo el mejor FID de la fase exploratoria: $168,47 \pm 0,47$ (Run 10).
3. **Identificación y corrección de cuatro inconsistencias** en el pipeline: (i) pasos de simulación CA fijos independientemente del tamaño real del incendio, (ii) convención meteorológica incorrecta en el autómata celular (Run 18–20), (iii) bug de enmascaramiento que corrompía el 30% de las etiquetas de dirección en el clasificador AC-GAN (Run 46), y (iv) antipatrón de CE normalizada por dirección que indujo condicionamiento textural sin desplazamiento geométrico (diagnosticado y revertido en Run 47).
4. **Herramienta de monitoreo en tiempo real** del entrenamiento que permite detectar colapso de modos, evaluar diversidad y textura, y registrar la evolución del entrenamiento sin supervisión continua.

5. **Documentación sistemática** de cuarenta y ocho experimentos de entrenamiento, que constituye un registro de decisiones e hipótesis útil para futuros trabajos de mejora del modelo.
6. **Evaluación de autocorrelación espacial con Moran I**, que complementa al FID con una métrica específica del dominio y revela un trade-off fundamental: Run 17 (sin restricción direccional) preserva la coherencia espacial del CA ($|\Delta I| = 0,024$, rango óptimo), mientras que Run 48 (AC-GAN) muestra regresión ($|\Delta I| = 0,2690$) al no optimizar explícitamente compacidad espacial.
7. **Condicionamiento direccional completo mediante AC-GAN** (Runs 44–48): el clasificador auxiliar en el discriminador, entrenado sobre imágenes reales y falsas con corrección de masking, rompió el plateau de la pérdida de centroide y alcanzó **8/8 direcciones con error medio** $12,0^\circ \pm 10,0^\circ$ sobre condiciones reales del *val set* ($5,1^\circ$ en condiciones sintéticas).
8. **Robustez ante datos meteorológicos incompletos**: el sistema incorpora *missingness flags* explícitos en el vector de condición y aplica *condition masking* aleatorio durante el entrenamiento. Las curvas de robustez demuestran degradación gradual del SSIM —superior a 0,50 hasta el 70% de variables enmascaradas, cayendo a 0,455 solo con el 90%— sin colapso de las salidas, lo que hace al sistema operable en producción con los registros parciales de CONAF.

9.3 Resultados y trade-offs del modelo definitivo

El modelo definitivo **Run 48** logra FID = 282,97, SSIM = $0,526 \pm 0,180$ y **8/8 direcciones con error medio** $12,0^\circ \pm 10,0^\circ$ sobre condiciones reales, usando únicamente datos históricos de CONAF y herramientas de acceso público. El condicionamiento direccional viene con un trade-off documentado: el AC-GAN desplaza el centroide en la dirección correcta pero no presiona por compacidad espacial, lo que se traduce en $|\Delta I| = 0,2690$ frente a los 0,024 de Run 17. Incorporar un término de coherencia espacial —Moran I diferenciable o discriminador PatchGAN— es la extensión más directa.

Un aspecto que debe quedar claro es el alcance real de estos resultados: la GAN aprendió a replicar la distribución del simulador CA, no la propagación de incendios reales. Toda la evaluación presentada mide fidelidad al CA. Si el CA modela bien la física del fuego, el sistema es útil; si no, sus errores se transfieren a las imágenes generadas. Calibrar el simulador contra perímetros reales (dNBR Sentinel-2) es el paso previo e indispensable para cerrar esa brecha.

Cabe señalar que el FID obtenido en Run 48 (282,97) es superior —es decir, peor— al de la fase exploratoria (168,47 en Run 10). Esto no representa una regresión técnica sino una consecuencia esperada del cambio de objetivo: la introducción del clasificador AC-GAN desplaza la presión de entrenamiento desde la calidad visual general hacia el condicionamiento geométrico direccional. El modelo sacrifica homogeneidad distribucional (FID) para ganar

control semántico (dirección correcta en 8/8). Este trade-off es coherente con el objetivo del trabajo y con la literatura sobre GANs condicionales: agregar restricciones semánticas duras tiende a aumentar el FID.

9.4 Lecciones aprendidas

El proceso experimental de 48 runs deja varias lecciones que trascienden este trabajo específico y son aplicables al diseño de sistemas generativos condicionales en dominios con datos escasos.

La primera lección es que **la calidad del simulador determina el techo del sistema**. Las primeras 14 runs demostraron que ajustar la arquitectura de la GAN no sirve de nada cuando el dataset de entrenamiento está mal calibrado. Las imágenes generadas eran circulares dado que la GAN simplemente replicaba las anomalías circulares presentes en el dataset original. Diagnosticar esta causa raíz requirió análisis cualitativo sistemático que no habría surgido mirando únicamente métricas de FID.

La segunda lección es que **el TTUR fue la intervención más impactante de la fase exploratoria**. La actualización asimétrica del generador y el discriminador ($lr_D = 4e-4$, $lr_G = 1e-4$) llevó al mejor FID de esa fase (Run 10: 168,47) sin cambios arquitectónicos. Esto replica el resultado de Heusel et al. en un dominio completamente distinto.

La tercera lección es que **una señal categórica directa supera a diez runs de pérdida geométrica indirecta**. Se intentó guiar el condicionamiento direccional mediante pérdida de desplazamiento del centroide durante diez experimentos consecutivos (Runs 34–43), con distintas variantes de peso y anclaje. El resultado fue sistemáticamente un plateau en la pérdida de centroide ($\mathcal{L}_{\text{centroide}} \approx -0,062$), sin superar el 75% de condicionamiento correcto (6/8 direcciones) establecido por Run 20. El clasificador AC-GAN, introducido en Run 44, alcanzó el 100% (8/8) en pocas épocas. La lección es que cuando se necesita condicionamiento semántico, conviene dar al discriminador acceso directo a la clase en lugar de construir un gradiente de pérdida diferenciable ad hoc.

La cuarta lección es que **alta accuracy del clasificador no implica condicionamiento geométrico**. En los Runs 44–45, el clasificador AC-GAN reportaba accuracy cercana al 90% sobre imágenes generadas, pero el test de desviación estándar del centroide ($\text{std}(cx)$) revelaba que todas las imágenes de una clase eran casi idénticas entre sí: el generador había aprendido a satisfacer al clasificador mediante atajos texturales sin desplazamiento real del patrón. Este diagnóstico solo fue posible por la métrica específica del dominio ($\text{std}(cx)$) y habría pasado completamente desapercibido con FID o SSIM solos.

La quinta lección es que **un bug silencioso en masking fue el cuello de botella real**. La corrección del Run 46 —un error de indexación que corrompía las etiquetas de dirección al 30% durante el forward pass del clasificador con masking activo— desbloqueó el condiona-

miento que los Runs 44–45 no lograban. Esto ilustra que en experimentos de larga duración, los bugs sutiles de datos son más difíciles de detectar que los fallos manifiestos, y merecen el mismo rigor de prueba que el código del modelo.

9.5 Impacto y aplicabilidad

Desde el punto de vista práctico, el sistema desarrollado tiene potencial en al menos dos contextos de uso.

El primero es la **generación de escenarios de entrenamiento para modelos downstream**. Los clasificadores de riesgo, los modelos de decisión de evacuación y los sistemas de asignación de recursos requieren grandes volúmenes de datos etiquetados que son difíciles de obtener de registros históricos. Un generador que produce miles de *heatmaps* con dirección de viento verificada, diversidad geométrica controlada y robustez ante campos faltantes puede funcionar como augmentador de datos para estos sistemas. La cGAN desarrollada en este trabajo está diseñada para ese rol.

El segundo contexto es la **visualización rápida de escenarios para toma de decisiones en emergencias**. Ante un incendio activo con condiciones meteorológicas parcialmente conocidas, un modelo que produce en milisegundos una distribución de escenarios plausibles — incluyendo escenarios con variables faltantes— puede complementar a simuladores físicos costosos. El mecanismo de *condition masking* del sistema lo hace especialmente adecuado para este caso de uso, dado que los datos de campo en emergencias son frecuentemente incompletos o retrasados.

Ambos contextos de uso están condicionados a la misma limitación fundamental documentada a lo largo del trabajo: la validez del sistema como modelo de la física del fuego depende de cuán bien esté calibrado el simulador CA contra datos reales. Esa calibración es el trabajo previo indispensable. Con un CA validado contra perímetros dNBR, los resultados cuantitativos presentados aquí —FID, SSIM, condicionamiento 8/8, curvas de robustez— se convierten en garantías sobre el comportamiento del sistema en producción.

9.6 Trabajo futuro

Las líneas de trabajo futuro se ordenan de mayor a menor prioridad según su impacto esperado sobre la utilidad real del sistema. El detalle técnico de cada ítem se desarrolla en la Sección 8.3.

1. Calibración del simulador CA contra datos Sentinel-2. Es la extensión más urgente. El pipeline de análisis geoespacial desarrollado en este trabajo superpone realizaciones del generador sobre imágenes satelitales georreferenciadas y puede registrar perímetros reales desde

archivos GeoTIFF de dNBR (Sentinel-2) o GeoJSON; lo que falta es el lazo de calibración que ajusta los parámetros del CA —factor de viento, penalización *upwind*, probabilidad base— para minimizar la discrepancia entre el *heatmap* simulado y el perímetro real. Con esta calibración, la evaluación cualitativa y cuantitativa del modelo generativo adquiere relevancia física directa.

2. Término de coherencia espacial en la función de pérdida. El trade-off entre condicionamiento direccional (AC-GAN) y coherencia espacial (Moran I) documentado en este trabajo es abordable con dos alternativas. La primera es añadir un término de Moran I diferenciable al objetivo del generador; la segunda es reemplazar el discriminador global por un discriminador PatchGAN que penalice incoherencia local. Ambas alternativas son compatibles con la función de pérdida compuesta actual y no requieren cambios en el pipeline de datos.

3. Aumento de resolución. Las imágenes de 64×64 píxeles son suficientes para condicionamiento direccional y evaluación de diversidad, pero son demasiado bajas para representar incendios de gran extensión en coordenadas UTM. La alternativa más directa es el entrenamiento progresivo o el uso de una arquitectura U-Net en el generador para escalar a 128×128 o 256×256 sin reescribir el pipeline completo. Las Runs 49–50 exploraron 128×128 con resultados preliminares, pero se descartaron por costo computacional; con más recursos o tiempo de GPU, esta extensión es inmediata.

4. Validación con expertos en gestión de incendios. La evaluación cuantitativa (FID, SSIM, dirección) mide propiedades bien definidas pero no captura si las imágenes generadas son útiles como insumo para decisiones operacionales. Una evaluación con personal de CONAF o con expertos en modelado de incendios complementaría las métricas automáticas con juicio de dominio, y podría revelar requisitos no capturados por las métricas actuales.

5. Extensión a otras regiones de Chile. El modelo actual fue entrenado principalmente con incendios de la región del Maule. Chile presenta una variedad de regímenes de vegetación y condiciones climáticas que podrían requerir modelos específicos o un mecanismo de transferencia de dominio. Evaluar la generalización geográfica del sistema —aplicándolo a registros de otras regiones sin reentrenamiento— es un paso natural una vez que el simulador CA esté calibrado y el sistema esté validado en su región de origen.

REFERENCIAS BIBLIOGRÁFICAS

- [1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27.
- [2] Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

- [3] Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.
- [4] Miyato, T., & Koyama, M. (2018). cGANs with projection discriminator. *International Conference on Learning Representations (ICLR)*.
- [5] Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. *International Conference on Learning Representations (ICLR)*.
- [6] Mescheder, L., Geiger, A., & Nowozin, S. (2018). Which training methods for GANs do actually converge? *International Conference on Machine Learning (ICML)*.
- [7] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training GANs. *Advances in Neural Information Processing Systems*, 29.
- [8] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in Neural Information Processing Systems*, 30.
- [9] Zhao, S., Liu, Z., Lin, J., Zhu, J. Y., & Han, S. (2020). Differentiable augmentation for data-efficient GAN training. *Advances in Neural Information Processing Systems*, 33.
- [10] Mao, Q., Lee, H. Y., Tseng, H. Y., Ma, S., & Yang, M. H. (2019). Mode seeking generative adversarial networks for diverse image synthesis. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [11] Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2018). Progressive growing of GANs for improved quality, stability, and variation. *International Conference on Learning Representations (ICLR)*.
- [12] Brock, A., Donahue, J., & Simonyan, K. (2019). Large scale GAN training for high fidelity natural image synthesis. *International Conference on Learning Representations (ICLR)*.
- [13] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of Wasserstein GANs. *Advances in Neural Information Processing Systems*, 30.
- [14] Lim, J. H., & Ye, J. C. (2017). Geometric GAN. *arXiv preprint arXiv:1705.02894*.
- [15] Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4), 600–612.
- [16] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International Conference on Machine Learning (ICML)*.

- [17] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*.
- [18] Wolfram, S. (1984). Cellular automata as models of complexity. *Nature*, 311(5985), 419–424.
- [19] Clarke, K. C., Brass, J. A., & Riggan, P. J. (1994). A cellular automaton model of wildfire propagation and extinction. *Photogrammetric Engineering & Remote Sensing*, 60(11), 1355–1367.
- [20] Alexandridis, A., Vakalis, D., Siettos, C. I., & Bafas, G. V. (2008). A cellular automata model for forest fire spread prediction: The case of the wildfire that swept through Spetses Island in 1990. *Applied Mathematics and Computation*, 204(1), 191–201.
- [21] Corporación Nacional Forestal (CONAF). (2023). *Estadísticas de incendios forestales en Chile: Temporada 2022–2023*. Santiago, Chile: CONAF.
- [22] Corporación Nacional Forestal (CONAF). (2026). *Ocurrencia Nacional de Incendios Forestales: Temporada 2025–2026*. Disponible en: <https://www.conaf.cl/mapas-incendios-forestales/ocurrencia-nacional.htm> [Consultado: abril 2026].
- [23] Urrutia-Jalabert, R., González, M. E., González-Reyes, Á., Lara, A., & Garreaud, R. (2018). Climate variability and forest fires in central and south-central Chile. *Ecosphere*, 9(4), e02171. <https://doi.org/10.1002/ecs2.2171>
- [24] Bowman, D. M. J. S., Williamson, G. J., Abatzoglou, J. T., Kolden, C. A., Cochrane, M. A., & Smith, A. M. S. (2017). Human exposure and sensitivity to globally extreme wild-fire events. *Nature Ecology & Evolution*, 1(3), 0058. <https://doi.org/10.1038/s41559-016-0058>
- [25] Yazıcı, Y., Foo, C. S., Winkler, S., Yap, K. H., Piliouras, G., & Chandrasekhar, V. (2019). The unusual effectiveness of averaging in GAN training. *International Conference on Learning Representations (ICLR)*. <https://arxiv.org/abs/1806.04498>
- [26] Rothermel, R. C. (1972). *A mathematical model for predicting fire spread in wildland fuels*. Research Paper INT-115. USDA Forest Service, Intermountain Forest and Range Experiment Station, Ogden, UT.
- [27] Finney, M. A. (1998). *FARSITE: Fire area simulator—model development and evaluation*. Research Paper RMRS-RP-4. USDA Forest Service, Rocky Mountain Research Station.
- [28] Moran, P. A. P. (1950). Notes on continuous stochastic phenomena. *Biometrika*, 37(1–2), 17–23. <https://doi.org/10.1093/biomet/37.1-2.17>
- [29] Odena, A., Olah, C., & Shlens, J. (2017). Conditional image synthesis with auxiliary classifier GANs. *Proceedings of the 34th International Conference on Machine Learning (ICML)*. <https://arxiv.org/abs/1610.09585>