

Bloque: Redes Neuronales con Imágenes

Ana Jiménez Pastor
anjipas@gmail.com

Organización módulo

1^a EVALUACIÓN

Tema 0. Introducción a la IA en imagen médica y entrenamiento de modelos y puesta en producción

Tema 1. Procesamiento de imagen con Python

Tema 2. Aplicaciones de las redes neuronales convolucionales

Tema 3. Introducción a las redes neuronales convolucionales

2^a EVALUACION

Tema 4. Transferencia de conocimiento

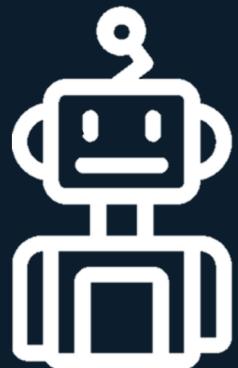
Tema 5. Aumento de datos

Tema 6. Segmentación

IA, Machine Learning, Deep Learning

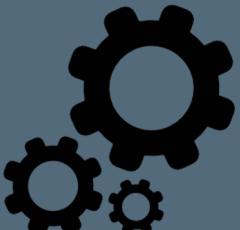
Inteligencia Artificial

Cualquier técnica que permita a los ordenadores imitar el comportamiento humano



Machine Learning

Capacidad de que un algoritmo sea capaz de aprender partir de unos datos de entrada



Deep Learning

Extrae patrones de los datos empleando redes neuronales profundas



1950

1980

2010

Visión por computador



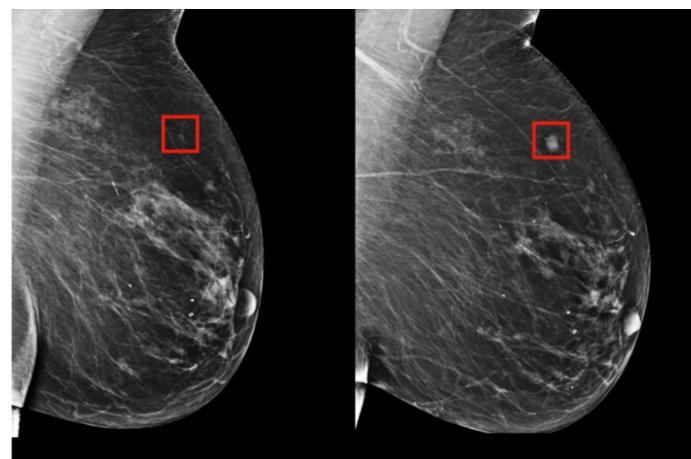
Coches autónomos



Reconocimiento de caras



Realidad aumentada

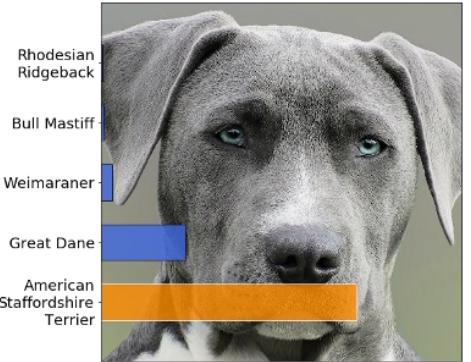


Salud

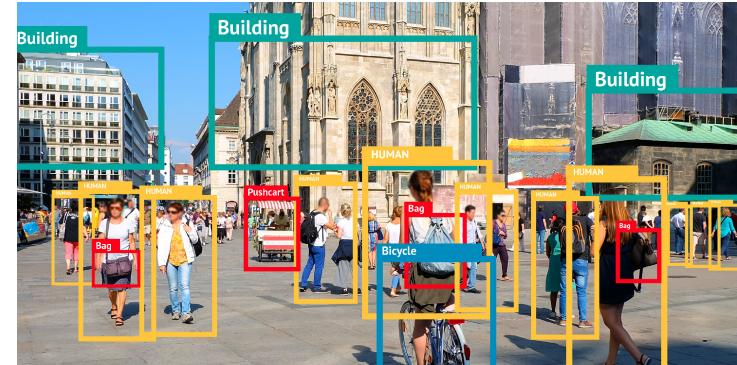
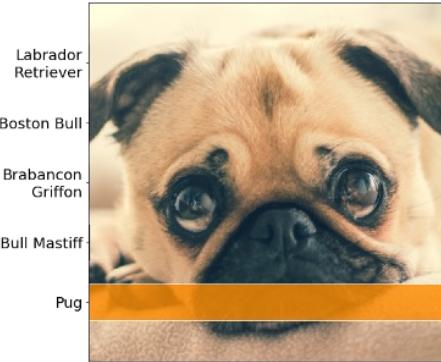


Robótica

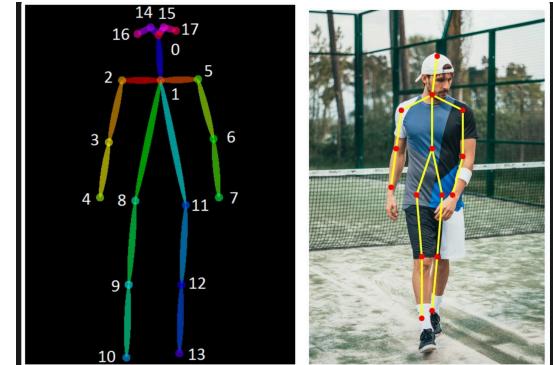
Visión por computador



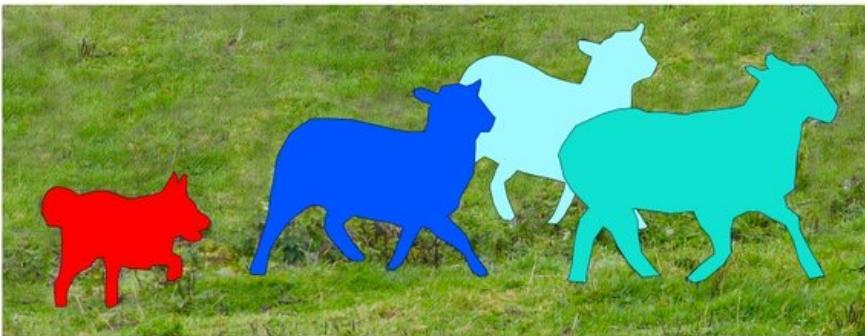
Clasificación / Regresión



Detección de objetos



Estimación de postura



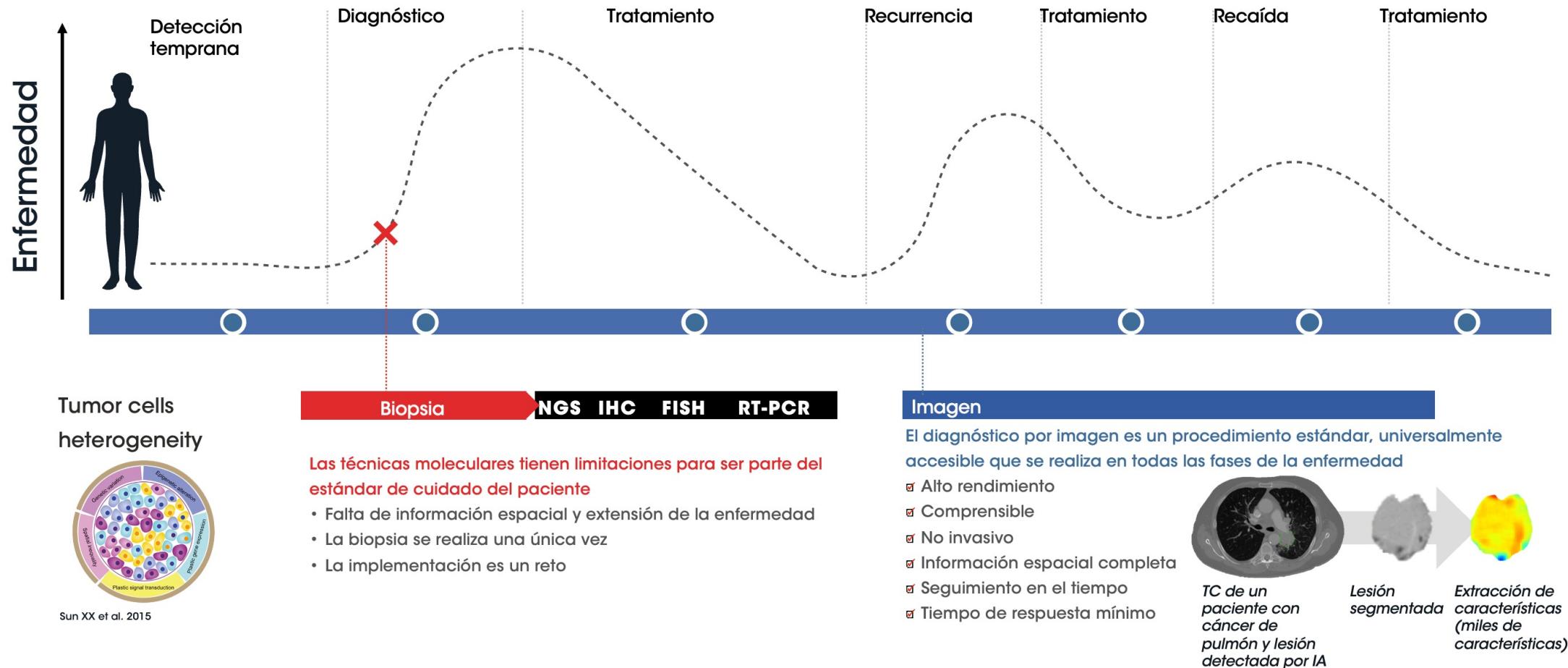
Segmentación de instancias



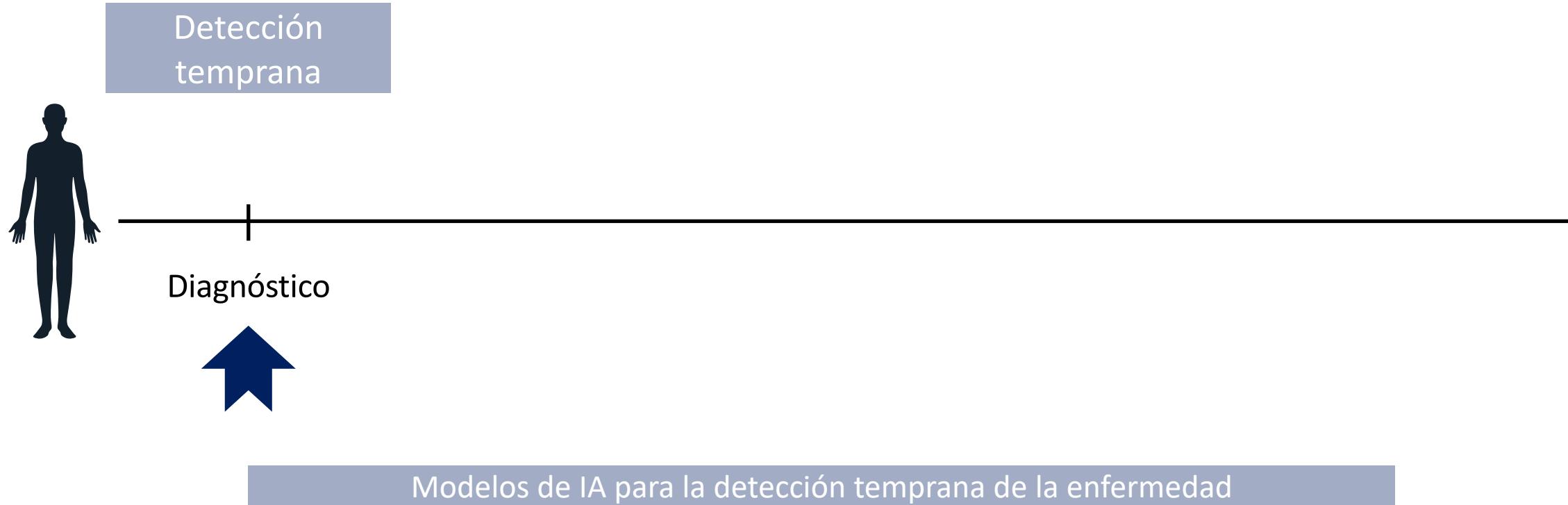
Segmentación semántica

Introducción IA en imagen médica

El camino del paciente



El camino del paciente



Detección temprana

Cribado del cáncer de mama: Mamografía cada 2 años entre los 50 y 69 años

Beneficios

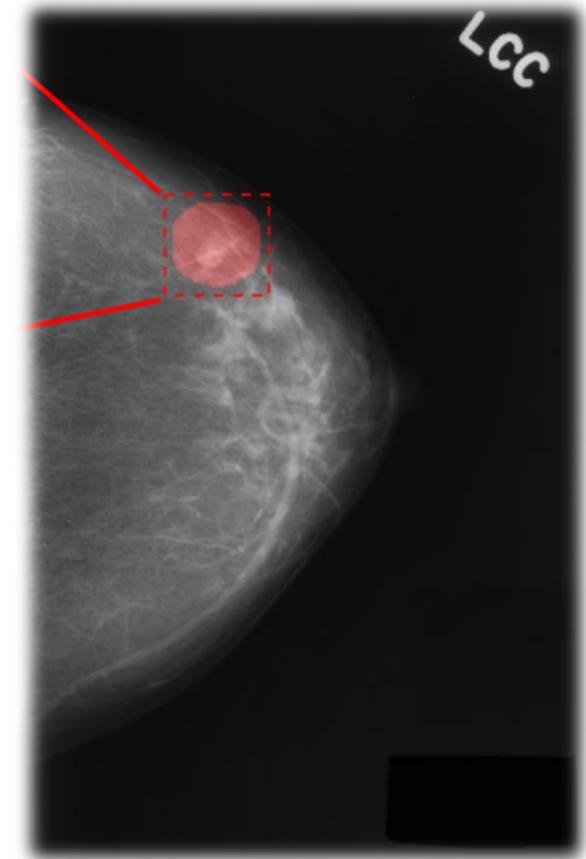
- Detección más temprana del cáncer.
- Reducción de la mortalidad (20-35%)
- Tratamientos menos agresivos.

Riesgos

- Elevado número de falsos positivos (1/3 mujeres).
- Sobrediagnóstico.
- Radiación.

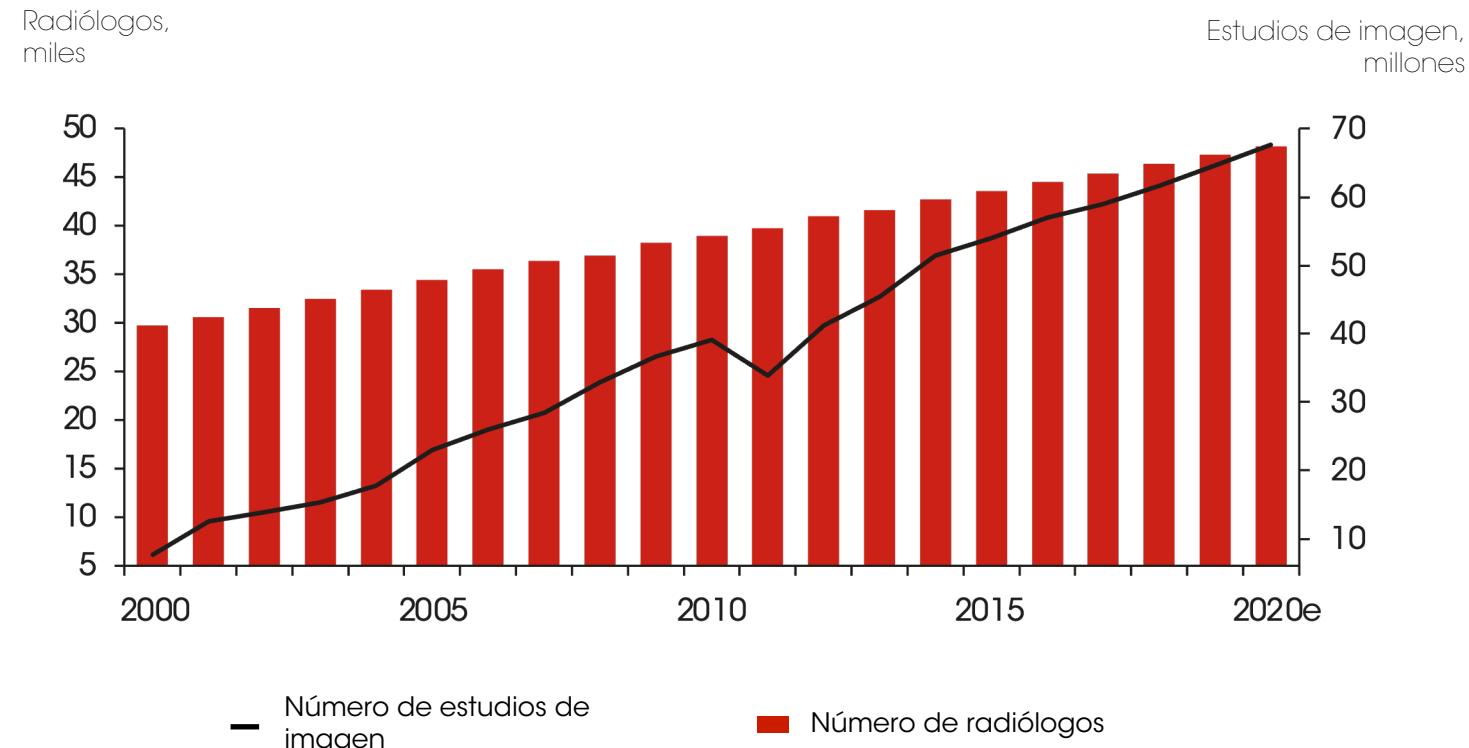
Problema

- Elevado número de mamografías por informar.
- En Malmö (Suecia): 65.000 mujeres estudiadas anualmente -> 2 lecturas -> 130.000 lecturas -> 50 imágenes/hora/radiólogo.



Fuente imagen: Lotter, William et al. (2017) A Multi-scale CNN and Curriculum Learning Strategy for Mammogram Classification.

Imagen médica



62% más radiólogos vs. 792% más imágenes médicas (EU 2000-2020)

Detección temprana



Los algoritmos de clasificación se pueden emplear como herramienta de **cribado de enfermedades**.

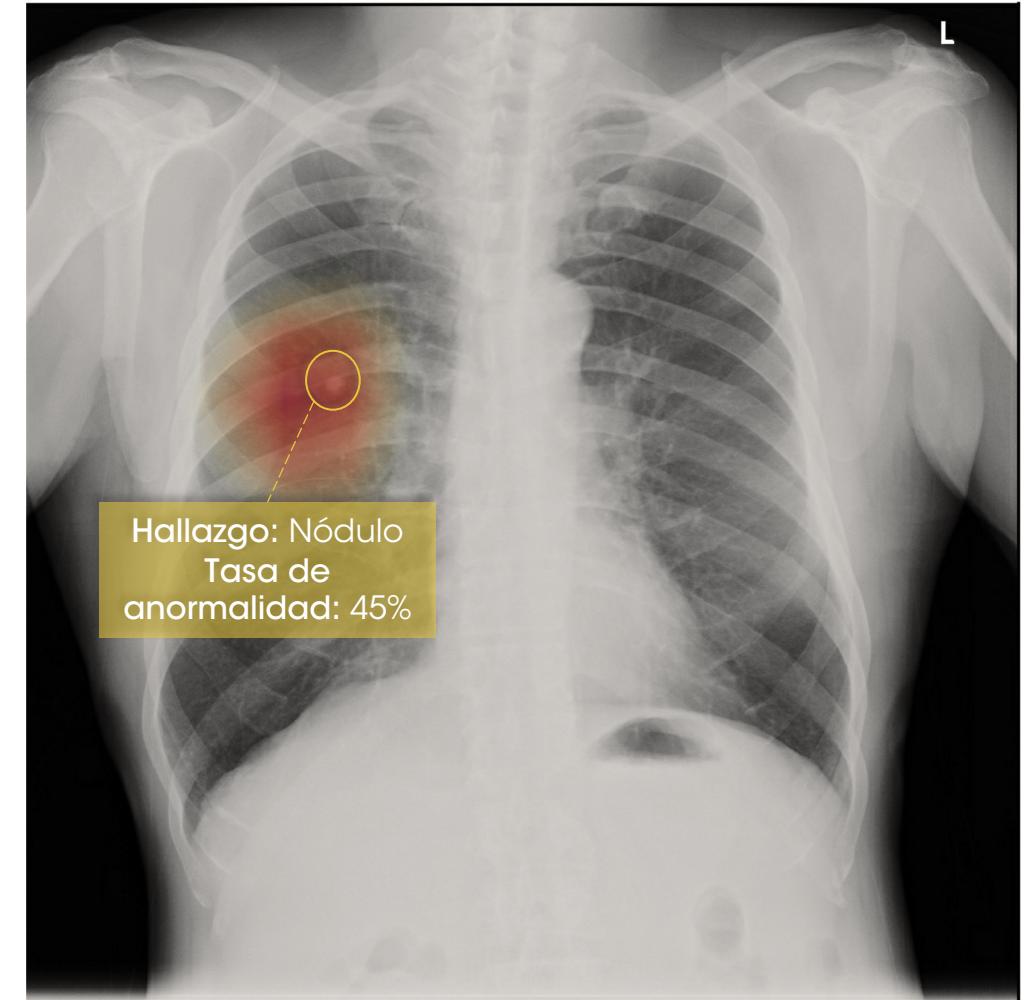
Radiografías de tórax:

Necesidad clínica

- La radiografía de tórax es la prueba de imagen más empleada en los hospitales.
- Los servicios de radiología tienen recursos limitados para informar todas las radiografías que se adquieren.

Solución

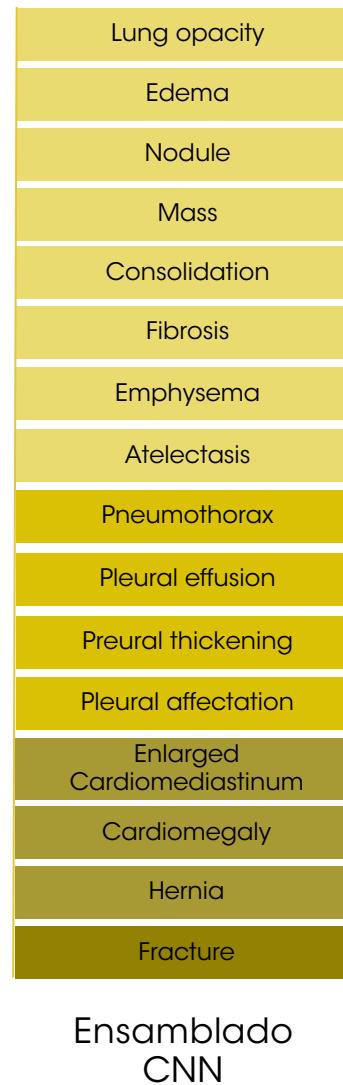
- Herramienta de cribado automático que ofrezca una probabilidad de anormalidad para ayudar en la priorización del informado de las imágenes.



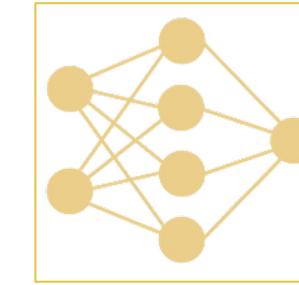
Detección temprana



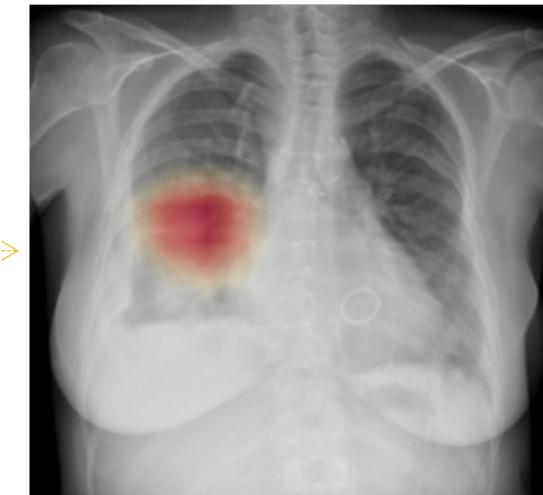
RX de tórax



Ensamblado CNN



Red totalmente conectada



Probabilidad patología: 0.87

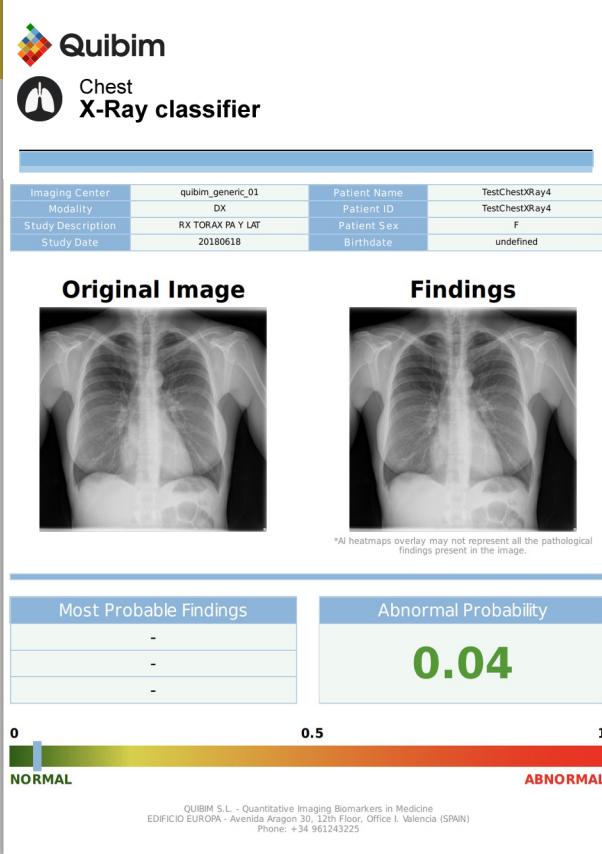
Probabilidad
anormalidad y
mapa de calor

Detección temprana



Ejemplo caso #1

Normal



Anormal

Ejemplo caso #2

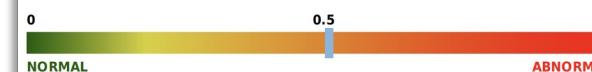
Quibim Chest X-Ray classifier

Imaging Center	quibim_generic_01	Patient Name	TestChestXRray
Modality	DX	Patient ID	TestChestXRray
Study Description	Chest PA View	Patient Sex	M
Study Date	20130930	Birthdate	undefined

Original Image **Findings**
*AI heatmaps overlay may not represent all the pathological findings present in the image.

Nodule
Mass
Fibrosis

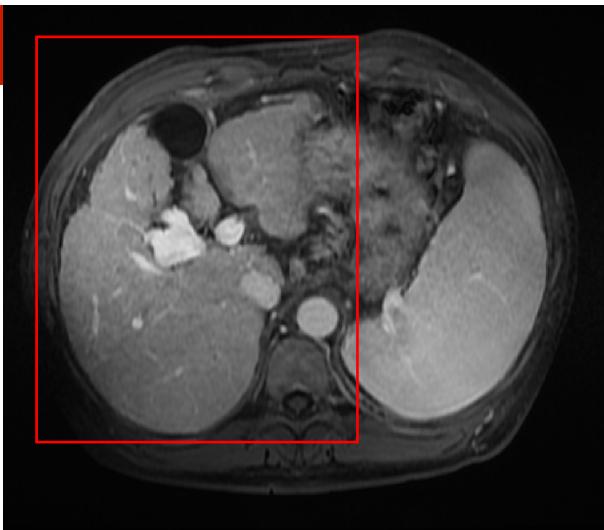
Abnormal Probability **0.51**



QUIBIM S.L. - Quantitative Imaging Biomarkers in Medicine
EDIFICIO EUROPA - Avenida Aragón 30, 12th Floor, Office I. Valencia (SPAIN)
Phone: +34 961243225

Otras aplicaciones

Detección



Segmentación



Otras aplicaciones

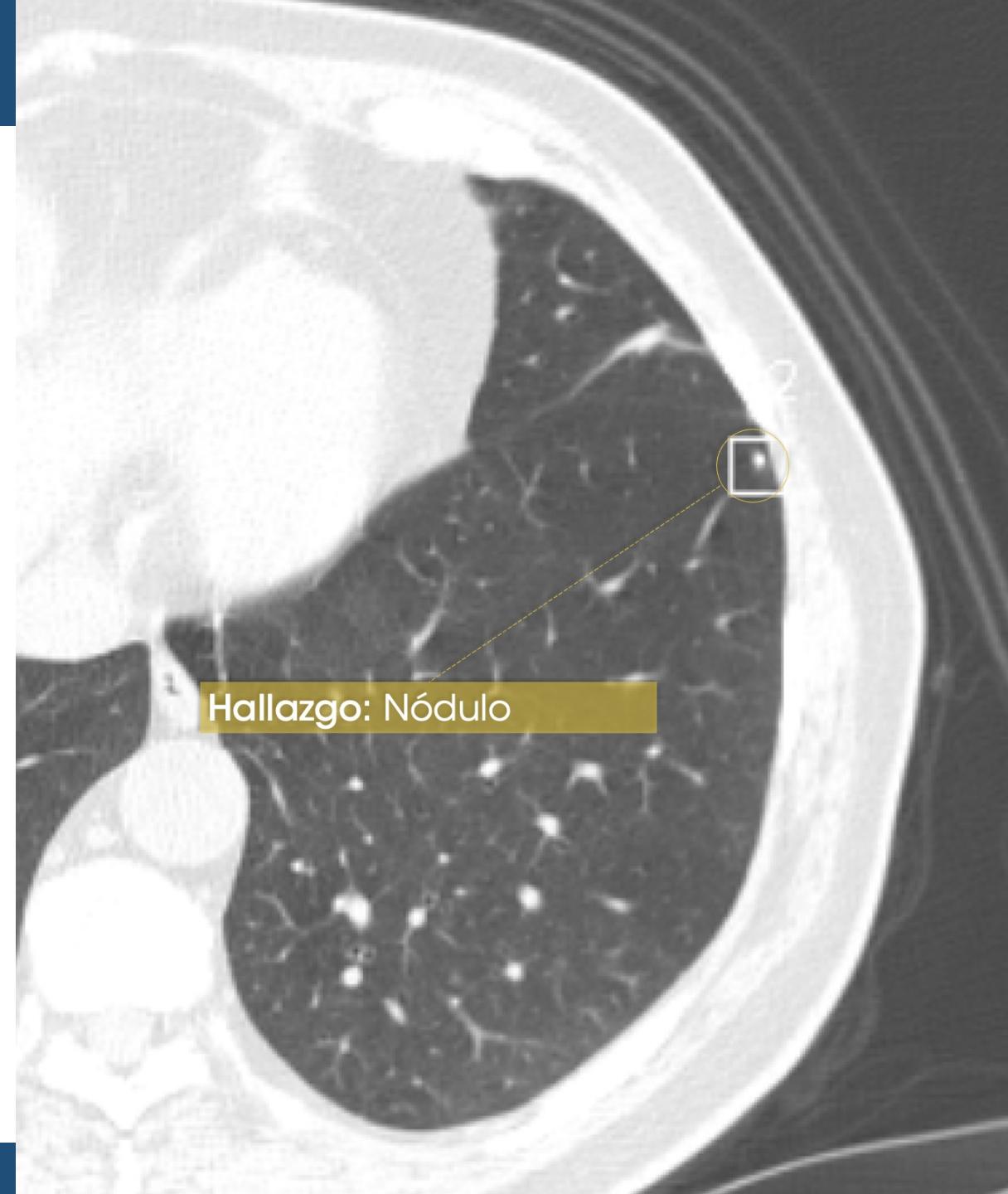
Detección de nódulos pulmonares

Necesidad clínica

- Entre el 10-30% de los nódulos pulmonares no se identifican en TC de screening de baja dosis.
- El 60% de los nódulos no llegan a consenso entre radiólogos.

Solución

- Herramienta automática para la detección de nódulos pulmonares en imágenes de TC.



Hallazgo: Nódulo

Otras aplicaciones

Detección de nódulos pulmonares

Necesidad clínica

- Entre el 10-30% de los nódulos pulmonares no se identifican en TC de screening de baja dosis.
- El 60% de los nódulos no llegan a consenso entre radiólogos.

Solución

- Herramienta automática para la detección de nódulos pulmonares en imágenes de TC.

 Quibim

Chest

Lung Nodule Detection

VERSION 1.0.0-research

Imaging Center	quibim_generic_01	Patient Name
Modality	CT	Patient ID
Study Description	TCcuello-torax-abdomen-pelvico...	Patient Sex
Study Date	20180621	Birthdate
		undefined

Finding 01

Size [mm]: 16.20
Volume [mL]: 3.37
Type: Part-Solid
Calc. [%]: 0.00
Lobe: Right Upper



Finding 02

Size [mm]: 8.96
Volume [mL]: 2.93
Type: Part-Solid
Calc. [%]: 0.00
Lobe: Not sure



Finding 03

Size [mm]: 10.44
Volume [mL]: 2.00
Type: Part-Solid
Calc. [%]: 0.00
Lobe: Left Upper



Finding 04

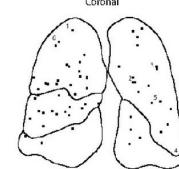
Size [mm]: 7.76
Volume [mL]: 1.77
Type: Solid
Calc. [%]: 0.00
Lobe: Not sure



Nodules Statistics

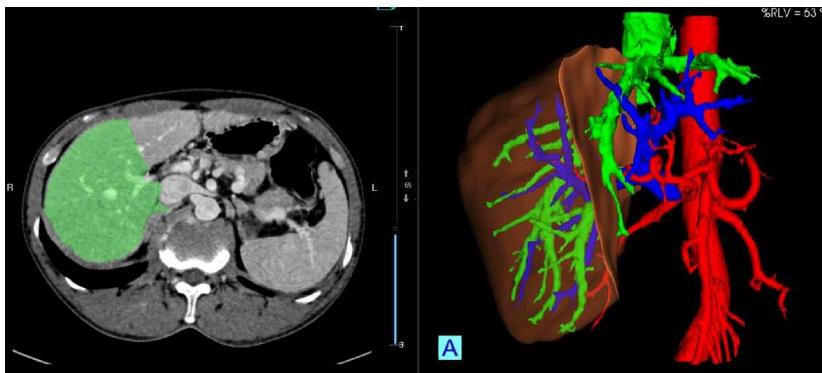
Total Number of nodules	62
Number of nodules - Right Upper	13
Number of nodules - Right Middle	7
Number of nodules - Right Lower	17
Number of nodules - Left Upper	10
Number of nodules - Left Lower	10
Dominant Nodule Size [mm]	10.20
Dominant Nodule Volume [mL]	3.37
Dominant Nodule Calcification [%]	0.00

Coronal

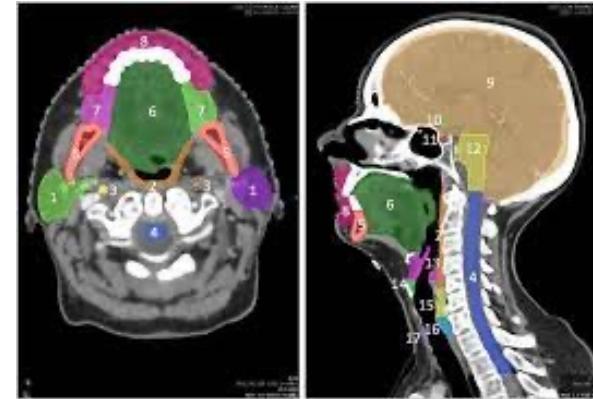


Otras aplicaciones

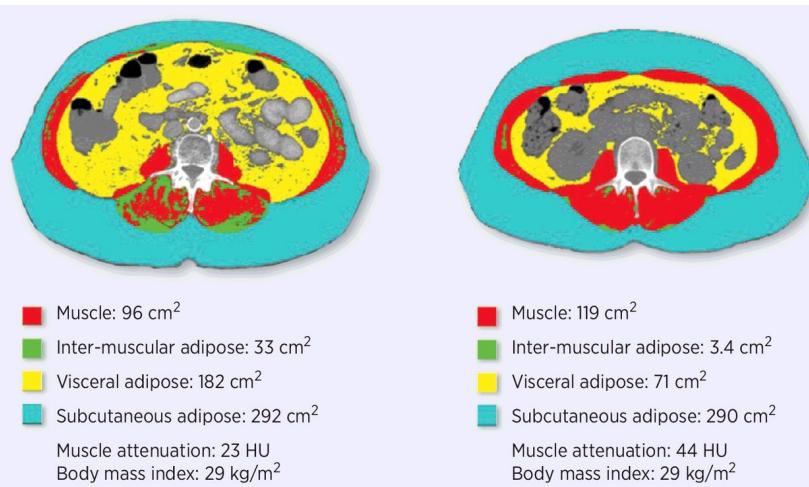
En radiología los métodos de **segmentación automática** son de gran importancia. Permiten automatizar procesos que son altamente costosos y **retrasan el flujo radiológico**.



Planificación quirúrgica



Radioterapia



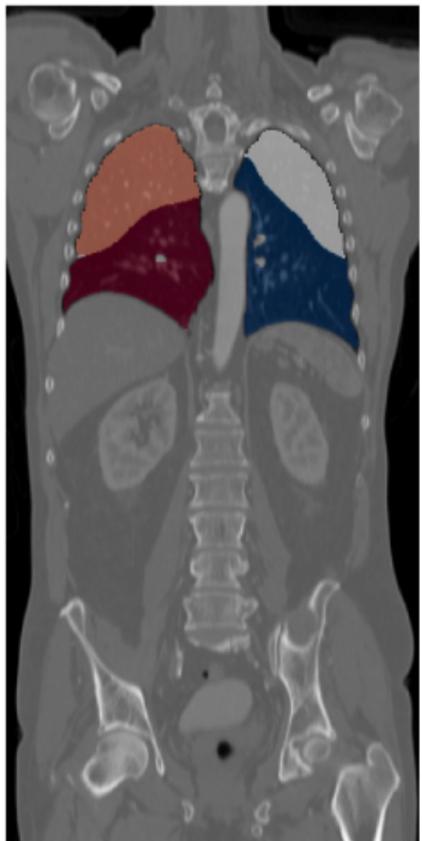
Composición corporal



Cuantificar

Otras aplicaciones

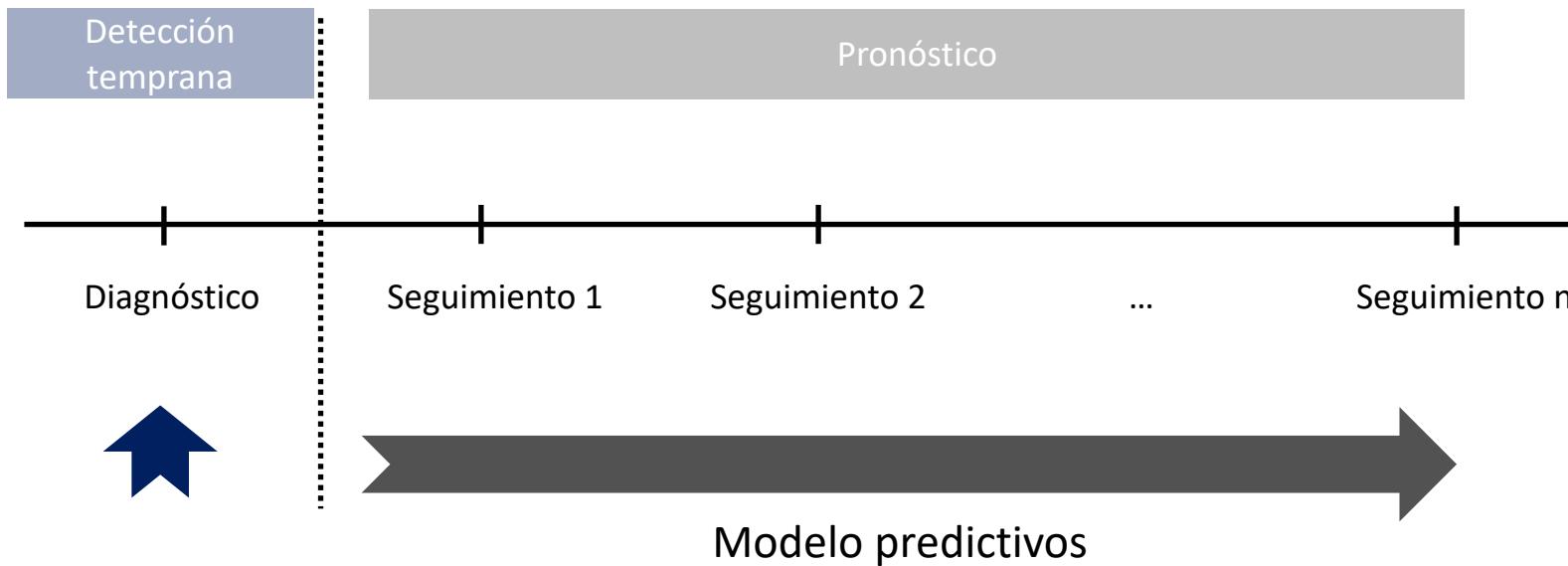
Ejemplo segmentación en cáncer de pulmón



Segmentación automática del pulmón y de la/s lesión/es:

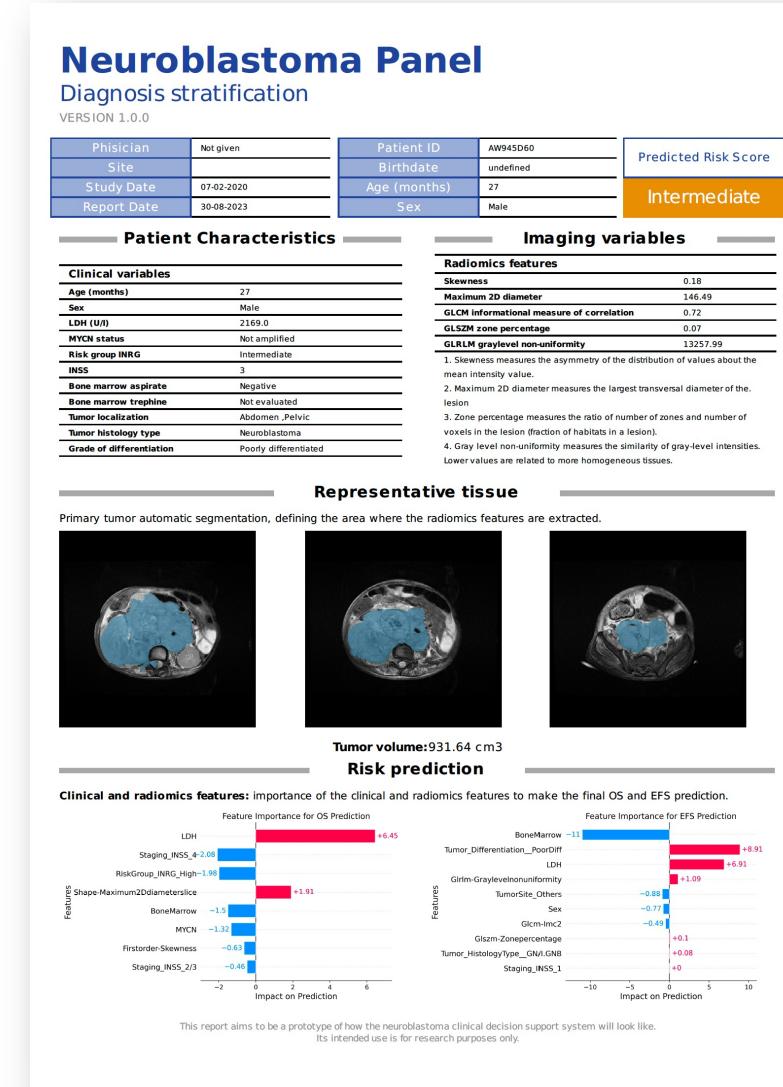
- Localización lesión
- Tamaño lesión
- Distancia a órganos vitales (e.g., corazón) o vasos importantes (e.g., aorta).

El camino del paciente



Modelos predictivos

- Pacientes pediátricos con diagnóstico de neuroblastoma.
 - Imagen de RM a diagnóstico.
 - Combinación con otra información del paciente: datos clínicos y de laboratorio.
-
- Predicción del riesgo del paciente --> Tratamiento personalizado.
 - Interpretabilidad de resultados para incrementar la confianza en el modelo.



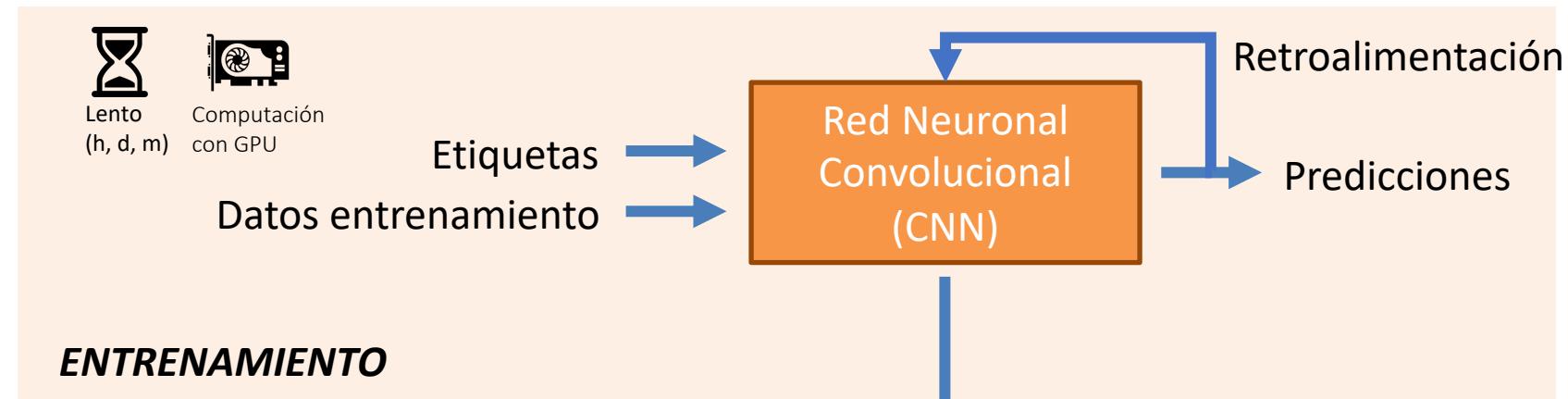
Entrenamiento de modelos y puesta en producción

En esta sesión vamos a ver como es el día a día de un empleado en Quibim en el **flujo de desarrollo** de un modelo de IA basado en redes neuronales convolucionales. Vamos a ver las distintas fases desde el entrenamiento de los modelos hasta su preparación para la puesta en producción.

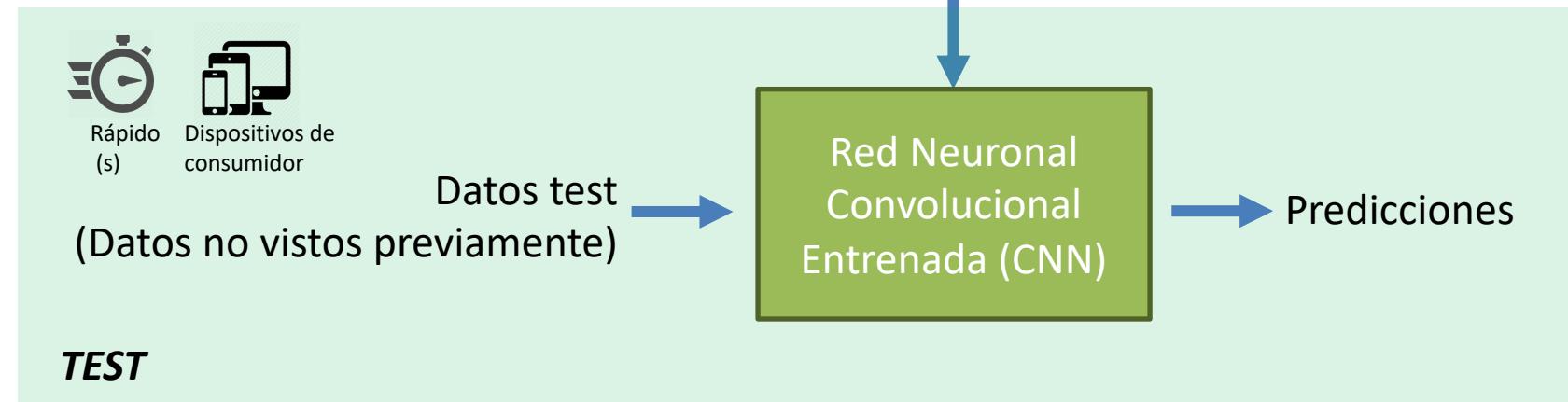


Entrenamiento vs. Test

Entrenamiento
~80%



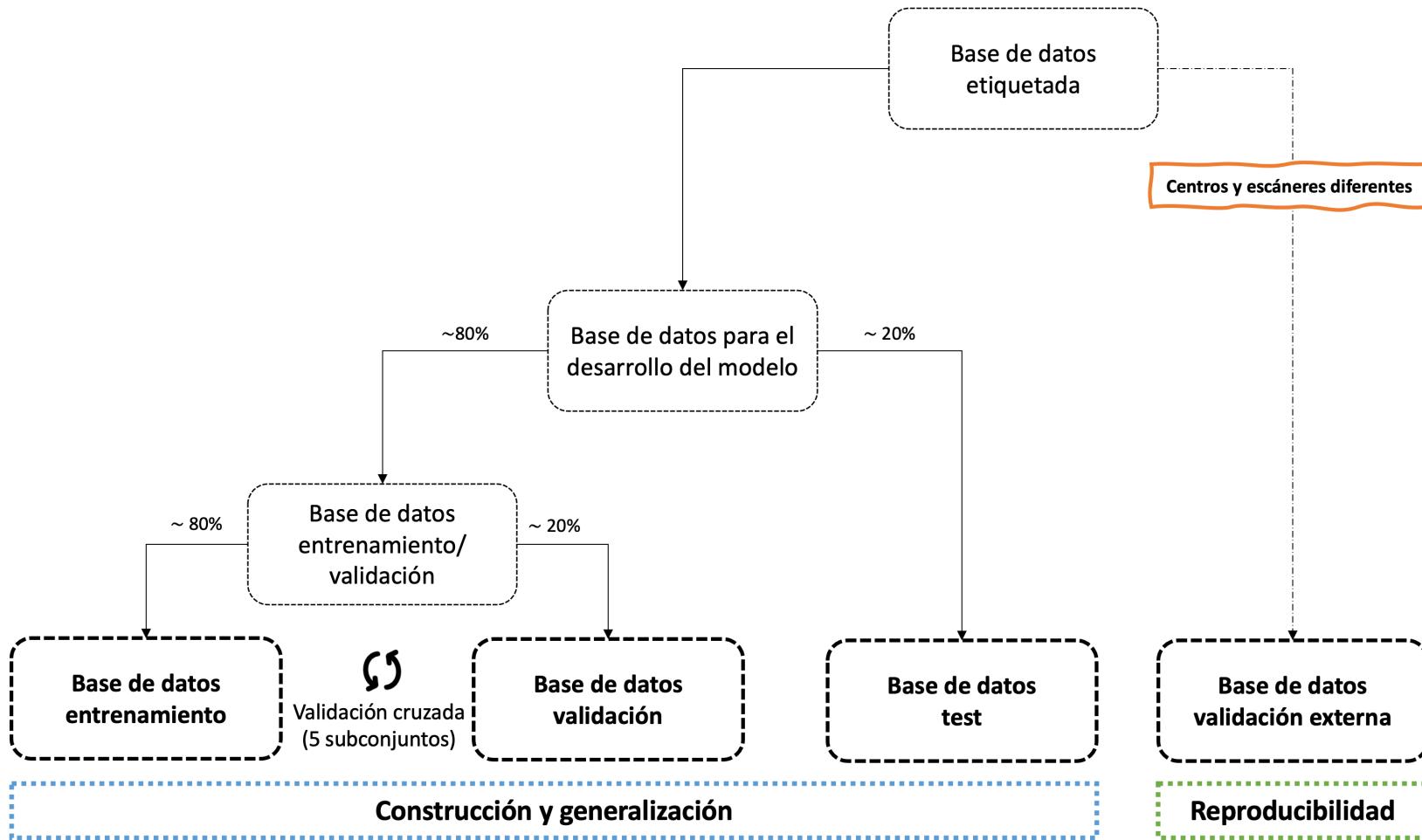
Test
~20%



Proceso de entrenamiento

Entrenamiento

Paso 1: Preparación de la base de datos



Entrenamiento

Paso 1: Preparación de la base de datos

¡Es importante mantener estas particiones constantes entre pruebas para realizar comparaciones apropiadas!



Entrenamiento (70% aprox.)

Conjunto de datos que se emplean para ajustar los pesos del modelo.

Validación (15% aprox.)

Conjunto de datos que se emplean, durante el proceso de entrenamiento, para evaluar el rendimiento del modelo en un conjunto de datos nuevo. Permite controlar el sobreajuste. Además se emplea para la optimización de los hiperparámetros y, en ocasiones, para seleccionar los pesos de la época que mejores resultados ofrece.

Test (15% aprox.)

Conjunto de datos que se emplean, una vez finalizado el entrenamiento, para evaluar el rendimiento del modelo en un conjunto de datos diferente al de entrenamiento y test. Esta evaluación adicional es importante para eliminar el sesgo introducido al emplear el conjunto de validación para optimizar los hiperparámetros.

Entrenamiento

Paso 1: Preparación de la base de datos

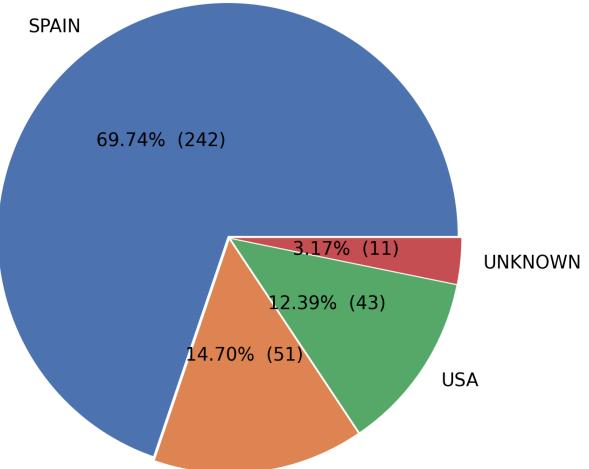
NOMBRE REAL	SEGMENTADAS	Numeración	Procedencia	TRAINING/ VALIDATION/ TEST	fabricante
11-Batch1	X	237	INDIA	TRAINING	GE
12-Batch1	X	238	INDIA	TRAINING	GE
13-Batch1	X	239	INDIA	TRAINING	GE
14-Batch1	X	240	INDIA	TEST	GE
15-Batch1	X	241	INDIA	VALIDACIÓN	GE
2-Batch2	X	242	INDIA	VALIDACIÓN	PHILIPS
3-Batch2	X	243	INDIA	TEST	PHILIPS
4-Batch2	X	244	INDIA	TRAINING	PHILIPS
5-Batch2	X	245	INDIA	TEST	PHILIPS
6-Batch2	X	246	INDIA	TRAINING	PHILIPS
7-Batch2	X	247	INDIA	VALIDACIÓN	PHILIPS
8-Batch2	X	248	INDIA	TRAINING	PHILIPS
9-Batch2	X	249	INDIA	TEST	PHILIPS
10-Batch2	X	250	INDIA	TEST	PHILIPS
11-Batch2	X	251	INDIA	TRAINING	PHILIPS
12-Batch2	X	252	INDIA	TEST	PHILIPS
13-Batch2	X	253	INDIA	TEST	PHILIPS
14-Batch2	X	254	INDIA	VALIDACIÓN	PHILIPS
15-Batch2	X	255	INDIA	TRAINING	SIEMENS
	X	304	SPAIN	TEST	GE
	X	305	SPAIN	TRAINING	GE
	X	306	SPAIN	TRAINING	GE
	X	307	SPAIN	TRAINING	GE
	X	308	SPAIN	TRAINING	GE
	X	309	SPAIN	TRAINING	GE
	X	310	SPAIN	TRAINING	GE
	X	311	SPAIN	TRAINING	GE
	X	312	SPAIN	TRAINING	GE
	X	313	SPAIN	TEST	GE
	X	314	SPAIN	VALIDACIÓN	GE
	X	315	SPAIN	VALIDACIÓN	GE
	X	316	SPAIN	TRAINING	GE
	X	317	SPAIN	TRAINING	GE
	X	318	SPAIN	TRAINING	GE
	X	319	SPAIN	TEST	Siemens
	X	320	SPAIN	TRAINING	GE
	X	321	SPAIN	TRAINING	GE
	X	322	SPAIN	TRAINING	GE
	X	323	SPAIN	TRAINING	GE

Entrenamiento

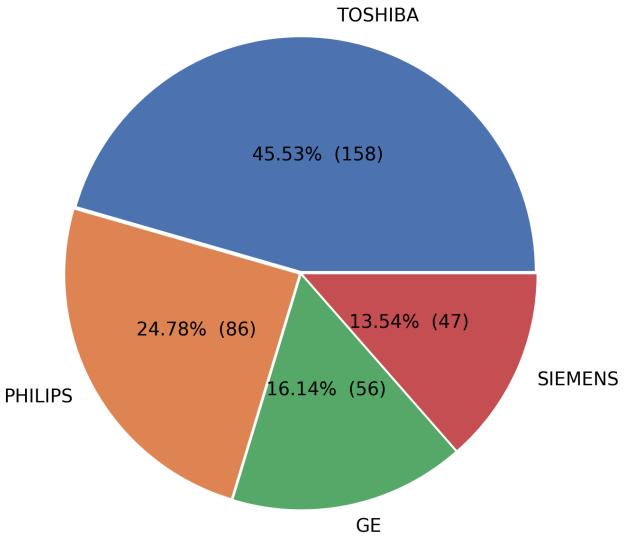
Paso 1: Preparación de la base de datos

Realizar las particiones incluyendo variabilidad en las bases de datos, especialmente en la de test para garantizar la reproducibilidad del método.

ORIGIN OF THE TRAINING DATA (N = 347)

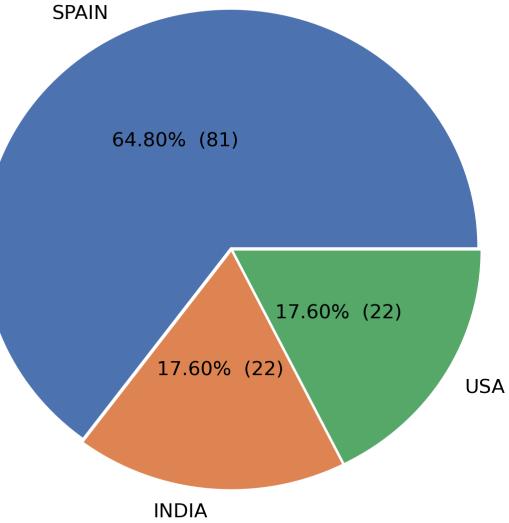


IMAGING VENDORS TRAINING DATASET (N = 347)

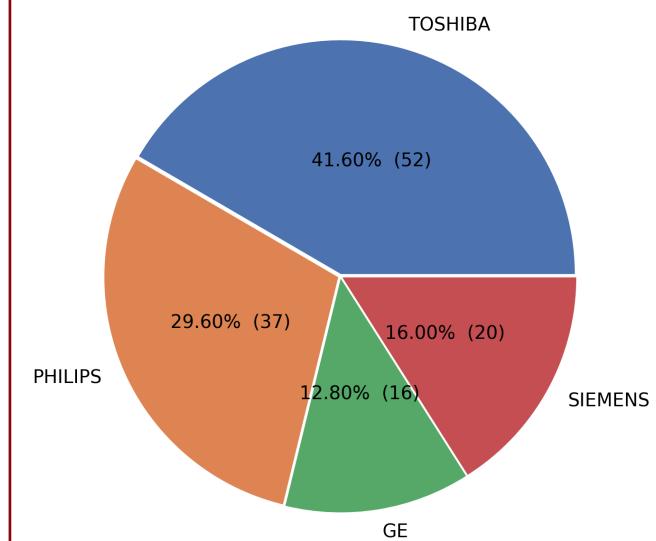


ENTRENAMIENTO

ORIGIN OF THE TESTING DATA (N = 125)



IMAGING VENDORS TESTING DATASET (N = 125)



TEST

Entrenamiento

Paso 2: Ajuste de hiperparámetros

La fase de ajuste de hiperparámetros hace referencia al diseño y selección de todo lo que debemos configurar manualmente antes del proceso de entrenamiento:

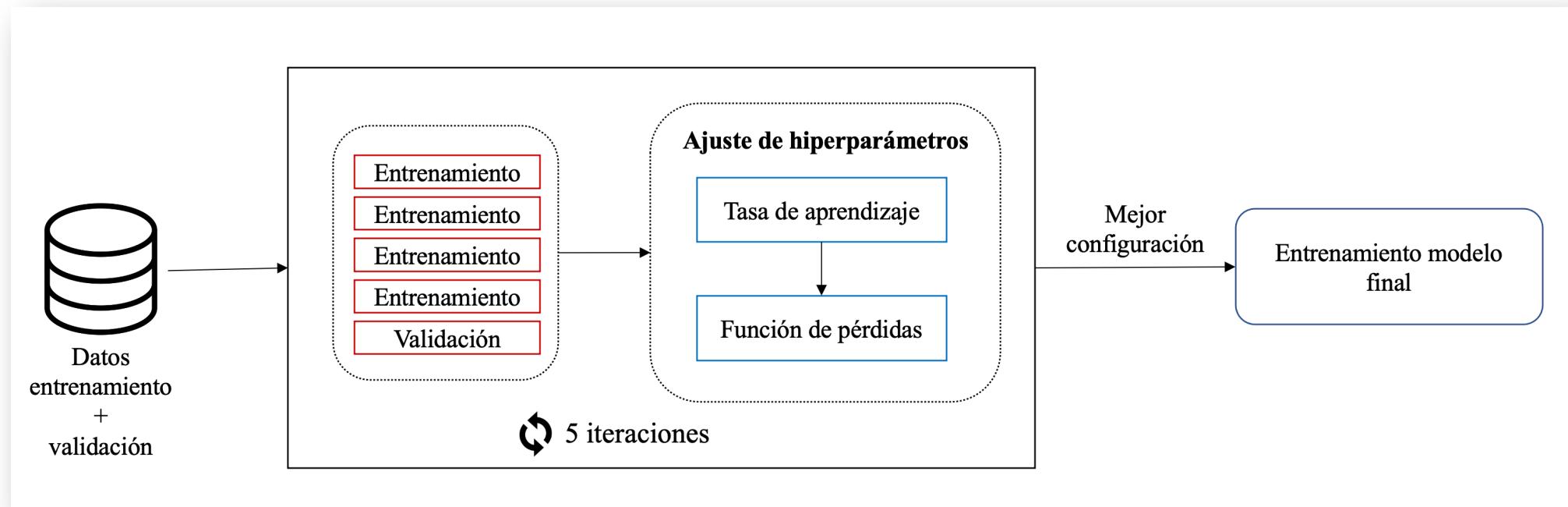
- Arquitectura:
 - Número de capas
 - Tipo de capa
 - Número de filtros por capa
 - ...
- Función de coste
- Tasa de aprendizaje
- Regularización
- Tamaño de lote

RECOMENDACIONES AL DISEÑAR UNA CNN

- Arquitectura → Parte de una ya existente
- Función de coste → Va a depender del problema a resolver.
 - Clasificación/regresión → Nada que probar
 - Segmentación → Aquí si debemos probar varias hasta encontrar la más apropiada
- Tasa de aprendizaje → Deberemos probar varias hasta encontrar la más apropiada
- Regularización → Prueba añadir o no para ver que funciona mejor.
- Tamaño de lote → Auméntalo lo máximo posible (hasta obtener error de memoria).

Entrenamiento

Paso 2: Ajuste de hiperparámetros



Ejemplo diseño de un sistema de ajuste de hiperparámetros para un modelo de segmentación.

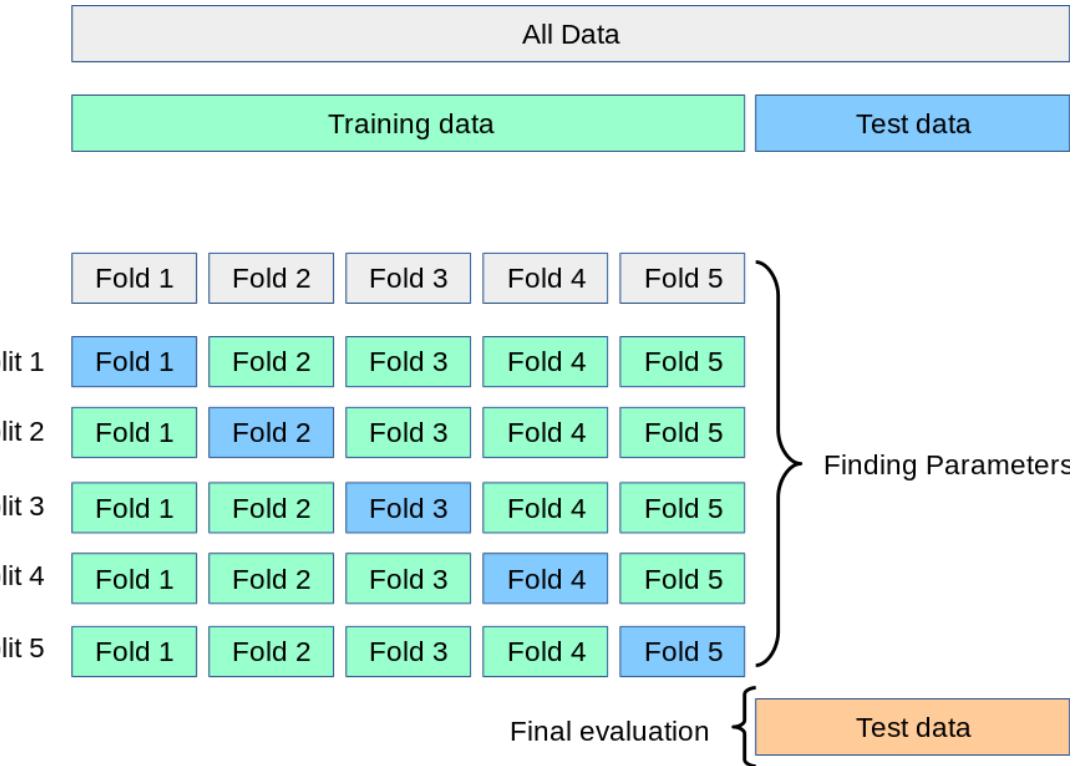
Entrenamiento

Paso 2: Ajuste de hiperparámetros

En el ejemplo anterior se emplea un algoritmo de validación cruzada (*k-fold cross validation*) para la validación del modelo.

La base de datos de entrenamiento+validación se divide en k particiones (5 en nuestro caso) y la CNN se entrena 5 veces, empleando en cada caso, $k-1$ (4 en nuestro caso) particiones para entrenar y 1 para validar.

Este proceso permite analizar como funciona la arquitectura diseñada ante diferentes conjuntos de datos.



Entrenamiento

Paso 2: Ajuste de hiperparámetros

Este proceso también los podemos hacer de manera más manual, sobretodo cuando el problema es más complejo y no sabemos "por donde tirar".

1	Setup	Name	Parameter	Values	Status	Skill Validation	Question	Findings
11	4 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, Freeze 90% Resnet, Normalization ImageNet	experiment 10	Labels	3	done		Does using 3 labels allows generalization?	Did not work
12	4 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, Freeze 90% Resnet, Normalization ImageNet	experiment 11	Labels	2	done		Does using 2 (Healthy / Not Healthy) labels allows generalization?	Did not work
13	4 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Not norm	experiment 12	Unfr and not Normalization		done		Allows generalization?	Did not work
14	4 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Not norm	experiment 13	Resnet Layers	Freeze 95%	done		Allows generalization?	Did not work
15	4 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Not norm	experiment 14	Resnet Layers	Freeze 99%	done		Allows generalization?	Not fitting
16	20 epochs, batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Not norm	experiment 15	Resnet Layers	Freeze 97%	done		Allows generalization?	Overfitting
17	4 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Not norm	experiment 16	Resnet Layers	Freeze 98%	done		Allows generalization?	Overfitting
18	20 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Unfreeze, Not norm	experiment 17	Lung seg		done		Allows generalization?	Overfitting
19	24 epochs, batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Unfreeze, Not norm, Lung seg	experiment 18	3D model	2+1D Resnet18	done		Allows generalization?	Overfitting
20	7 epochs (stopped manually), batch size 1, LR 1e-5, Nx224x224, 1 GPU, 2 labels, Unfreeze, Norm, Lung seg	experiment 19	Resnet Layers	Freeze 100%	done		Allows generalization?	Not learning
21	batch size 1, LR 1e-3, 2 labels, 32x224x224, 1 GPU, Freeze Resnet and UnFreeze Resnet	experiment 20	Dummy data		done		Was the system working properly?	The dummy experiment helped to show that the Freeze was not done properly
22	batch size 1, LR 1e-3, Nx224x224, 1 GPU, 2 labels, Norm, Lung seg, No dataaug	experiment 21	Resnet Layers	Freeze 100% properly	done		Is the system with Frozen layers learning now?	Yes, but with overfitting, let's activate the data augmentation
23	batch size 1, LR 1e-3, Nx224x224, 1 GPU, 2 labels, Norm, Lung seg, Dataaug	experiment 22	Resnet Layers	Freeze 100% properly	done		Allows generalization?	
24	batch size 1, LR 1e-3, Nx224x224, 1 GPU, Freeze, Norm, Lung seg, Dataaug	experiment 23	Labels	3	done		How are results?	Bad, very reduced learning
25	batch size 1, LR 1e-4, Nx224x224, 1 GPU, 3 labels, Freeze, Norm, Lung seg, Dataaug	experiment 24	Balanced	81 samples per class	done		Forcing balance dataset improves distribution?	Not at all, worsen results
26	batch size 1, LR 1e-4, Nx224x224, 1 GPU, 3 labels, Freeze, Norm, Lung seg, Dataaug	experiment 25	3D model	2+1D Resnet18	done		Does a 3D net properly frozen working?	It works worse than the 2D approach
27								
28								
29	50 epochs, batch size 10x1, LR 1e-4, 64x224x224, 1 GPU, 3 labels, No Freeze, No Norm, No Lung seg, Dataaug	experiment 1	3D simple model + batch size > 1	4 3D conv blocks	done	val acc 0.50	does batch size > 1 and a simple model provide a normal behavior?	Yes!!!! It is finally learning
30	50 epochs, batch size 10x8, LR 1e-3, 64x224x224, 8 GPU, 2 labels (normal vs abnormal), No Freeze, No Norm, No Lung seg, Dataaug	experiment 2	multi class classes_lr	8.2e-3	done	val acc 0.70	is the 2 class problem working well?	Kind of, but more improvement is needed to compete with the 2D approach

Experimento 29!

Entrenamiento

Paso 2: Ajuste de hiperparámetros

También existen herramientas de monitorización de modelos.



Weights & Biases



Entrenamiento

ai-dev > Projects > test-project-3 > Table

Runs (20)

Sort | Tag | Move | Create Sweep | Columns

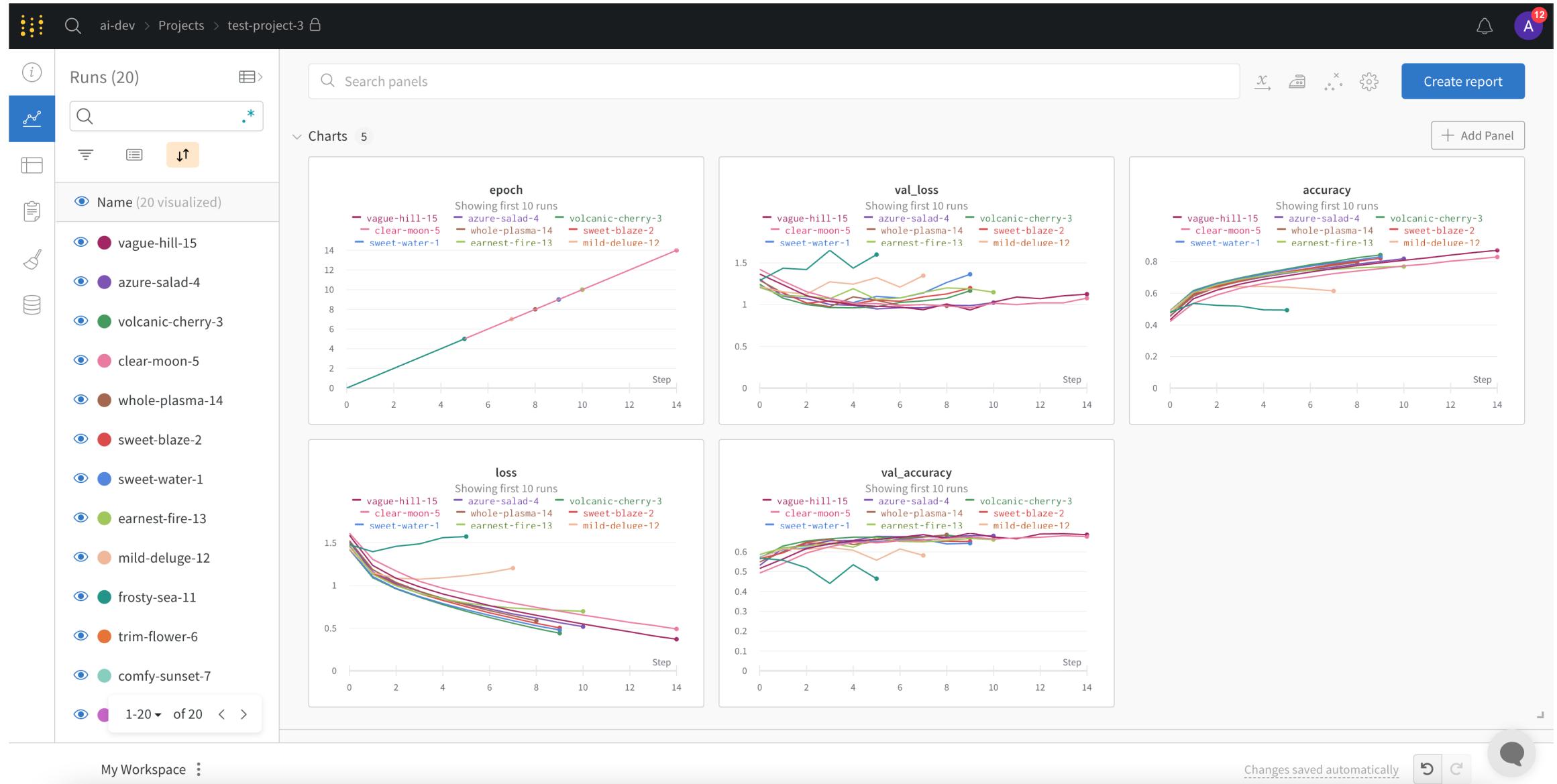
	Name (20 visualized)	Notes	User	Tags	Created	Runtime	Sweep	batch_size	optimizer	GFLOPs	accuracy	best_epoch	best_val_	epoch	loss	val_accurac	val_loss
·	vague-hill-15	Add notes	anjipas		4w ago	1m 10s	-	64	rmsprop	0.002092	0.872	9	0.9356	14	0.3701	0.6863	1.125
·	azure-salad-4	Add notes	anjipas		4w ago	1m 13s	-	32	adam	0.002092	0.8188	5	0.9483	10	0.5174	0.6806	1.024
·	volcanic-cherry-3	Add notes	anjipas		4w ago	1m 47s	-	16	adam	0.002092	0.8428	4	0.9613	9	0.4405	0.671	1.166
·	clear-moon-5	Add notes	anjipas		4w ago	58s	-	64	adam	0.002092	0.8299	9	0.962	14	0.4903	0.6771	1.077
·	whole-plasma-14	Add notes	anjipas		4w ago	1m 7s	-	32	rmsprop	0.002092	0.7964	3	0.9758	8	0.5852	0.6855	0.9856
·	sweet-blaze-2	Add notes	anjipas		4w ago	3m 8s	-	8	adam	0.002092	0.821	4	1.01	9	0.5031	0.6522	1.198
·	sweet-water-1	Add notes	anjipas		4w ago	5m 56s	-	4	adam	0.002092	0.8309	4	1.025	9	0.4783	0.6433	1.363
·	earnest-fire-13	Add notes	anjipas		4w ago	2m 14s	-	16	rmsprop	0.002092	0.769	5	1.062	10	0.697	0.6621	1.149
·	mild-deluge-12	Add notes	anjipas		4w ago	2m 56s	-	8	rmsprop	0.002092	0.6147	2	1.125	7	1.203	0.5819	1.347
·	frosty-sea-11	Add notes	anjipas		4w ago	4m 10s	-	4	rmsprop	0.002092	0.4934	0	1.29	5	1.573	0.4648	1.599
·	trim-flower-6	Add notes	anjipas		4w ago	12m 29s	-	4	adagrad	0.002092	0.521	19	1.399	19	1.372	0.5055	1.399
·	comfy-sunset-7	Add notes	anjipas		4w ago	6m 24s	-	8	adagrad	0.002092	0.5002	19	1.451	19	1.43	0.4933	1.451
·	apricot-glade-8	Add notes	anjipas		4w ago	4m 29s	-	16	adagrad	0.002092	0.4857	19	1.485	19	1.462	0.4785	1.485
·	classic-lion-9	Add notes	anjipas		4w ago	2m 28s	-	32	adagrad	0.002092	0.4612	19	1.552	19	1.538	1-20 ▾ of 20 < >	

My Workspace :

Changes saved automatically



Entrenamiento



Entrenamiento

The screenshot shows a file browser interface with the following details:

- Path:** ai-dev > Projects > test-project-3 > Runs > vague-hill-15 > Files
- Search Bar:** Search (with placeholder "Search")
- File List:**
 - artifact /**: 9 subfolders, 0 files
 - media /**: 1 subfolder, 0 files
 - config.yaml**: 4 weeks ago, 578.0B, download link
 - model-best.h5**: 4 weeks ago, 1.4MB, download link
 - output.log**: 4 weeks ago, 3.0KB, download link
 - requirements.txt**: 4 weeks ago, 7.4KB, download link
 - wandb-metadata.json**: 4 weeks ago, 615.0B, download link
 - wandb-summary.json**: 4 weeks ago, 516.0B, download link
- Sidebar Icons:** (from top to bottom) i, ⚙️, 🏠, ➔, 📄 (selected), 📰
- User Profile:** A purple circle with a white letter 'A' and a red notification badge with the number '12'.
- Bottom Right:** A small speech bubble icon.

Entrenamiento

The screenshot shows a web-based machine learning model registry interface. The top navigation bar includes a search bar, project navigation (ai-dev > Projects > model-registry), and a user profile icon with a red notification badge (A 12). The main content area displays the details for a specific artifact named "cifar-10".

Artifacts sidebar:

- Model Registry →
- Find matching artifacts
- MODEL
- cifar-10
 - v1 latest (selected)
 - v0

cifar-10 Version 1 Overview tab (selected):

Collection

Type	model	Owner Entity	ai-dev
Alias Count	1	Owner Project	model-registry
		Created At	September 25th, 2022

Description: Add a markdown description...

Version

Full Name	ai-dev/model-registry/cifar-10:v1	Created At	September 25th, 2022 17:03:25
Aliases	latest v1 +	Num Consumers	0
Digest	7f033d8dbeab665ff09cedfb23908aee	Num Files	2
Source Version	model-fearless-puddle-20:v19	Size	2.2MB
Created By	fearless-puddle-20		
Description	What changed in this version?		

Entrenamiento

The screenshot shows a user interface for managing machine learning artifacts. The left sidebar has a blue header labeled "Artifacts" with icons for search, model registry, and a database. The main area shows a breadcrumb path: ai-dev > Projects > model-registry > Artifacts > model > cifar-10 > v1 > metadata. The title is "cifar-10" and the version is "Version 1". The tabs are Overview, Metadata (selected), Usage, Files, and Lineage. A search bar at the top right says "Search keys". Below it is a table with columns "Key" and "Value". The table contains two sections: "Run Config" and "Run History at Log Step".

Key	Value
batch_size	64
optimizer	"adadelta"
accuracy	0.24017499387264252
epoch	19
loss	2.150836229324341
val_accuracy	0.24369999766349792
val_loss	2.143587589263916

Entrenamiento

Paso 3: Entrenamiento/validación



NVIDIA DGX

- Entrenamientos más rápidos
- Bases de datos más grandes
- **55k€**

SYSTEM SPECIFICATIONS		
	NVIDIA DGX Station A100 320GB	NVIDIA DGX Station A100 160GB
GPUs	4x NVIDIA A100 80 GB GPUs	4x NVIDIA A100 40 GB GPUs
GPU Memory	320 GB total	160 GB total
Performance	2.5 petaFLOPS AI 5 petaOPS INT8	
System Power Usage	1.5 kW at 100–120 Vac	
CPU	Single AMD 7742, 64 cores, 2.25 GHz (base)–3.4 GHz (max boost)	
System Memory	512 GB DDR4	
Networking	Dual-port 10Gbase-T Ethernet LAN Single-port 1Gbase-T Ethernet BMC management port	
Storage	OS: 1x 1.92 TB NVME drive Internal storage: 7.68 TB U.2 NVME drive	
DGX Display Adapter	4 GB GPU memory, 4x Mini DisplayPort	
System Acoustics	<37 dB	
Software	Ubuntu Linux OS	
System Weight	91.0 lbs (43.1 kgs)	
Packaged System Weight	127.7 lbs (57.93 kgs)	
System Dimensions	Height: 25.1 in (639 mm) Width: 10.1 in (256 mm) Length: 20.4 in (518 mm)	
Operating Temperature Range	5–35 °C (41–95 °F)	

Entrenamiento

Paso 3: Entrenamiento/validación

¿Entonces si no tengo un *cluster* con muchas GPU no puedo entrenar modelos de CNN?

La mayoría de ordenadores cuentan con una tarjeta gráfica de NVIDIA (especialmente los de *Gaming*), ya solo haciendo uso de esta vamos a notar una diferencia significativa frente a hacer uso directamente de la CPU.



NVIDIA GeForce RTX 3050 –
4GB RAM GPU



NVIDIA GeForce RTX 3070 –
8GB RAM GPU

- GeForce RTX 2060: 6 GM RAM GPU
- GeForce GTX 1650: 4 GB RAM GPU
- GeForce GTX 1660 Ti: 6 GB RAM GPU

...

Entrenamiento

Paso 3: Entrenamiento/validación

Ejercicio 1:

Vamos a implementar y entrenar en Colab un modelo empleando CPU y GPU de manera independiente.

Para ello, abre el [link](#) (también está en Moodle) y ejecuta las celdas.

¿Cuánto tiempo ha tardado el entrenamiento?

Una vez finalizado el entrenamiento, cambia el entorno de ejecución:

Entorno de ejecución → Cambiar tipo de entorno de ejecución → En aceleración por hardware seleccionar “GPU” → Guardar

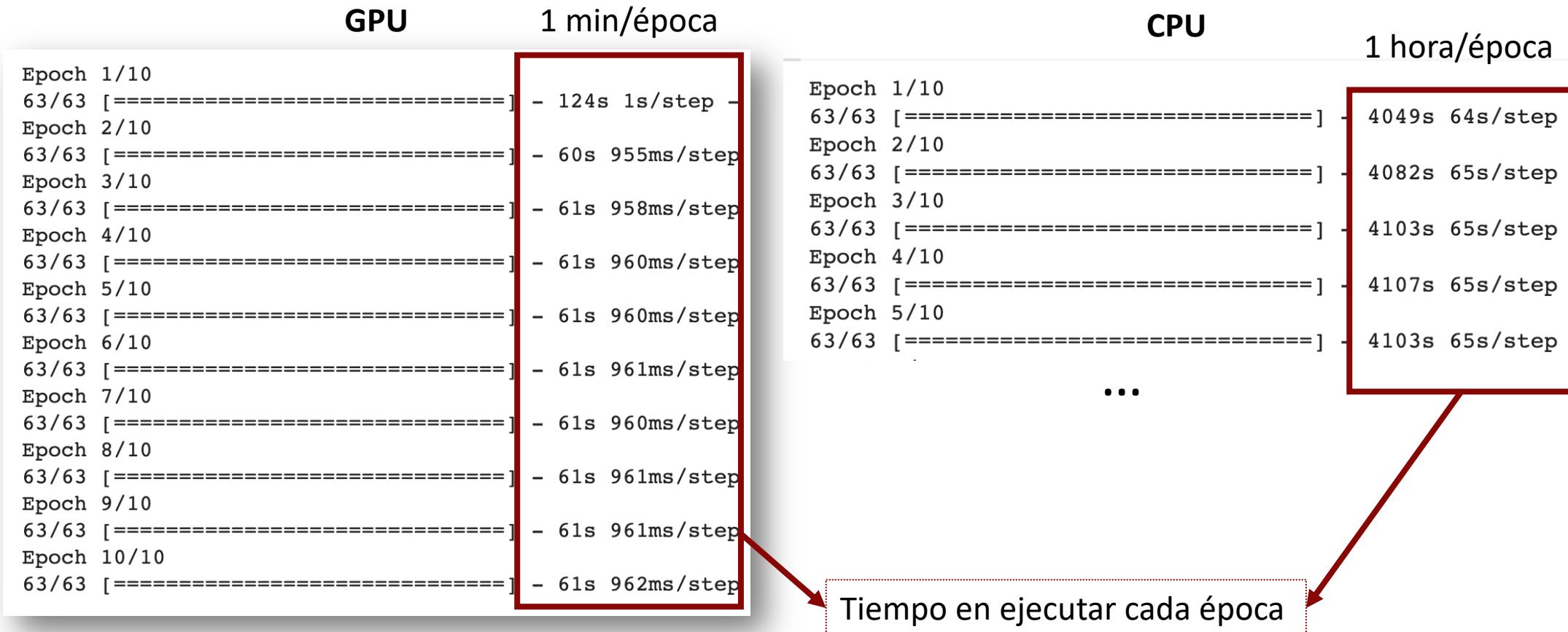
Vuelve a ejecutar todo el código. ¿Cuánto tiempo ha tardado esta vez el entrenamiento?

¿Has notado alguna diferencia?

Entrenamiento

Paso 3: Entrenamiento/validación

Ejemplo entrenamiento (misma configuración) en Colab haciendo uso de GPU y CPU.

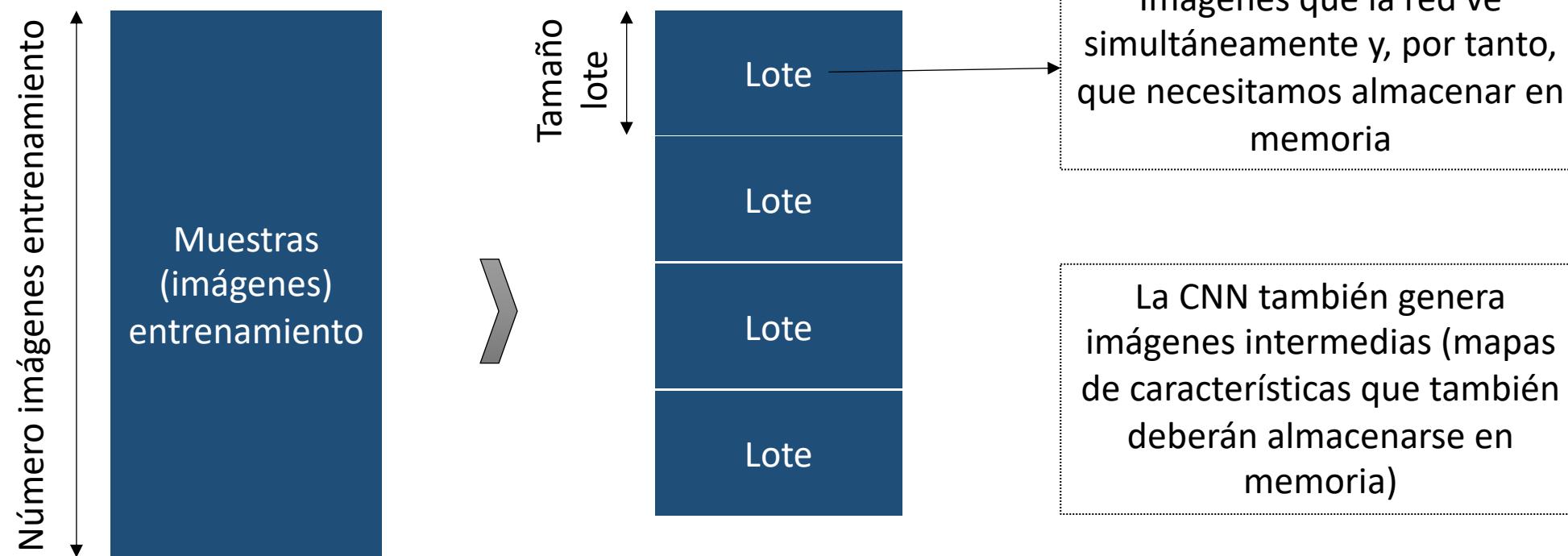


Entrenamiento

Paso 3: Entrenamiento/validación

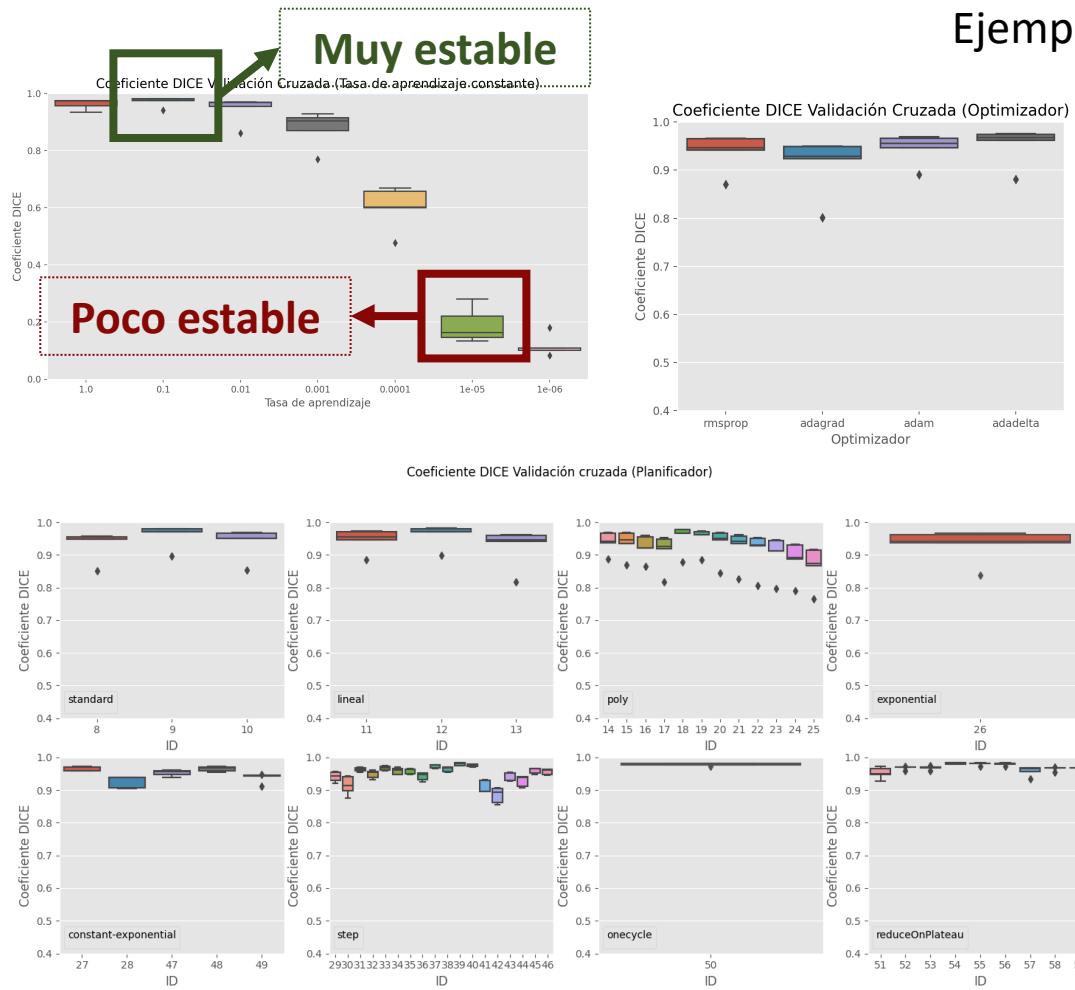
¿Qué me va a limitar la memoria?

- Tamaño de imagen
- Número de imágenes que mi modelo ve simultáneamente

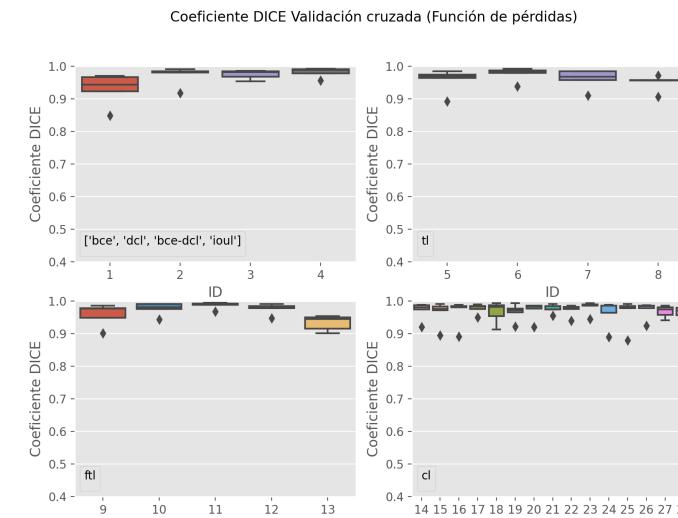


Entrenamiento

Repetiremos los pasos 2 y 3 con diferentes hiperparámetros y seleccionaremos la configuración que mejores resultados en validación nos de.



Ejemplo resultados validación empleando validación cruzada.



Nos quedaremos con la configuración que **mejor rendimiento** nos de y, al mismo tiempo, en caso de haber realizado validación cruzada, **mayor estabilidad** entre iteraciones del proceso de entropía cruzada (no nos vale de nada un modelo que funciona perfecto solo ante un conjunto concreto de casos de validación)

Entrenamiento

En este punto ya tenemos el mejor modelo que hemos podido obtener. De este nos **guardamos el modelo** (arquitectura + pesos ajustados) y será lo que pongamos en producción.

Aunque antes de poner en producción quizás nos pidan documentar los resultados obtenidos (regulatorio, documentación interna de la empresa, publicación científica...), para ello deberemos de **extraer métricas** de como de bueno es nuestro modelo. Para esto emplearemos el conjunto que nos hemos dejado de test.

Las métricas variarán en función del problema que estemos resolviendo (clasificación, regresión, segmentación, detección de objetos, ...).

European Radiology
<https://doi.org/10.1007/s00330-021-07838-5>

IMAGING INFORMATICS AND ARTIFICIAL INTELLIGENCE

Precise whole liver automatic segmentation and quantification of PDFF and R2* on MR images

Ana Jimenez-Pastor¹  · Angel Alberich-Bayarri¹ · Rafael Lopez-Gonzalez¹ · David Marti-Aguado² · Manuela França³ · Rodrigo San Martin Bachmann⁴ · Juan Mazzucco⁵ · Luis Marti-Bonmatí^{6,7}

Check for updates

Title:	WML-VAL-2.0 White matter lesions analysis module description
Document number:	WML-DES-2.0 Document version: 1
Effective date:	25/06/2021

Quibim

WML-VAL-2.0 White matter lesions analysis module validation

Puesta en producción

Puesta en producción

Durante el proceso de entrenamiento la forma de programar ha sido más “estilo libre”, sin embargo, el código que desarrollemos para ponerlo en producción debe de **seguir unas pautas** que faciliten su lectura, compresión, trazabilidad y despliegue.

Para facilitar gran parte de este proceso es una buena práctica emplear **repositorios de GIT**.



Puesta en producción



<https://github.com/>



<https://gitlab.com/>



<https://bitbucket.org/>

Puesta en producción

Preparación repositorio

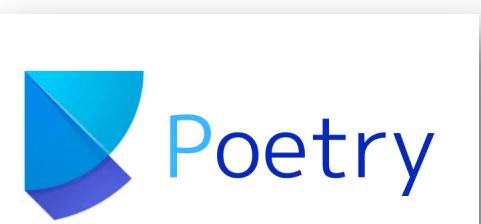
Lo ideal es que cada módulo se ejecute en su propio entorno virtual, de este modo nos evitaremos problemas de compatibilidades entre versiones. Para esto en Quibim empleamos poetry es una herramienta que facilita el empaquetado y manejo de dependencias en Python.

```
[tool.poetry]
name = "white_matter_lesion_segmentation"
version = "0.1.0"
description = ""
authors = ["Ana Jimenez <anajimenez@quibim.com>"]

[tool.poetry.dependencies]
python = "^3.6"
keras = "^2.4"
tensorflow = "^2.3"
scikit-image = "^0.16.0"
opencv-python = "^4.1"

[tool.poetry.dev-dependencies]

[build-system]
requires = ["poetry>=0.12"]
build-backend = "poetry.masonry.api"
```



<https://python-poetry.org/>

Puesta en producción

Preparación repositorio

Ejercicio 2

Vamos a ver cómo podemos instalar un código basado en poetry.

Para ello vamos a descargarnos la carpeta “Ejercicio 2. Proyecto poetry” de Moodle y seguimos los siguientes pasos:

1. Descomprimimos la carpeta y, con la terminal nos establecemos en la carpeta descomprimida (run_poetry)
2. Primero vamos a crear un entorno virtual de Python (para evitar instalarlo a nivel de sistema). Para ello, seleccionamos, o bien la versión que tenemos por defecto a nivel de sistema, o una versión específica de python:
 - *python –m venv ./myenv* (python instalado a nivel de sistema)
 - */usr/local/bin/python3.9 -m venv ./myenv* (versión específica de python)

3. Activamos el entorno virtual:

source myenv/bin/activate

*El entorno está activo cuando aparece éste entre paréntesis

```
[anajimenez@QC116 run_poetry % source myenv/bin/activate  
(myenv) anajimenez@QC116 run_poetry % ]
```

Puesta en producción

Preparación repositorio

Ejercicio 2

4. Instalamos *poetry* mediante el comando: “*pip install poetry*” (esto solo será necesario hacerlo la primera vez que vayamos a hacer uso de este paquete).
5. Instalamos las dependencias requeridas mediante el comando: “*poetry install*”. Con este comando se irán instalando una a una las dependencias especificadas en el archivo *pyproject.toml*
**Cuando termina la instalación, veréis que se genera un archivo *poetry.lock*, este fija la versión específica de cada librería.*
6. Ejecutamos el código mediante el comando: “*poetry run python rotate_image.py*” (con *poetry run* estamos especificando que ejecute el resto del comando en el entorno creado).
** Como resultados de ejecutar el código anterior, se os deberá de haber creado una nueva imagen en el directorio llamada *dogs_rotated.jpeg*.*

El entorno virtual generado se puede usar también desde el propio Visual Studio o PyCharm.

Puesta en producción

Preparación repositorio

Ejercicio 3

Ahora, en lugar de instalar un proyecto ya preparado, vamos a crear uno desde cero.

1. Crea un directorio nuevo.
2. Abre un terminal y cambia el directorio al que acabamos de crear.
3. Ejecuta los puntos 2 a 4 del ejercicio anterior.
4. Ejecuta *poetry init*
5. Acepta los valores por defecto y cuando pregunte sobre instalar dependencia especifica “no”.

```
root@ip-172-31-10-10:~# poetry init
(myenv) anajimenez@QC116 new_folder % poetry init
This command will guide you through creating your pyproject.toml config.

Package name [new_folder]:
Version [0.1.0]:
Description []:
Author [None, n to skip]: Ana Jimenez
License []:
Compatible Python versions [^3.9]:

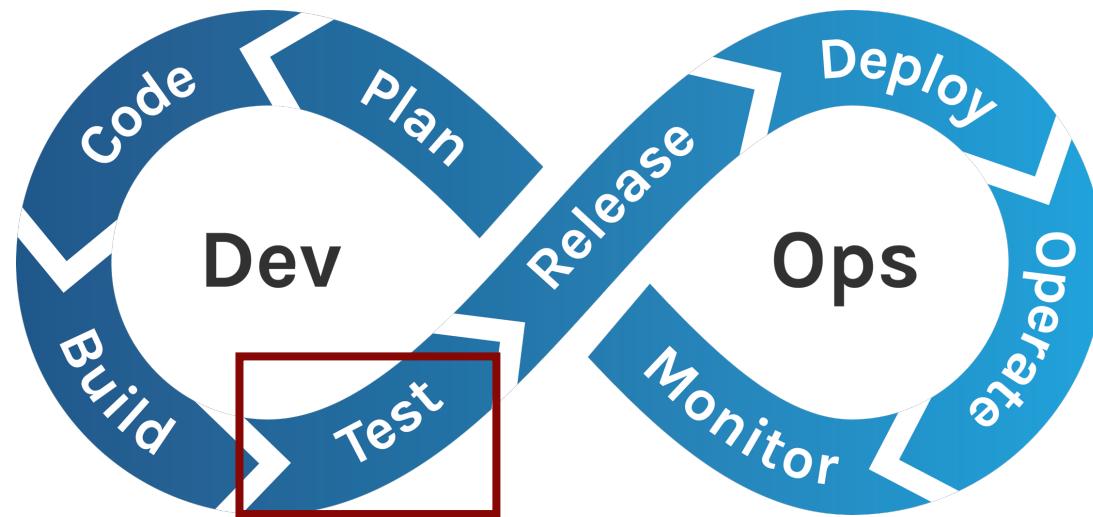
Would you like to define your main dependencies interactively? (yes/no) [yes] no
Would you like to define your development dependencies interactively? (yes/no) [yes] no
Generated file
```

6. Ejecuta *poetry add matplotlib* (en este punto se instala matplotlib en el entorno y se añade la librería al archivo *pyproject.toml*).

Puesta en producción

CI/CD

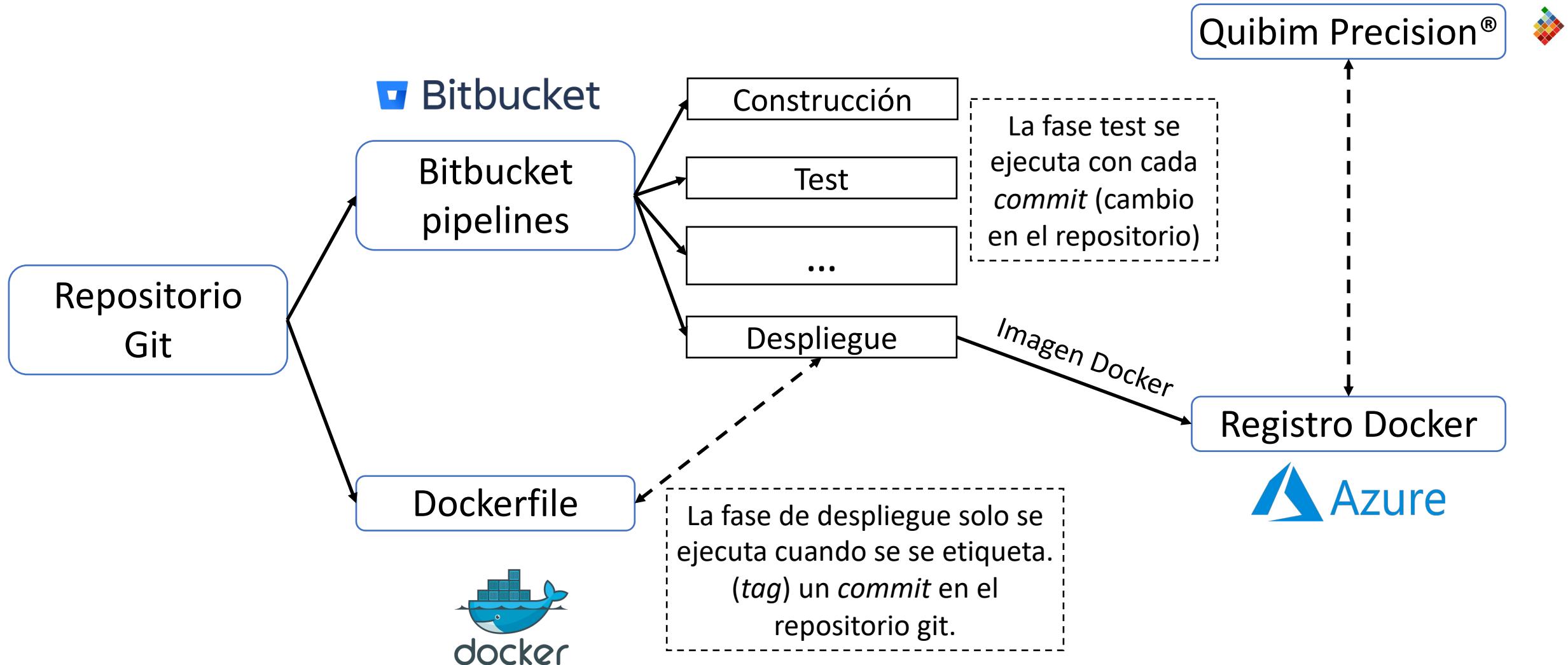
- Integración continua (CI) y despliegue continuo (CD).
- Permite a los equipos de desarrollo desplegar nuevas versiones de código/modelos más frecuentemente de manera más sencilla.
- Permite establecer rutinas de testeo automático de cada cambio introducido en el repositorio.



Testeo unitario, de integración, de rendimiento, etc.

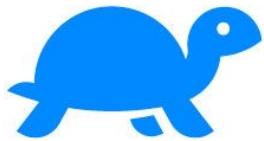
Puesta en producción

CI/CD



Puesta en producción

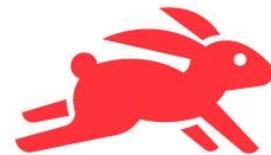
Ejecución



slow

Modelos cuya ejecución no se realiza en tiempo real, sino que requieren de un tiempo largo de procesado.

Arquitectura basada en colas
(Nomad, Kubernetes)



fast

Modelos cuya ejecución se realiza en tiempo real, por tanto, se le da respuesta de manera interactiva al usuario

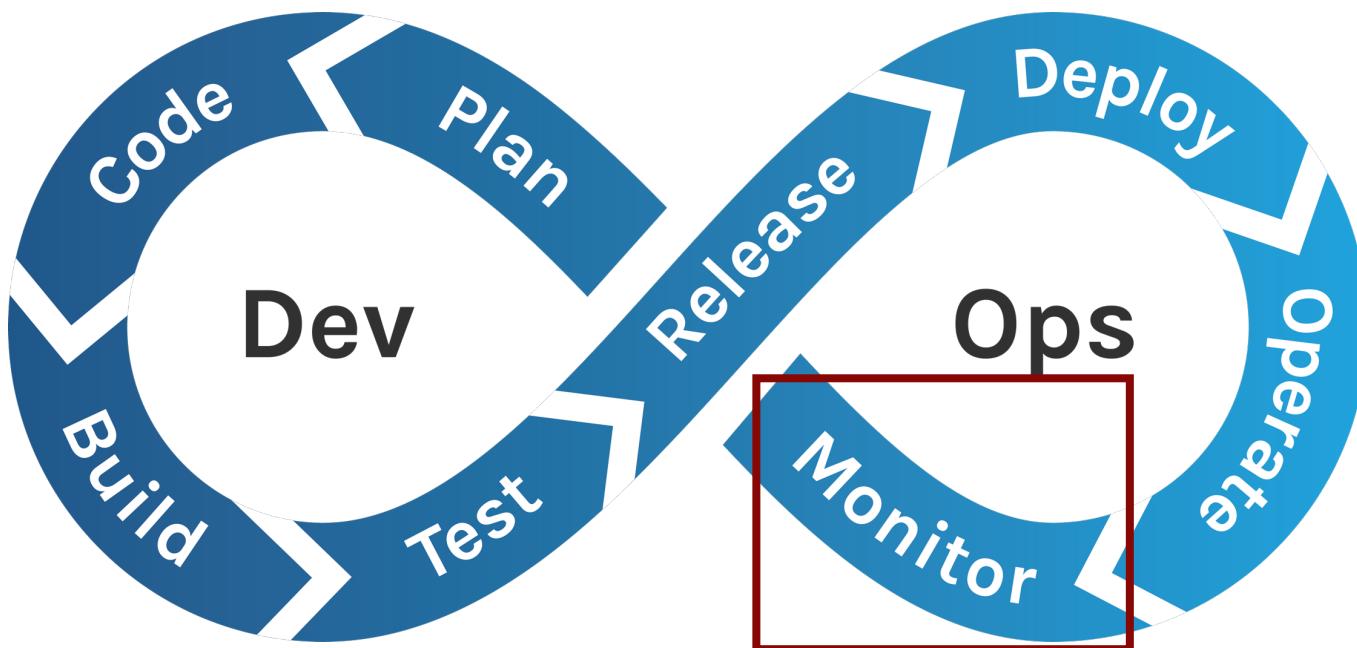
Servicio API REST
(FastAPI)

Tras puesta en producción

Tras puesta en producción

MLOps

Tras la puesta en producción de un modelo, nada nos garantiza que este funcione correctamente durante tiempo indefinido: **deriva de datos, deriva del modelo.**



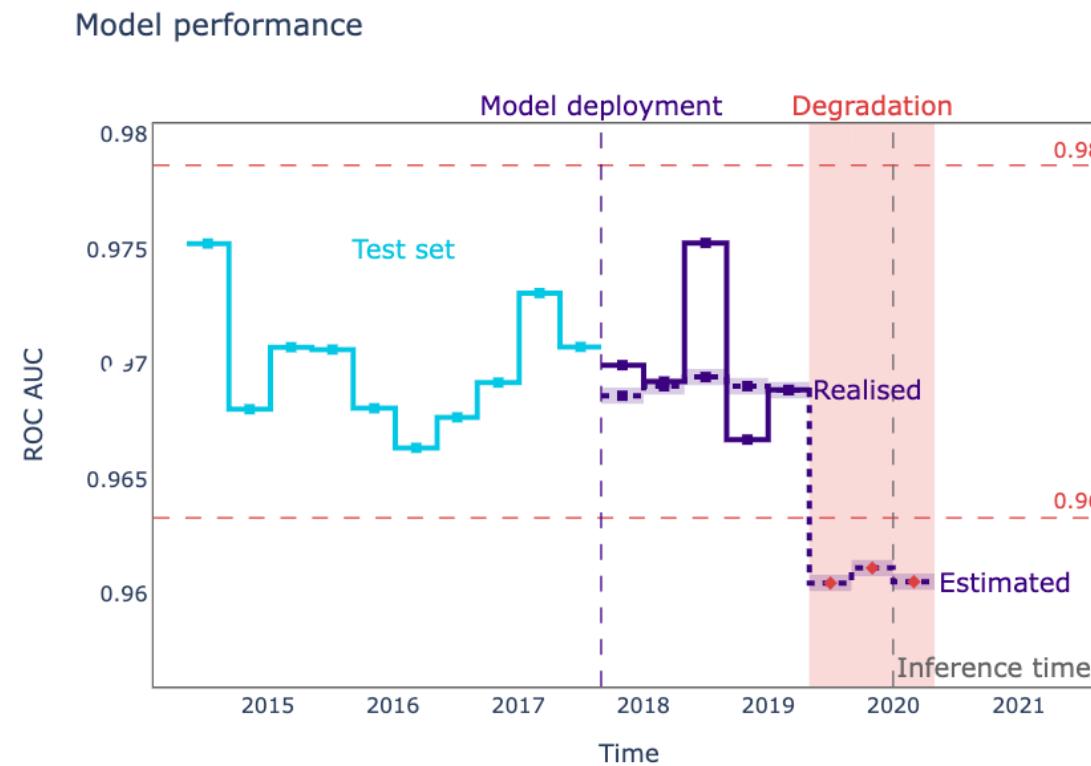
nannyML

EVIDENTLY AI

ALIBI DETECT

Tras puesta en producción

Deriva de los modelos



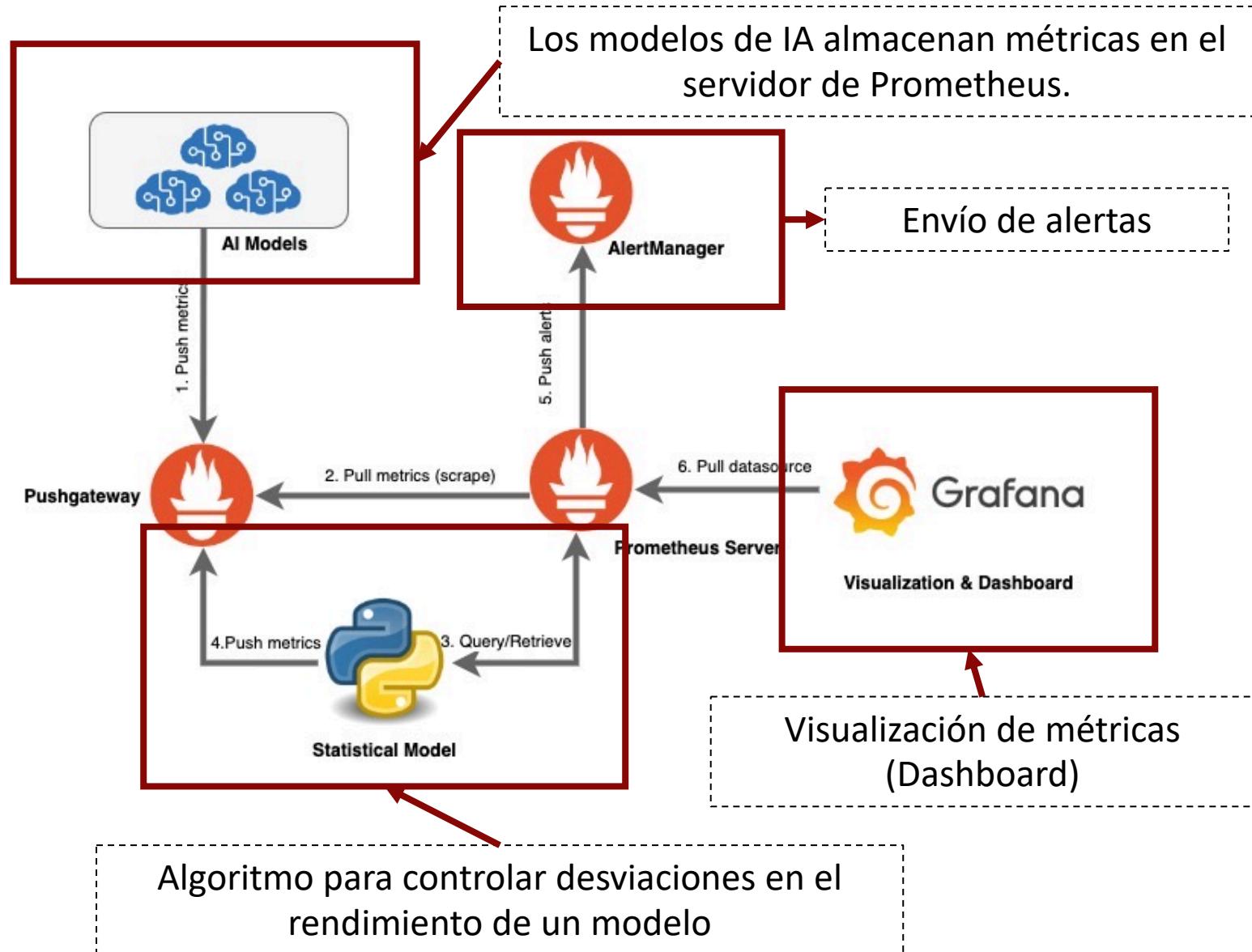
Tras puesta en producción

Monitorización de modelos de IA

Sistema de almacenamiento y visualización de métricas asociadas a un modelo.

Evaluación de las métricas almacenadas para detectar desviaciones del funcionamiento normal.

Envío de alerta en caso de detectar una desviación



Bloque: Redes Neuronales con Imágenes

Ana Jiménez Pastor
anjipas@gmail.com