Clase 112 — Transfer learning, unsupervised pretraining

Parte: 2 — Deep Learning · Fuente: Géron, cap. 11 § Reusing Pretrained Layers. Duración estimada: 70 min.

Clase 112 — Transfer learning, unsupervised pretraining

> *Parte: 2 — Deep Learning · Fuente: Géron, cap. 11 § Reusing Pretrained Layers. Duración estimada: 70 min.*

Objetivo

Aplicar transfer learning — el patrón dominante en producción cuando hay pocos datos: tomar un modelo preentrenado (ImageNet para visión, BERT/GPT para texto), reemplazar la cabeza, congelar las capas base, fine-tunear la cabeza, y opcionalmente descongelar gradualmente para una segunda fase con LR muy bajo. Conocer unsupervised pretraining como hermano histórico (autoencoders, contrastive learning).

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Cargar un modelo preentrenado: keras.applications.MobileNetV3Small(weights='imagenet', include_top=False).
- Congelar capas: base.trainable = False.
- Construir un modelo nuevo con la base + un head custom (GlobalAveragePooling2D + Dense).
- Fine-tunear en dos etapas: (a) solo head con LR normal, (b) toda la red con LR 10× más bajo.
- Reconocer cuándo NO usar transfer (dataset muy distinto al de origen).

Temas

- Por qué funciona: las capas tempranas aprenden features generales (bordes, texturas); las tardías son task-specific.
- Pipeline estándar: load → freeze → new head → fit → unfreeze → fit con LR bajo.
- LR diferencial: capas tempranas más bajo (1e-5), capas tardías más alto (1e-3).
- Unsupervised pretraining: autoencoders (clase 130), contrastive (SimCLR, MoCo, CLIP).
- Self-supervised: cómo BERT/GPT se pre-entrenan sin labels (masked LM / autoregressive).

Definiciones y características

- Transfer learning: reusar pesos preentrenados en una tarea como punto de partida para otra.
- Feature extraction: usar el modelo base como extractor fijo (trainable=False) y entrenar solo la cabeza.
- Fine-tuning: descongelar (parte o todo) el modelo base y continuar entrenando con LR bajo.
- Catastrophic forgetting: si descongelás y usás LR alto, el modelo "olvida" lo aprendido. Por eso LR bajo en fine-tuning.
- Frozen / Trainable: layer.trainable = False excluye los pesos del backprop.
- Self-supervised: pre-entrenar con un objetivo derivado de los propios datos (predecir palabra masked, contrastar augmentaciones).

Dataset / recursos

- Visión: dataset chico de 2-5 clases (perros/gatos, flores). tf.keras.utils.image_dataset_from_directory.

- Modelo base: MobileNetV3Small o EfficientNetB0 (más liviano que ResNet50).
- Librerías: tensorflow, keras, keras.applications.

Ejercicios

1. Carga: base = MobileNetV3Small(weights='imagenet', include_top=False, input_shape=(224,224,3)); base.trainable = False.
2. Modelo completo: model = Sequential([base, GlobalAveragePooling2D(), Dropout(0.2), Dense(num_classes, activation='softmax')]).
3. Etapa 1: compilar con Adam(1e-3) y entrenar 10 épocas. Reportar accuracy.
4. Etapa 2: base.trainable = True y recompilar con Adam(1e-5). Entrenar 10 épocas más. Verificar mejora.
5. Sin transfer: entrenar la misma arquitectura desde cero (weights=None). Comparar cuántas épocas necesita para igualar accuracy de transfer (típicamente: nunca lo iguala con dataset chico).

Homework verificable

Clasificación de un dataset propio de imágenes (≥ 3 clases, ≥ 100 imágenes por clase):

1. Pipeline con image_dataset_from_directory + augmentation (RandomFlip, RandomRotation).
2. Transfer learning con EfficientNetB0.
3. Reportar accuracy de las dos etapas (frozen y unfreeze).
4. Comparar contra entrenar desde cero.

Criterio de aceptación: la etapa 2 (unfreeze + LR bajo) debe mejorar la etapa 1 por ≥ 2 pp; el modelo desde cero queda al menos 10 pp por debajo.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
Olvido base.trainable = False y la etapa 1	Está entrenando millones de params. Fix: c
Recompilar entre etapas pero trainable cam	El cambio de trainable requiere recompilar
Imágenes en [0, 255] cuando el modelo espe	Pre-procesar con keras.applications.<model
BN en training=True se actualiza en fine-t	Sin cuidado, las moving averages se siguen
Catastrophic forgetting	LR demasiado alto al descongelar. Fix: 10-

Preguntas frecuentes

¿Cuántas imágenes mínimas para transfer learning?

Para fine-tuning: 100-500 por clase. Para feature extraction (solo head): incluso 20-50 por clase puede funcionar.

¿Qué modelo base elijo?

Empezá con MobileNetV3Small o EfficientNetB0 (chicos, rápidos). Si necesitás más capacidad y tenés GPU, EfficientNetB3-B5 o ConvNeXt.

¿Cuántas capas descongelo?

Empezá con todo descongelado + LR bajo. Si overfittea, congelar las primeras N capas (las más genéricas).

¿Transfer funciona para datos médicos / satelitales?

Sí, pero menos: las features de ImageNet son menos relevantes. Aún así, suele superar entrenar desde cero. Para datasets muy específicos, considerar pretraining con MAE o SimCLR sobre los datos propios.

¿Y para texto?

Misma idea: cargar bert-base o distilbert desde Hugging Face (clase 127), agregar head, fine-tunear. Estándar industrial.

Referencias

- Géron, cap. 11 — Reusing Pretrained Layers.
- Keras Applications — catálogo de modelos preentrenados.
- Pan & Yang (2010), A Survey on Transfer Learning, IEEE TKDE.
- Chen et al. (2020), A Simple Framework for Contrastive Learning of Visual Representations (SimCLR), ICML.
- He et al. (2022), Masked Autoencoders Are Scalable Vision Learners (MAE), CVPR.

Siguiente clase

Clase 113 — Optimizadores: Momentum, Nesterov, AdaGrad, RMSProp, Adam, AdamW (+ Lion, Sophia)

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb