Tarea 5 EL7008 – Primavera 2021
Redes neuronales convolucionales para clasificar edad

Profesor: Javier Ruiz del Solar
Auxiliar: Patricio Loncomilla

Fecha enunciado: 9 de Noviembre de 2021
Fecha entrega: 22 de Noviembre de 2021

El objetivo de esta tarea es diseñar y construir un sistema de clasificación de edad, que utilice redes convolucionales (CNN). Para esto, es necesario el uso de *pytorch* en colaboratory. La base de datos de edades a usar es la misma que en las tareas anteriores.

Se usará como base un archivo del siguiente repositorio:
https://github.com/timesler/facenet-pytorch

En particular, se debe usar el siguiente archivo como base:
https://github.com/timesler/facenet-pytorch/blob/master/models/inception_resnet_v1.py

Preparación de Conjuntos de Entrenamiento y Test

En esta tarea se debe utilizar las imágenes de la base de datos subida a U-Cursos (600 imágenes), la cual incluye 200 imágenes de personas con edades en el rango 1-4 años, 200 imágenes con personas en el rango 5-27 años y 200 imágenes de personas con 28+ años. Esta base de datos debe ser separada en 60% para entrenamiento, 20% para validación y 20% para prueba.

El archivo se debe copiar en un notebook y se debe modificar de modo que:

(a) En la clase InceptionResnetV1, sólo las últimas capas convolucionales y fully-connected sean entrenables (el resto deben ser congeladas). Las capas anteriores a block8(x) en la función forward() deben ser creadas usando *with torch.no_grad*():
(b) Crear un lector de la base de datos, que debe heredar de la clase *Dataset*. Se recomienda crear el siguiente constructor: *def __init__*(*self*, *ind0*, *ind1*), donde *ind0*,*ind1* indica el rango de imágenes que deben ser leídas del dataset. En particular, el objeto dataset debe leer las imágenes contenidos entre los índices *ind0*,*ind1* para todos los rangos de edad a la vez.
(c) Se debe agregar un código que permita entrenar y evaluar la red. Para esto, se recomienda usar *nn.CrossEntropyLoss*(), y optimizador Adam. Además, la red convolucional debe ser creada indicando *classify=True*. Se debe notar que no se debe crear carpetas nuevas con imágenes para realizar este paso, basta con usar el lector de dataset creado en el paso anterior. En el caso de los optimizadores, se debe buscar una tasa de aprendizaje apropiada para la tarea, en el caso en que los parámetros por defecto no parezcan apropiados.

Para la realización de la tarea, se pide realizar los siguientes pasos, describiéndolos en el informe:

1. Modificar el código que define la red convolucional, congelando las capas convolucionales antes de block8(x)

2. Implementar transformaciones en los conjuntos de entrenamiento, validación y prueba que implementen *data augmentation*. En la transformación de entrenamiento se debe usar ColorJitter(0.4, 0.4, 0.4), RandomCrop(140), RandomHorizontalFlip() y Normalize(mean_train,mean_std), donde mean_train y mean_std corresponden a la media y desviación estándar promedio de los pixeles en las imágenes en el conjunto de entrenamiento (por canal). En el caso de la transformación de validación y prueba, se debe usar solamente CenterCrop(140) y Normalize(mean_train,mean_std).
3. Crear el lector de dataset de caras, y los dataloaders correspondientes para el conjunto de entrenamiento, validación y prueba, usando las transformaciones definidas en el punto anterior. Las imágenes deben ser redimensionadas a 160x160 para poder ser usadas por la red. La transformación de entrenamiento se debe usar en el dataloader de entrenamiento y el de prueba se debe usar en los dataloaders de validación y prueba.
4. Programar el código principal que permita entrenar y evaluar la red (se recomienda batch de tamaño 32)
5. Entrenar una red neuronal convolucional con las imágenes del conjunto de entrenamiento. Se debe usar el conjunto de validación para prevenir sobreajuste de la red. Para cada red a ser entrenada:
    a. Calcule y grafique el *loss* de entrenamiento y el de validación. Ambas curvas deben mostrarse en un mismo gráfico, colocando las leyendas adecuadas. Indique además el tiempo de entrenamiento.
    b. Genere la matriz de confusión normalizada y el accuracy normalizado, usando el conjunto de entrenamiento (recordar usar las transformaciones de test en este ítem). Se debe mostrar la matriz de confusión con distintos colores o tonos de gris según el valor del accuracy. La matriz se debe calcular usando scikit-learn, usando la opción normalize = "true".
    c. Genere la matriz de confusión normalizada y el accuracy normalizado, usando el conjunto de validación. Se debe mostrar la matriz de confusión con distintos colores según el valor del accuracy.
    d. Genere la matriz de confusión normalizada y el accuracy normalizado, usando el conjunto de prueba. Se debe mostrar la matriz de confusión con distintos colores según el valor del accuracy.
6. Repetir los pasos del punto 6, pero esta vez usando batch_size = 8.
7. Repetir los pasos del punto 6, pero esta vez sin usar *data augmentation*.
8. Repetir los pasos del punto 6, pero esta vez sin congelar capas.
9. Repetir los pasos del punto 6, pero esta vez usando optimizador Adam con una tasa de aprendizaje distinta.
10. Repetir los pasos del punto 6, pero esta vez usando optimizador SGD.
11. Repetir los pasos del punto 6, pero esta vez usando optimizador SGD con una tasa de aprendizaje distinta.
12. Analizar el desempeño de las configuraciones probadas
13. Analizar el desempeño del sistema implementado comparando CNN v/s HOG v/s la implementación de LBP de la tarea anterior. Si no se hicieron las tareas 3 o 4, considerar para HOG accuracy 80%, y para LBP accuracy 70%.

Se debe entregar las instrucciones para su ejecución en un archivo README.txt.

Los informes, los códigos y el archivo README.txt deben ser subidos a U-Cursos hasta las 23:59 horas del lunes 22 de Noviembre. Cada día de atraso (incluyendo fines de semana) descuenta un punto.

Importante: La evaluación de esta tarea considerará el correcto funcionamiento del código, la calidad de los experimentos realizados y de su análisis, las conclusiones, así como la prolijidad y calidad del informe entregado.

Nota: Todos los pasos indicados deben ser abordados en el informe, incluyendo los trozos de código en los ítems relevantes. Además, se debe agregar el código en un anexo en modo texto (no como pantallazo).