Tarea 4 EL7008 – Primavera 2019
Clasificación de edad usando LBP

Profesor: Javier Ruiz del Solar
Auxiliar: Patricio Loncomilla
Ayudantes: Gabriel Azócar, Giovanni País, Nicolás Cruz

Fecha enunciado: Martes 1 de Octubre de 2019
Fecha entrega: Lunes 14 de Octubre de 2019

El objetivo de esta tarea es diseñar y construir un sistema de clasificación de edad, que utilice características tipo *Histogramas LBP* y un clasificador estadístico *SVM* o *Random Forest.* Se sugiere utilizar la librería OpenCV, pues contiene varias de las funcionalidades requeridas en la tarea.

Preparación de Conjuntos de Entrenamiento y Test

Para las tareas de entrenamiento y validación se deben utilizar las imágenes de la base de datos subida a U-Cursos, la cual incluye imágenes con caras de personas correspondientes a varios rangos de edad. La base de datos entregada es un subconjunto de *The Images of Groups Dataset*, en la cual se seleccionó y recortó el conjunto de caras a ser usado en esta tarea. Se entregan dos versiones distintas (db1) y (db2) de la base de datos procesada:

(db1) 200 imágenes entre 1-4 años y 200 imágenes de 5+ años

(db2) 200 imágenes entre 1-4 años, 200 imágenes entre 5-27 años, y 200 imágenes entre 28+ años

Cada versión de la base de datos entregada debe ser separada en 70% para entrenamiento y 30% para realizar la evaluación.

Extracción de Características

Histogramas LBP[1]: El método extrae las características utilizando histogramas de imágenes LBP. Se definen tres niveles diferentes de localidad: a nivel de píxel, a nivel regional y a nivel global. Los dos primeros niveles de la localidad se realizan dividiendo la imagen LBP (imagen original con trasformada LBP) del rostro en pequeñas regiones; se extraen histogramas (características a ser utilizadas). Estos histogramas se utilizan para una representación eficiente de la información de textura (ver Figura 1). El nivel global de localidad, es decir, descripción del rostro, se obtiene concatenando histogramas LBP locales. En este caso deben utilizar las características $LBP_{8,1}^{u2}$ (ver paper) y dividir en 4 regiones/cuadrantes las caras.

[1] Ver paper: T. Ahonen, A. Hadid, M. Pietikainen, “Face recognition with local binary patterns”

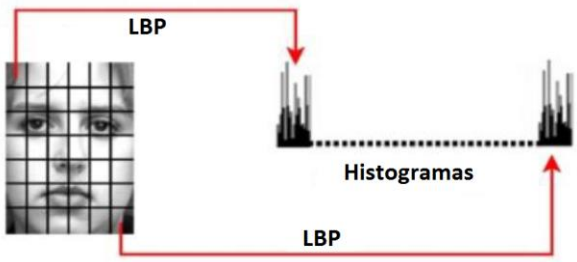

Figura 1: Histogramas LBP.

Clasificación

Se debe entrenar un clasificador SVM que pueda diferenciar entre rangos de edades utilizando los Histogramas LBP. Además, se debe entrenar un *Random Forest* con los parámetros por defecto. OpenCV tiene una implementación de SVM y una de *Random Fores*t. En esta tarea se dará soporte para OpenCV3 o superior.

Se pide:

1. Implementar en C++ una función que reciba una imagen, y retorne la imagen con la transformada LBP uniforme $\left(LBP_{8,1}^{u2}\right)$. Mostrar en el informe un ejemplo de una cara a la que se le aplique la transformada implementada.
2. Implementar en C++ Histogramas LBP (ver el paper para más detalles), utilizar las características LBP uniformes. El código debe permitir dividir la cara en n×n regiones a ser procesadas.
3. Extraer los Histogramas LBP uniformes de cada cara del conjunto de entrenamiento, considerando la base de datos (db1), usando 4x4 regiones.
4. Entrenar un SVM (*cv::CvSVM* en opencv2, *cv::ml::SVM* en opencv3) con las características extraídas a cada cara del conjunto de entrenamiento. Elegir un kernel y probar diferentes configuraciones de parámetros.
5. Entrenar un clasificador *Random Forest* (*cv::CvRTrees* en opencv2, *cv::ml::Rtrees* en opencv3) con los parámetros por defecto usando el mismo conjunto de entrenamiento.
6. Realizar las pruebas correspondientes con el conjunto de evaluación. Calcular la tasa de error del clasificador SVM y del *Random Forest* (suma de la diagonal). ¿Qué tan bien funciona cada clasificador? ¿Cómo se pueden mejorar los resultados?
7. Analizar los resultados obtenidos con los clasificadores entrenados.
8. Repetir los puntos 3-7, pero usando esta vez la base de datos (db2) y 2x2 regiones.
9. Documentar cada uno de los pasos anteriores en el informe

El código entregado debe ejecutar un entrenamiento y mostrar evaluación del clasificador entrenado. Se debe entregar **un único archivo** llamado main.cpp. Debe entregar las instrucciones para su ejecución en un archivo README.txt.

El código debe ejecutarse sin requerir ningún tipo de intervención por parte del usuario, de modo de facilitar su revisión por parte de los ayudantes. Por el mismo motivo, las imágenes deben ser leídas usando un ciclo ***for*** u otra construcción equivalente, y no línea por línea.

Los informes, los códigos y el archivo README.txt deben ser subidos a U-Cursos hasta las 23:59 horas del día lunes 14 de octubre.

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, el cual debe reflejar cada uno de los pasos solicitados en la tarea. Se entrega una pauta que indica la estructura esperada del informe.

Nota: En esta tarea, el problema corresponde a clasificación multiclase. OpenCV es capaz de resolver este tipo de problemas, pero se recomienda usar los siguientes tipos de datos:
Matriz con características de entrenamiento: CV_32FC1
Matriz con características de prueba: CV_32FC1
Matriz con etiquetas de entrenamiento (1 columna): CV_32SC1

La función predict() toma como entrada una fila conteniendo características de tipo CV_32FC1, y devuelve un número correspondiente a la clase. En el caso de *random forests*, se le debe entregar sólo una fila a la vez.

Nota: Es necesario usar CV_32SC1 para la matriz de etiquetas de entrenamiento para que OpenCV detecte que el problema es multiclase. Las características deben ser CV_32FC1