Redes Convolucionales y Modelos Preentrenados con Optimización

2. ¿Qué son las CNNs?

Son un tipo de red neuronal especializada en procesar datos con estructura de grilla, como las imágenes.

Imitan el comportamiento de la corteza visual.

Uso: clasificación de imágenes, detección de objetos, segmentación.

3. Arquitectura de una CNN

Convolución: extrae patrones locales.

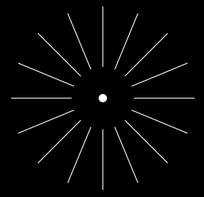
ReLU: activación no lineal.

Pooling: reduce dimensionalidad (ej: MaxPooling).

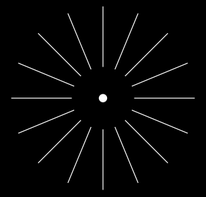
Flatten: convierte datos 2D en vector.

Densa (Fully Connected): toma decisiones.

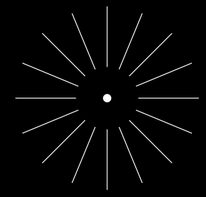
4. ¿Qué son los modelos preentrenados?



Son modelos entrenados previamente sobre conjuntos de datos masivos y variados (como ImageNet).



Capturan representaciones generales útiles para tareas de visión.

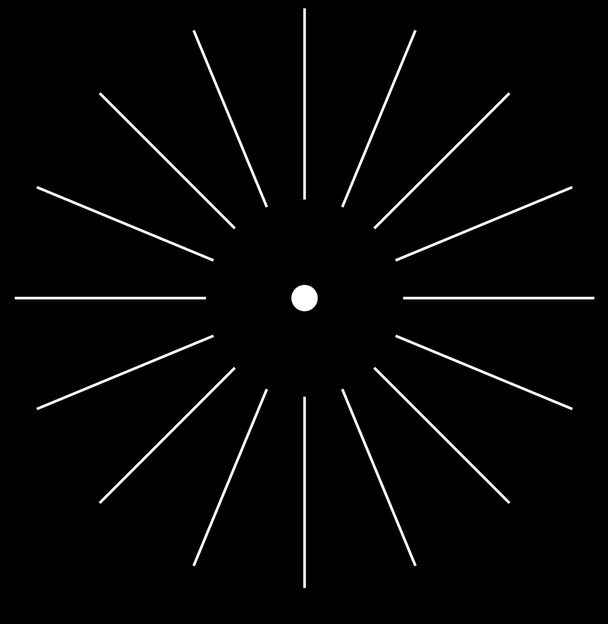


Se adaptan a nuevos problemas específicos mediante técnicas como transfer learning.

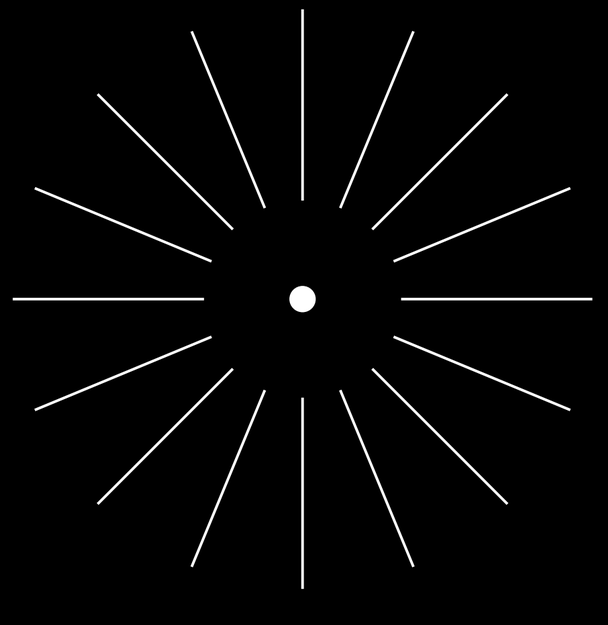
5. ¿Qué es Transfer Learning?

- Técnica que consiste en reutilizar un modelo entrenado en una tarea (como clasificación en ImageNet) como punto de partida para una tarea distinta pero relacionada.
- Proceso:
  a. Se carga un modelo preentrenado.
  b. Se congelan sus capas.
  c. Se añade una nueva cabeza de clasificación.
  d. Se entrena con los datos del nuevo problema.
- Ejemplo típico: adaptar una red entrenada para 1000 clases de ImageNet a una clasificación personalizada de 5 clases (por ejemplo, posturas de yoga).

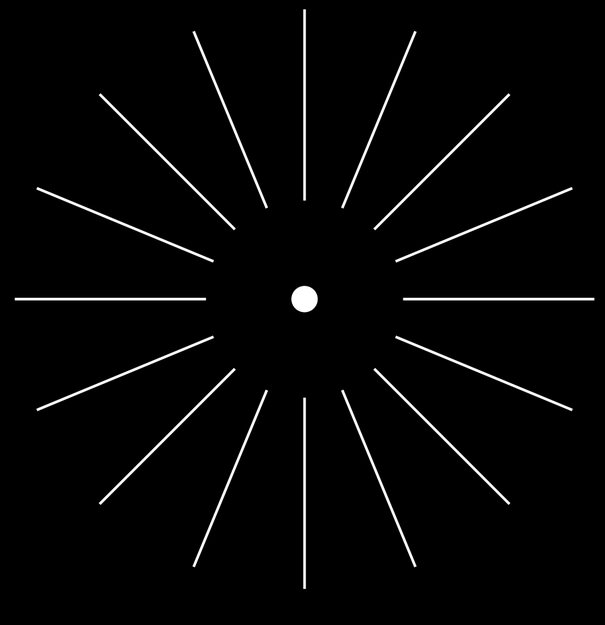
6. Ejemplos de modelos preentrenados y diferencias



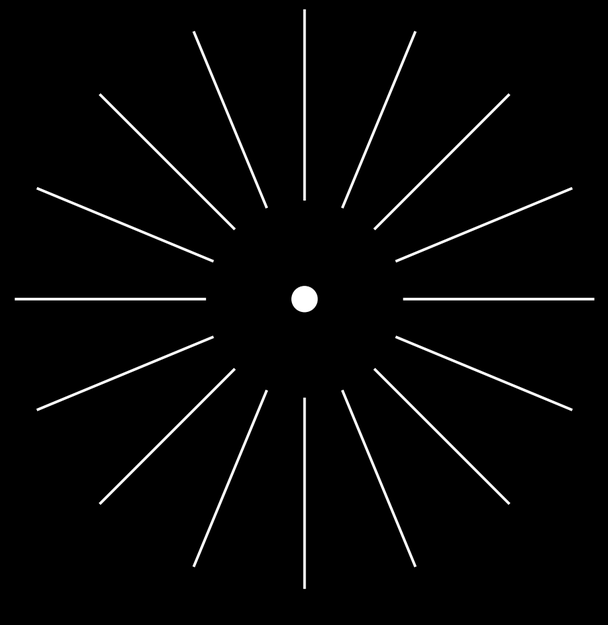
VGG16: arquitectura simple y profunda, usa muchas capas convolucionales seguidas. Muy pesado en parámetros.



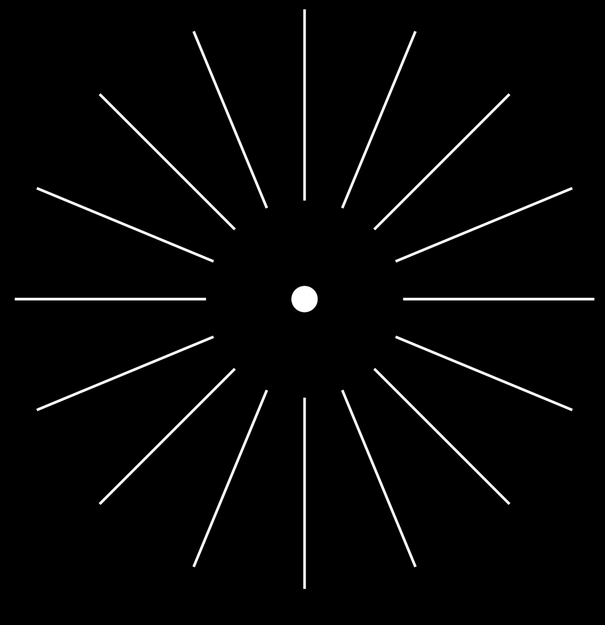
ResNet50V2: introduce conexiones residuales que facilitan el entrenamiento de redes profundas.



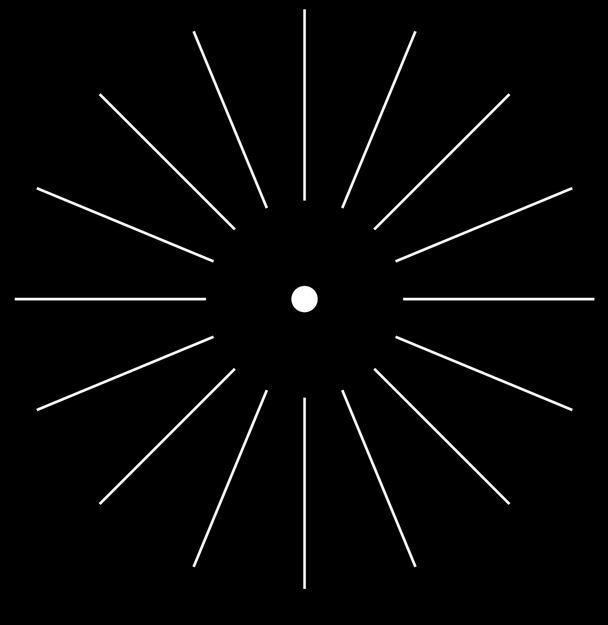
InceptionV3: arquitectura más eficiente, con bloques "Inception" que combinan filtros de diferentes tamaños.



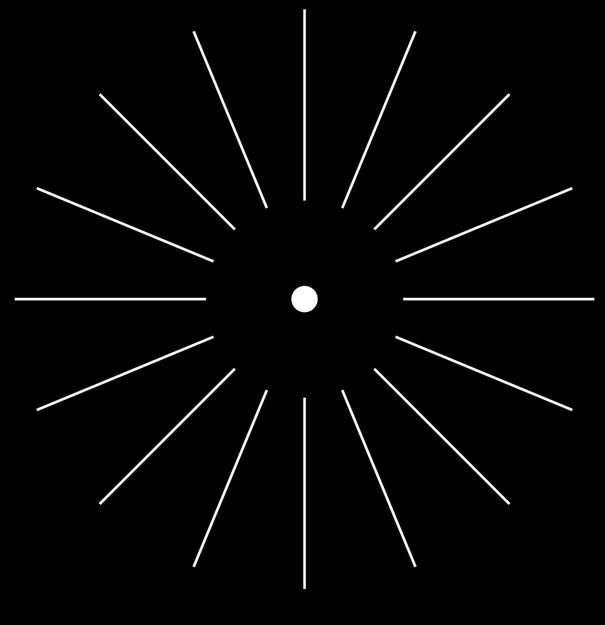
ResNet50: versión clásica con conexiones residuales, excelente precisión y buena generalización.



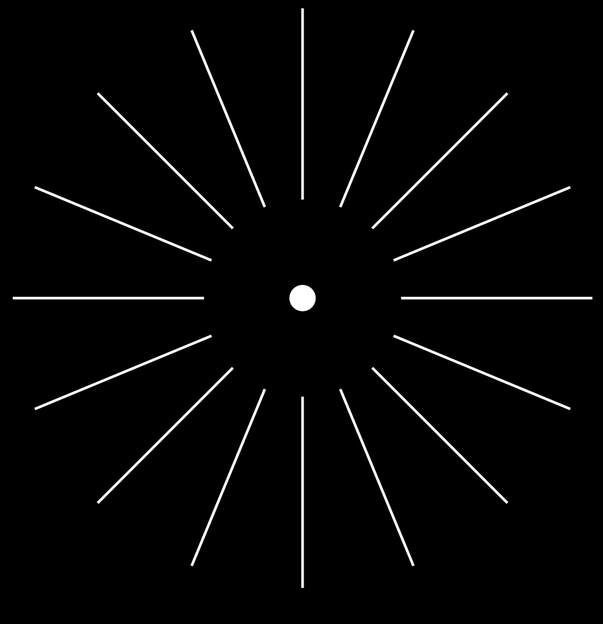
EfficientNet (v1 y v2): optimiza precisión y eficiencia con escalado compuesto de profundidad, anchura y resolución.



MobileNetV3: red ligera, ideal para dispositivos móviles sin sacrificar demasiado rendimiento.



ConvNeXt (2022): combina lo mejor de CNNs y transformers, logrando gran rendimiento en benchmarks recientes.



7. Ventajas de los modelos preentrenados

Ahorro de tiempo y recursos computacionales.

Buen punto de partida para nuevos proyectos.

Mejoran la precisión con pocos datos.

Reutilización del conocimiento aprendido.

Permiten lograr buenos resultados sin necesidad de entrenar desde cero.

8. Uso básico de un modelo preentrenado (VGG16)

```
from tensorflow.keras.applications import VGG16
from tensorflow.keras import layers, models
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
for layer in base_model.layers:
layer.trainable = False
x = layers.Flatten()(base_model.output)
x = layers.Dense(512, activation='relu')(x)
x = layers.Dropout(0.5)(x)
x = layers.Dense(5, activation='softmax')(x)
model = models.Model(inputs=base_model.input, outputs=x)
```

8b. Uso de InceptionV3 personalizado

```
from tensorflow.keras.applications.inception_v3 import InceptionV3
from tensorflow.keras import layers, models
base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
for layer in base_model.layers:
layer.trainable = False  # congelamos las capas del modelo base
x = layers.GlobalAveragePooling2D()(base_model.output)
x = layers.Dense(512, activation='relu')(x)
x = layers.Dropout(0.5)(x)
x = layers.Dense(5, activation='softmax')(x)
model = models.Model(inputs=base_model.input, outputs=x)
model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])
```

9. Optimización del modelo

Ajuste de hiperparámetros: learning rate, tipo de optimizador.

Augmentación de datos: rotación, flips, zoom para generalización.

Callbacks: EarlyStopping, ReduceLROnPlateau, ModelCheckpoint.

Regularización: Dropout, L2 para evitar sobreajuste.

10. Conclusión y próximos pasos

Las CNNs son la base de muchos modelos de visión artificial.

Combinar buena arquitectura + optimización es clave.

Los modelos preentrenados son un atajo poderoso.

Próxima parada: probar diferentes modelos y comparar resultados.