Deep Learning para Business Analytics

Día 05 · Tema 2 — Transfer learning

Reúsa modelos preentrenados; entrena lo justo.

Eduardo C. Garrido-Merchán

Profesor Propio Adjunto, Métodos Cuantitativos, ICADE

ecgarrido@comillas.edu

Madrid, Día 05

ejemplos del día

Lo que vamos a cubrir hoy.

1. Por qué transfer learning

2. El Hub de Hugging Face

3. Receta exprés Hugging Face

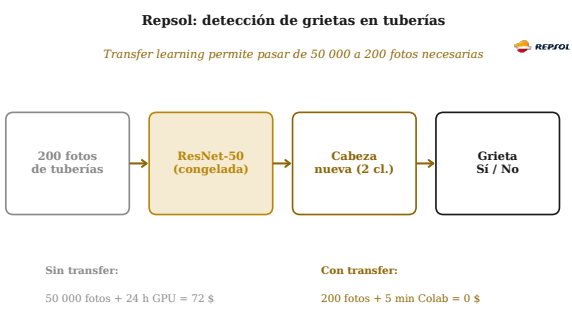
4. Casos reales

Al final del día: eliges entre congelar, fine-tunear parcial o total, sabes qué modelo bajar de 🤖 Hugging Face Hub para tu caso, y entrenas un 📸 OCR de facturas en 30 minutos.

Parte 1 / 4

Por qué transfer learning

Repsol detecta grietas en tuberías con 200 fotos y transfer learning.



El problema. 🏭 Miles de km de tuberías que inspeccionar. Grietas no detectadas = fugas, multas, accidentes. La inspección manual tarda semanas.

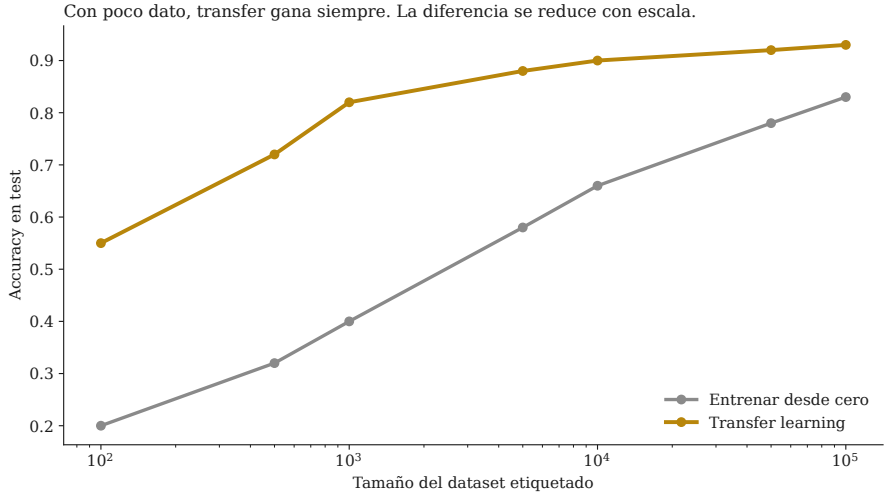
Sin transfer. ⚠️ Entrenar una CNN desde cero requiere ~50 000 fotos etiquetadas y **72 $** en GPU.

Con transfer. 🚀 Backbone ResNet-50 congelado + cabeza de 2 clases. Solo **200 fotos** y 5 minutos en Colab gratis.

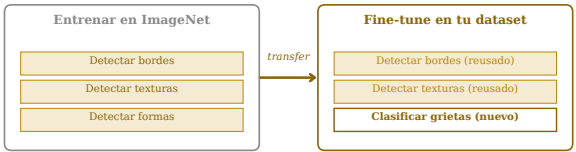
Resultado. Recall de grieta >95 %. Inspección de 1 km de tubería: de 2 horas a 3 minutos.

Idea clave. Transfer learning democratiza la visión: cualquier empresa con 200 fotos puede empezar.

Con poco dato, transfer gana. La diferencia se reduce con escala.



Transfer learning = reusar lo universal, reentrenar lo específico.



Primeras capas: bordes y texturas universales (se reusan).

Últimas capas: rasgos específicos del dominio (se reentrenan).

Analogía: aprender a conducir en Madrid y adaptarse a Londres.

Analogía. 🧠 Aprender a conducir en Madrid y adaptarse a Londres. Los pedales son iguales; lo que cambia es el carril.

Capas universales. Las primeras capas de una CNN detectan bordes, esquinas y texturas. Son iguales para fotos de gatos, tuberías o moda.

Capas específicas. Las últimas capas combinan esas piezas en objetos del dominio: “grieta”, “estante vacío”, “camiseta”.

Consecuencia. Solo hay que reentrenar las capas finales ⇒ **pocos datos y poco tiempo**.

Idea clave. Lo universal se hereda gratis; lo específico se entrena con tus datos.

🚀 Tres niveles de transfer, ordenados por coste creciente.

Tres niveles de transfer learning — escoge según dato y presupuesto.

A. Feature extraction

Congelar backbone,
entrenar solo la cabeza
(Linear final).

100 ej. + GPU 5 min

B. Fine-tuning parcial

Descongelar 1-3 últimas
capas conv + cabeza,
lr bajo.

1 000 ej. + GPU 30 min

C. Fine-tuning total

Descongelar todo,
lr muy bajo,
cosine schedule.

10 000 ej. + GPU 2-4 h

Ejemplo a mano: ¿qué nivel uso si tengo 800 fotos de productos?

Datos. 📸 800 fotos de productos del catálogo de Inditex; 10 categorías; cada foto etiquetada por una persona en 3 minutos ⇒ inversión de ≈ 40 horas-persona ya hecha.

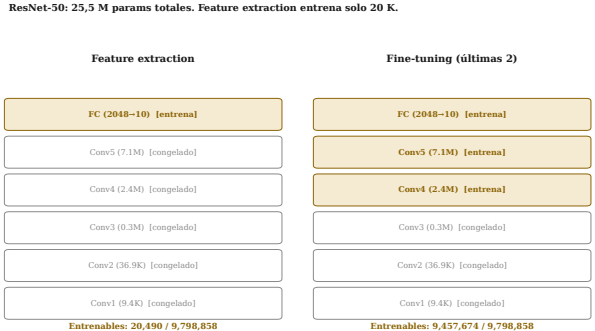
Decisión. Por tamaño (800), nivel A (📚 feature extraction) es lo razonable. Backbone ResNet-50 congelado; cabeza Linear $2048 \to 10$.

Cálculo de parámetros entrenables. params $= 2048 \times 10 + 10 =$ **20 490**. Esto cabe en cualquier portátil sin GPU.

Cálculo del tiempo. 800 imágenes / 32 batch ≈ 25 batches por época. 10 épocas ⇒ 250 batches. A 0,3 s/batch en CPU son 75 segundos por entrenamiento completo.

Idea clave. Tu primera demo de visión cabe en una hora del aula con el equipo que ya tienes.

Feature extraction vs fine-tuning: qué pasa con los pesos.



ResNet-50. 📚 Total: 25,5 M parámetros en 5 bloques conv + 1 cabeza FC.

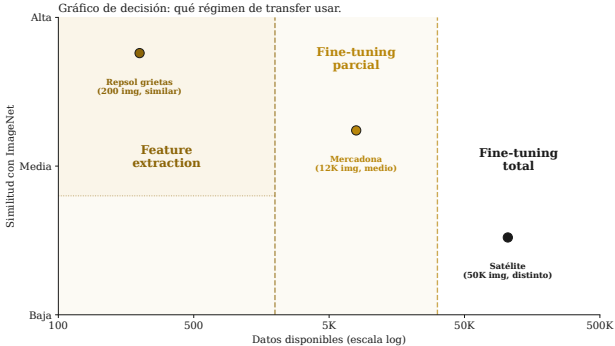
Feature extraction. Solo la cabeza FC entrena: $2048 \times 10 + 10 = \mathbf{20\,490}$ parámetros. El resto está congelado.

Fine-tuning parcial. Se descongelan los últimos 2 bloques conv (Conv4 + Conv5 + FC). Entrenables: ~**22 M** parámetros. Más capacidad, pero necesita más datos y lr bajo (~1e-5).

Regla práctica. <1 000 fotos ⇒ feature extraction. 1 000–10 000 ⇒ parcial. >10 000 ⇒ total.

Idea clave. La diferencia entre regímenes es de *tres órdenes de magnitud* en parámetros entrenables.

Cuándo cada régimen de transfer: el gráfico de decisión.



Eje X: datos disponibles. 🎯 Desde 100 hasta 500 000 imágenes etiquetadas.

Eje Y: similitud con ImageNet. Fotos de objetos cotidianos ≈ alta. Imágenes médicas o satélite ≈ baja.

Reglas de decisión. Pocos datos + alta similitud ⇒ **feature extraction**. Datos medios o similitud media ⇒ **fine-tuning parcial**. Muchos datos + baja similitud ⇒ **fine-tuning total**.

Casos reales. Repsol (200 fotos, alta similitud) → feature extraction. Mercadona (12 K, media) → parcial. Satélite (50 K, baja) → total.

Parte 2 / 4

El Hub de Hugging Face

🔗 El Hub: dónde se publica el trabajo ya hecho.

Hugging Face Hub

475 000 modelos preentrenados, descargables y gratis.

Modelo	Caso de uso	Parámetros	Popularidad
google/vit-base-patch16-224	Visión general	86M	12 K ↓/d
microsoft/resnet-50	Visión clásica	25M	9 K ↓/d
openai/clip-vit-large	Visión + texto	428M	60 K ↓/d
microsoft/dit-base	Documentos OCR	85M	3 K ↓/d
nvidia/segformer-b0	Segmentación	13M	5 K ↓/d

🤖 Concepto → aplicación: qué modelo bajar según el caso.

Modelo	Caso de negocio	Entradas / Salidas	Licencia
ViT-base	🛒 Catálogo retail genérico.	Imagen → vector / clase.	Apache 2.0
ResNet-50	🏭 Control calidad en planta.	Imagen → OK/defecto.	Apache 2.0
CLIP-large	🔍 Búsqueda similitud texto-imagen.	Texto + imagen → score.	MIT
DiT-base	📄 OCR de documentos y facturas.	Imagen documento → campos.	MIT
SegFormer-b0	🌍 Segmentación imágenes satélite.	Imagen → máscara por clase.	Apache 2.0

Idea clave. Antes de elegir, comprueba la licencia: Apache y MIT permiten uso comercial; otras pueden no.

Parte 3 / 4

Receta exprés Hugging Face

🚀 Receta exprés: seis pasos para fine-tunear.

1. Elegir modelo del Hub	`AutoImageProcessor + AutoModelForImageClassification`
2. Sustituir la cabeza	`num_labels = tu_num_clases, ignore_mismatched_sizes=True`
3. Congelar backbone (opcional)	`for p in model.backbone.parameters(): p.requires_grad = False`
4. Configurar TrainingArguments	`learning_rate=2e-5, num_train_epochs=3, weight_decay=0.01`
5. Entrenar con Trainer	`trainer = Trainer(...); trainer.train()`
6. Evaluar y publicar	`trainer.evaluate() ; trainer.push_to_hub()`

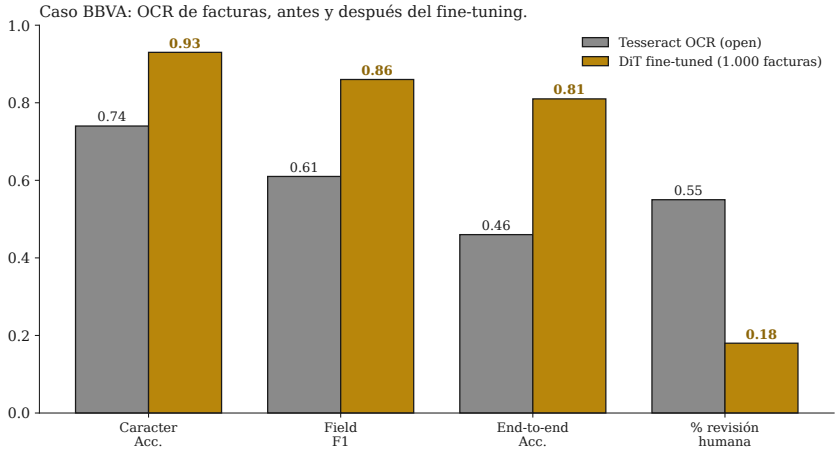
🔧 Código mínimo que aplica la receta.

```
from transformers import (AutoImageProcessor,
     AutoModelForImageClassification,
     TrainingArguments, Trainer)
from datasets import load_dataset

ds = load_dataset("fashion_mnist")
proc = AutoImageProcessor.from_pretrained("microsoft/resnet-50")

def transform(batch):
    return proc(batch[ïmage"], return_tensors="pt")
ds = ds.with_transform(transform)

model = AutoModelForImageClassification.from_pretrained(
    "microsoft/resnet-50", num_labels=10,
    ignore_mismatched_sizes=True)

args = TrainingArguments(output_dir=.ºut", num_train_epochs=3,
    per_device_train_batch_size=32, learning_rate=2e-5,
    weight_decay=0.01, lr_scheduler_type=çosine")
```

Parte 4 / 4

Casos reales

Caso BBVA: OCR de facturas con DiT fine-tuneado.



⭐ Más casos de marca con la misma receta.

Repsol — clasificación de marcas de gasolineras. 📸 Backbone ViT-base, 12 marcas (Repsol, Cepsa, BP…), 5 000 fotos de calle. Accuracy 94 %. Caso de negocio: cumplimiento de identidad corporativa en franquicias.

Mercadona — control de calidad en frutería. 🛒 Backbone EfficientNet-B0 ligera, 4 clases (OK, dañado, podrido, dudoso), 12 000 fotos. Recall del defecto 96 %. Caso: reducción de devoluciones en tienda.

Iberdrola — segmentación de imágenes satélite. 🌍 Backbone SegFormer-b0, 6 clases de uso de suelo, 1 800 tiles Sentinel-2 anotados internamente. mIoU 0,78. Caso: estimación de potencial fotovoltaico en suelo rústico.

Idea clave. La misma receta cambia de sector porque cambian solo dataset, cabeza y licencia. Lo demás es repetir.

🎓 Lo que el alumno se lleva a casa de este día.

Concepto 1. 📚 Tres niveles de transfer (feature extraction, fine-tuning parcial, fine-tuning total) corresponden a tres rangos de dataset y de coste.

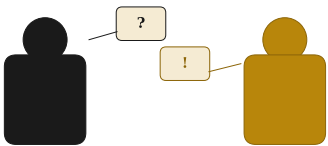
Concepto 2. 🔗 El Hub de Hugging Face concentra el trabajo de la comunidad; antes de entrenar, busca.

Concepto 3. 🚀 La receta exprés de seis pasos (Hub → cabeza → congelar → args → Trainer → evaluar) sirve para los cinco casos de negocio del Tema 2.

Para la práctica. 🔧 Fine-tunear un ViT o ResNet sobre EuroSAT, Fashion-MNIST o tu dataset propio.

Próximo día. 🧠 Día 06 (Tema 2.3): RNN, LSTM y GRU para forecasting.

Pausa para debatir: ¿Ética de los modelos del Hub?



La pregunta. 🤖 Un modelo de Hugging Face fue entrenado con datos posiblemente protegidos. ¿Lo usas en tu producto?

Posición A. ✅ Sí, si la licencia lo permite: el riesgo legal lo asume quien entrenó.

Posición B. 🛡️ No, sin auditoría de procedencia: tu producto hereda el problema.

Reglas. 5 minutos, dos turnos de palabra por bando, móviles boca abajo. Yo modero, no participo.

Hasta el Día 06.

Forecasting de demanda con redes recurrentes.

Eduardo C. Garrido-Merchán

ecgarrido@comillas.edu