Redes Neuronales Convolucionales (CNN)

UNMSM – FISI – Unidad de Post Grado

Profesor: Juan Gamarra Moreno

Introducción

Limitaciones de las redes neuronales estándar

- Redes neuronales densas (fully connected) no son ideales para imágenes porque:
  - Requieren demasiados parámetros al procesar datos con alta dimensionalidad (por ejemplo, imágenes grandes).
  - No tienen en cuenta la estructura espacial (relaciones locales entre píxeles).
- Ejemplo: Una imagen de 28×28 píxeles, tiene 784 entradas, pero una imagen de alta resolución, como 1080×720, tendría más de 2 millones de entradas.

CNN e imágenes

- CNNs explotan la estructura espacial de las imágenes, procesando grupos locales de píxeles en lugar de procesar cada píxel individualmente.
- Reducción de dimensionalidad mediante convoluciones y pooling:
  - Preservan las características importantes (bordes, texturas).
  - Disminuyen los requisitos de almacenamiento y cómputo.

Estructura de una CNN

Estructura de una CNN

Capas principales:

- Convolucionales:
  - Aplicación de filtros/kernels.
  - Detección de características locales (bordes, texturas).
- Pooling:
  - Max-pooling, average-pooling.
  - Reducción de dimensionalidad y extracción de características relevantes.
- Fully Connected:
  - Conexión completa para realizar la clasificación final.

Función de pérdida y optimización (categorical crossentropy, Adam optimizer).

Capas principales

- **Capas Convolucionales:**
  - Aplican filtros (kernels) a regiones locales de la imagen para detectar patrones (bordes, esquinas, texturas).
  - Por ejemplo, un filtro 3×3 pasa sobre una imagen para detectar bordes horizontales o verticales.
  - La salida es un **mapa de características** que representa las características detectadas.

  Fórmula de la operación convolucional:

  - $Output[i,j] = \sum_{m,n} Kernel[m,n] \cdot Input[i+m, j+n]$

  Ejemplo

  - Los filtros son aprendidos durante el entrenamiento.

Capas principales

- **Capas de Pooling:**
  - Reducen la dimensionalidad de los mapas de características mientras preservan información relevante.
  - **Tipos comunes:**
    - **Max-Pooling:** Selecciona el valor máximo en una región.
    - **Average-Pooling:** Calcula el promedio de los valores en una región.
  - Ayuda a la invariancia a la traslación y reduce el riesgo de sobreajuste.

  Ejemplo
    - Por ejemplo, un mapa de 4×4 es reducido a 2×2 mediante Max-Pooling.

Capas principales

- **Fully Connected (FC):**
  - Conecta todas las características extraídas de las capas anteriores a las salidas.
  - La salida final es una probabilidad asignada a cada clase.

  Función de activación de salida:
  - Para clasificación multiclase, se utiliza **Softmax** para asignar probabilidades a las clases.

Función de Pérdida y Optimización

- **Categorical Crossentropy:**
  - Mide la diferencia entre las distribuciones de salida del modelo y las etiquetas reales.

$$Loss = -\sum_{i} y_i \cdot \log(\hat{y}_i)$$

  - Donde y_i es la etiqueta real y $\hat{y}_i$ la probabilidad predicha.
- **Adam Optimizer:**
  - Combina las ventajas de **momentum** y **RMSProp** para ajustar los pesos de manera eficiente.

Uso de las CNN

Casos de Uso de las CNN

- **Clasificación de imágenes:**
  - Ejemplo: Clasificación de CIFAR-10 o MNIST para identificar objetos o dígitos.
- **Segmentación de objetos:**
  - Identificación precisa de objetos dentro de una imagen (ejemplo: contornos de personas).
- **Detección de anomalías:**
  - Usos en seguridad industrial, como detectar defectos en productos.

Resumen

1. CNNs resuelven problemas espaciales mejor que redes tradicionales.
2. Procesan imágenes con una combinación de capas convolucionales, pooling y fully connected.
3. Aplicaciones prácticas en casi todas las áreas relacionadas con visión por computadora.

Este enfoque estructurado y visual asegura que los estudiantes comprendan los conceptos clave y las aplicaciones de las CNN.

Redes Pre-entrenadas

¿Qué son las Redes Pre-entrenadas?

- Las redes pre-entrenadas son modelos de redes neuronales que han sido entrenados previamente en un conjunto de datos grande y diverso, como **ImageNet**, para resolver una tarea general, como la clasificación de imágenes. Luego, estos modelos pueden ser reutilizados y adaptados a tareas específicas mediante técnicas como **transfer learning**.

Características Principales

1. **Entrenadas en Grandes Conjuntos de Datos:**
   - Los modelos pre-entrenados suelen ser entrenados en conjuntos como ImageNet (con más de 1 millón de imágenes clasificadas en 1,000 categorías).
2. **Reutilizables:**
   - En lugar de entrenar un modelo desde cero, puedes usar un modelo pre-entrenado y ajustarlo a tu tarea con menos recursos computacionales.
3. **Arquitecturas Populares:**
   - Ejemplos incluyen **VGG16**, **ResNet**, **Inception**, **EfficientNet**, entre otros.
4. **Ahorro de Recursos:**
   - Entrenar un modelo desde cero requiere tiempo, poder computacional y grandes volúmenes de datos. Las redes pre-entrenadas eliminan gran parte de este costo.

Ventajas de las Redes Pre-entrenadas

1. **Transferencia de Conocimiento:**
   - Las características aprendidas en tareas generales (como detectar bordes o texturas) pueden ser útiles en tareas específicas.
2. **Reducción del Tiempo de Entrenamiento:**
   - Solo necesitas ajustar las últimas capas del modelo o aplicar un ajuste fino (**fine-tuning**) al modelo completo.
3. **Mejor Rendimiento con Datos Limitados:**
   - Al aprovechar características generales pre-aprendidas, es posible lograr un buen desempeño incluso con conjuntos de datos pequeños.

Uso Típico de Redes Pre-entrenadas

1. **Congelar las Capas Inferiores:**
   - Las primeras capas del modelo (que capturan características genéricas como bordes y texturas) se mantienen fijas.
   - Solo se entrenan las capas superiores que están más adaptadas a la tarea específica.
2. **Ajuste Fino (Fine-tuning):**
   - Se reentrenan las capas superiores y opcionalmente algunas capas inferiores con un conjunto de datos específico.

Casos de Uso

- **Clasificación de Imágenes:** Personalizar un modelo pre-entrenado para identificar defectos en productos específicos.
- **Detección de Objetos:** Usar redes como YOLO o Faster R-CNN.
- **Segmentación Semántica:** Adaptar redes como U-Net para tareas médicas.

Las redes pre-entrenadas son una herramienta poderosa que permite a investigadores y desarrolladores resolver problemas complejos con recursos computacionales y datos limitados.

Transfer Learning y Fine-Tuning

Transfer Learning

Definición:

Es el proceso de reutilizar un modelo pre-entrenado (como ResNet50, VGG16, etc.) en un problema nuevo, generalmente con un conjunto de datos diferente. En Transfer Learning, las capas del modelo base suelen mantenerse **fijas** y no se entrenan.

Transfer Learning

Características Clave:

- **Congelar capas pre-entrenadas:** Las capas del modelo base se utilizan "tal cual", sin modificar los pesos pre-entrenados.
- **Añadir capas personalizadas:** Se agregan nuevas capas en la parte superior del modelo para adaptarlo a la nueva tarea (por ejemplo, clasificar 10 clases en lugar de las 1000 originales de ImageNet).
- **Menor costo computacional:** Como solo se entrenan las capas añadidas, requiere menos recursos y tiempo.
- **Ideal para datos limitados:** Funciona bien cuando el conjunto de datos específico es pequeño.

Transfer Learning

Ejemplo:

- Usar ResNet50 pre-entrenada en ImageNet para clasificar el conjunto de datos CIFAR-10, congelando todas las capas del modelo base y entrenando solo las nuevas capas superiores.

```
# Congelar capas del modelo base
base_model.trainable = False
```

Fine-Tuning

Definición:

Es un paso adicional que afina (ajusta) los pesos de las capas del modelo base junto con las capas personalizadas. Esto permite al modelo adaptarse mejor al problema específico.

Fine-Tuning

Características Clave:

- **Descongelar capas específicas:** Se descongelan algunas capas superiores o todas las capas del modelo base para reentrenarlas.
- **Reentrenar con datos nuevos:** Se utiliza un conjunto de datos específico para ajustar los pesos del modelo, con una tasa de aprendizaje baja para evitar sobreajuste.
- **Mayor costo computacional:** Reentrenar más capas requiere más recursos.
- **Mejora del rendimiento:** Es especialmente útil cuando el conjunto de datos objetivo es grande y tiene características diferentes al conjunto usado en el pre-entrenamiento.

Fine-Tuning

Ejemplo:

- Descongelar las últimas capas de ResNet50 y ajustar los pesos usando el conjunto de datos CIFAR-10.

```
# Descongelar capas superiores del modelo base
base_model.trainable = True

# Reentrenar todo el modelo con una tasa de aprendizaje baja
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-5),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Diferencias Clave

Aspecto	Transfer Learning	Fine-Tuning
Objetivo	Reutilizar las características aprendidas en el modelo base.	Ajustar pesos del modelo base para mejorar el rendimiento.
Capas del modelo base	Se congelan y no se entrenan.	Se descongelan y se reentrenan (parcial o totalmente).
Tasa de aprendizaje	Alta (para entrenar las capas nuevas).	Baja (para evitar alterar demasiado los pesos pre-entrenados).
Costo computacional	Bajo.	Alto.
Cuándo usarlo	Cuando el conjunto de datos es pequeño y similar al pre-entrenado.	Cuando el conjunto de datos es grande o diferente al pre-entrenado.