PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
IIC-3697 APRENDIZAJE PROFUNDO

TAREA 2: EXPERIMENTOS CON REDES NEURONALES CONVOLUCIONALES (CNNS)

Fecha de Entrega: 17 de Mayo

Objetivo

Estimad@s, en esta actividad tendrán la oportunidad de continuar experimentando con CNNs. En particular, pondrán en práctica lo que conversamos sobre refinamiento de modelos (finetuning), funciones de pérdida y funciones de activación. Adicionalmente, a diferencia de la tarea 1 en que utilizaron Keras, en esta oportunidad utilizarán PyTorch (`https://pytorch.org/`).

1 Finetuning y funciones de activación (50%)

En esta parte de la tarea utilizarán el set de datos *Birds*, este consiste de:

- 11K imágenes totales.
- 200 clases.
- Set de datos esta particionado en train, validación y test.
- `https://drive.google.com/file/d/1rBezXbc18qILDXJS1YUjM5M4MeCDjU0l/view?usp=sharing`.

Actividad 1

Entrenar un modelo *ResNet50* con *PyTorch* utilizando el set de datos *Birds* inicializando los pesos en forma aleatoria. En base al set de entrenamiento, validación y test, analizar sus resultados incluyendo gráficos que indiquen:

- Evolución de función de pérdida vs número de épocas.
- Evolución de exactitud de clasificación vs número de épocas.
- Exactitud de clasificación final en set de test.

Actividad 2

Repita la actividad anterior, pero en lugar de inicializar los pesos en forma aleatoria, pre-entrene la red utilizando Imagenet y utilice los pesos resultantes como punto de inicio para entrenar *ResNet50* con el set de datos de *Birds*.

Analizar sus resultados incluyendo gráficos que indiquen:

- Evolución de función de pérdida vs número de épocas.
- Evolución de exactitud de clasificación vs número de épocas.

- Exactitud de clasificación final en set de test.

Actividad 3

Realice una tabla resumen que compare las 2 estrategias de entrenamiento. ¿Cuál es la que obtiene mejores resultados? ¿Cuál(es) clases tienen mejores y peores resultados en términos de exactitud en cada una de las estrategias de entrenamiento? Justifique y entregue ejemplos.

Actividad 4

Repita la **Actividad 1**, pero en lugar de utilizar funciones de activación del tipo ReLU, utilice funciones sigmoidales del tipo $h(x) = \frac{1}{1+e^{-x}}$. Compare sus resultados con los obtenidos en la **Actividad 1**.

2 Parte 2: Funciones de Pérdida (50%)

En esta parte de la tarea utilizarán el set de datos *VGGFaces2*:

- *VGGFaces2* es un subconjunto del set *VGGFaces*.
- 52K imágenes de train, 6.5K imágenes de validación y 6.5K imágenes de test.
- 100 clases.
- `https://drive.google.com/file/d/1torb1hkMbRyMs7CR-oGGzp1DxCOT4dzx/view?usp=sharing`.

Actividad 5

Investigar sobre las siguientes funciones de pérdida utilizadas en aprendizaje profundo: i) Triplet loss[1] y ii) Focal loss[2]. Hacer un cuadro comparativo que incluya:

- ¿Cómo funciona? Explicar intuición.
- ¿Qué parámetros tiene y para qué sirven?
- ¿Para qué tipo de tareas es posible usarlas?

Actividad 6

Entrenar *VGGFaces2* con triplet loss. Para ello deben considerar un ejemplo base de una clase (anchor) y combinar con otro positivo **aleatorio** de la misma clase y otro negativo **aleatorio** de cualquier otra clase. Luego deberán entrenar una **red de su preferencia** en *PyTorch* utilizando la función de pérdida triplet loss.

Analizar sus resultados incluyendo gráficos que indiquen, en set de entrenamiento y validación:

- Evolución de función de pérdida vs número de épocas.
- Evolución de exactitud de clasificación vs número de épocas.
- Exactitud de clasificación final en set de test.

¿Cuál(es) clases tienen mejores y peores resultados en términos de exactitud? Comente sus resultados y entregue ejemplos.

Actividad 7

¿Qué otras aplicaciones puede tener triplet loss? Investigar e indicar un ejemplo de aplicación, distinto al indicado anteriormente, donde esta función de pérdida es relevante. Justifique y entregue ejemplos prácticos.

[1] `torch.nn.TripletMarginLoss` en *pytorch*. `https://pytorch.org/docs/stable/nn.html#torch.nn.TripletMarginLoss`

[2] `https://arxiv.org/abs/1708.02002`

3 Consideraciones y formato de entrega

La tarea deberá ser entregada via SIDING en un cuestionario que se habilitará oportunamente. Se deberá desarrollar la tarea en un Jupyter Notebook con todas las celdas ejecutadas, es decir, no se debe borrar el resultado de las celdas antes de entregar. Si las celdas se encuentran vacías, se asumirá que la celda no fue ejecutada. Es importante que todas las actividades tengan respuestas explícitas, es decir, no basta con el output de una celda para responder.