Práctica obligatoria 2

Reconocimiento de objetos

El enunciado corresponde a la primera práctica puntuable de la asignatura: **entrega día 29 de mayo de 2021.**

1 Copias de código o de la memoria

El código desarrollado en las prácticas debe de ser original. La copia (total o parcial) de prácticas será sancionada, al menos, con el SUSPENSO global de la asignatura en la convocatoria correspondiente. En estos casos, además, no regirá la liberación de partes de la asignatura (habrá que volver a presentarse al examen) y podrá significar, en la siguiente convocatoria y a discreción del profesor, el tener que **resolver nuevas pruebas y la defensa de las mismas de forma oral**. **Las sanciones derivadas de la copia, afectarán tanto al alumno que copia como al alumno copiado.**

Para evitar que **cuando se usa código de terceros** sea considerado una copia, se debe **citar siempre la procedencia** de cada parte de código no desarrollada por el propio alumno (con comentarios en el propio código y con mención expresa en la memoria de las prácticas). El plagio o copia de terceros (p.ej. una página web) ya sea en el código a desarrollar en las prácticas y/o de parte de la memoria de las prácticas, sin la cita correspondiente, acarrearán las mismas sanciones que en la copia prácticas de otros alumnos.

2 Reconocimiento de señales de tráfico

Se desea construir un sistema que permita el reconocimiento de ciertas señales de tráfico en imágenes realistas (tomadas en la calle desde un coche).

Siguiendo un enfoque Top-Down, el alumno deberá construir dos funciones:

- Función de aprendizaje: La primera función recibirá el nombre de un directorio que contendrá las imágenes de aprendizaje. La función utilizará dichas imágenes para entrenar un clasificador que posteriormente será usado en la siguiente función.

- Función de reconocimiento: Esta segunda función recibirá un clasificador entrenado y la imagen recortada de una señal. La función devolverá una cadena con el nombre de la clase que predice el clasificador.

2.1 Sistema básico de reconocimiento

La figura 1 muestra la línea base de un sistema de reconocimiento que funciona para el problema de las señales de tráfico. Este funcionamiento básico es el que el alumno debe de seguir para construir una aplicación que reconozca las señales de tráfico.

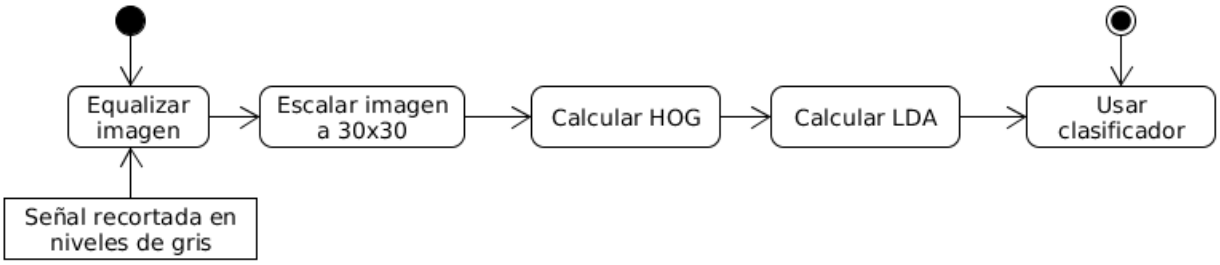

Figura 1. Esquema de funcionamiento del sistema de reconocimiento básico a implementar.

En este sistema básico podemos distinguir 3 partes bien diferenciadas:

1. Extracción del vector de características: ecualización, escalado de la imagen y extracción de HOG (Histograma de Orientación de Gradientes). Usar para ello la clase: cv2.HOGDescriptor y el método "compute" de esa clase para calcular el vector de características HOG de una imagen (ver referencias que explican el HOG al final del enunciado).

2. Reducción de la dimensionalidad con LDA.

3. Entrenamiento del clasificador Bayesiano con Gaussianas. Hay dos posibilidades en este caso:

    - cv2.ml.NormalBayesClassifier de OpenCV.

    - sklearn.discriminant_analysis.LinearDiscriminanAnalysis ofrece la reducción de dimensionalidad de LDA (con los métodos fit y transform) y además proyección LDA + clasificación con un Bayesiano con Gaussianas (fit y predict).

2.2 Ejercicio 1

El ejercicio 1 consiste en desarrollar el sistema de reconocimiento básico utilizando uno de los clasificadores Bayesianos con Gaussianas y obtener una tasa de acierto superior al 50% en un problema de 42 clases de señales de tráfico para las que se disponen de datos de entrenamiento.

Además de disponer de un código funcional, se valorará:

- La descripción de las salidas obtenidas en cada uno de los pasos del algoritmo.

2.3 Ejercicio 2

El ejercicio 2 consiste en comparar los resultados que ofrece el sistema de reconocimiento con otras alternativas. Entre las posibles alternativas se proponen:

- Otros vectores de características:

    - Por ejemplo, utilizando el color u otras medidas disponibles en OpenCV o en skimage.feature (http://scikit-image.org/docs/dev/api/skimage.feature.html). Un vector de características típico en el caso de detección de objetos es el LBP (o Locally Binary Patterns) o también las Haar-like-features.

- Otros algoritmos de reducción de dimensionalidad:
    - Por ejemplo, PCA (Principal Component Analysis) disponible en sklearn (http://scikit-learn.org/stable/modules/decomposition.html).
- Otros clasificadores:
    - Euclídeo, KNN o cualquier otro clasificador (supervisado) disponible en OpenCV y/o Sklearn.

El código desarrollado deberá permitir elegir cada uno de los diferentes clasificadores mediante el parámetro correspondiente.

Además, la comparativa quedará reflejada en la memoria de la práctica. La claridad y adecuación en la exposición de los resultados (tasas de acierto, matrices de confusión, gráficos, curvas ROC o de Precisión/Recall, …) será valorada en la nota.

Para obtener la máxima puntuación habría que probar con relativo éxito 3 alternativas (entendiendo como alternativa el cambio de cualquiera de los componentes del sistema básico – extracción de características, algoritmo de reducción de dimensionalidad o clasificador).

2.4 Ejercicio 3

El ejercicio 3 consiste en unir el sistema de reconocimiento diseñado en el apartado anterior con el detector de señales de tráfico desarrollado en la práctica obligatoria 1, de manera que el sistema detecte y reconozca las señales que hay dentro de una escena de carretera. **Se puede plantear este apartado de varias maneras:**

a) Entrenar un clasificador señal/no señal para mejorar la detección del algoritmo de la práctica 1 (puede realizarse con imágenes en color).

b) Entrenar un clasificador con tres clases: peligro, prohibición, stop.

c) Entrenar un clasificador para distinguir entre diferentes señales de peligro, de prohibición y la de stop (el número de clases será menor que 42).

Si una imagen tiene un tamaño superior a 100x100 píxeles el sistema supondrá que corresponde a una escena real, en otro caso se supondrá que es una señal recortada.

Deberá quedar reflejada en la memoria de la práctica los resultados obtenidos sobre las escenas de test suministradas (dando las estadísticas de detecciones con el nuevo clasificador, tal y como se hizo en la práctica 1, y también proporcionando las imágenes con el resultado de detecciones y clase de cada señal de trafico sobre-impresionada).

Para obtener una puntuación máxima se valorará la prueba de todo el sistema sobre un pequeño vídeo (de menos de 1 minuto). Para ello, el alumno podrá de capturar su propio vídeo utilizando un coche y una cámara (una de mano o la del móvil) o utilizar un vídeo de youtube y generar una secuencia de imágenes a partir de dicho vídeo en el que se muestren las detecciones y la señal reconocida. Un ejemplo de visualización de los resultados de detección + reconocimiento podría ser el siguiente:

https://youtu.be/lr5_4XXgjDY?list=PLcD_yLvcdUlmQx0P95ijBucBXmRbmN1_U

3 Datos de entrenamiento, test y formato de salida

Los **datos de entrenamiento** proporcionados son imágenes .ppm con señales recortadas. Dichas imágenes se proporcionan en varios subdirectorios. Cada subdirectorio contendrá un número variable de imágenes de señales del mismo tipo. El nombre del subdirectorio se corresponderá con el tipo de señal.

También se proporciona un directorio con imágenes de test recortadas de los mismos ficheros de test utilizados en la práctica obligatoria 1. Obsérvese que los dos primeros caracteres del nombre de cada imagen de test identifica el tipo de señal. Esta información debe utilizarse para poder contar los éxitos y generar las estadísticas (tasas de acierto, matrices de confusión, gráficas, etc).

Finalmente, para la última parte de la práctica se pueden usar las imágenes de test de la práctica obligatoria 1.

Todas las prácticas entregadas deberán:

- Utilizar el script python *main.py* que se proporciona junto con este enunciado.
- La llamada a *main.py* tiene la siguiente estructura:

  python main.py –train_path /un/ejemplo --test_path /un/ejemplo [–classifier BAYES]

  - **El primer parámetro** es el directorio donde están los subdirectorios de entrenamiento.
  - **El segundo parámetr**o es el directorio donde están las imágenes a reconocer. Si una imagen tiene un tamaño igual o inferior a 100x100 píxeles se supondrá el caso base (una señal recortada), si el tamaño es mayor a 100x100 (en cualquiera de las dos dimensiones) se supondrá que es una escena real y deberá llamar a la función de segmentación desarrollada en la práctica 1.
  - **El tercer parámetro** es el tipo de clasificador (con sus alternativas del ejercicio 2): LDA-BAYES, KNN-PCA, etc.
- El resultado de *main.py* deberá ser un fichero “resultado.txt”, con una línea por cada resultado de reconocimiento. Dicha línea seguirá el siguiente formato:

  ```
  00_00000.ppm; 00
  01_00008.ppm; 01
  ```

  Obsérvese que en el caso de imágenes de señales recortadas solo se genera una línea por fichero, pero en el en caso de imágenes de escenas reales pueden generarse más de una línea por fichero.

4 Normas de presentación

La presentación seguirá las siguientes normas:

- Su presentación se realizará a través del Aula Virtual**.**
- Para presentarla se deberá entregar un único fichero ZIP que contendrá el código fuente en python y un fichero PDF con la descripción del sistema desarrollado.
- Dicho fichero PDF incluirá una explicación con el algoritmo desarrollado, los métodos de OpenCV y/o sklearn utilizados, copias de las pantallas correspondientes a la ejecución del programa y unas estadísticas correspondientes al resultado de la ejecución del programa sobre las muestras de test. Si se han probado diferentes sistemas de clasificación deberán reflejarse los diferentes resultados que se hayan obtenido.
- Se permitirán grupos de hasta 3 alumnos.

La puntuación de esta práctica corresponde al 35 % de la asignatura. La práctica se valorará sobre 10 y cada parte tendrá la siguiente puntuación:

- Ejercicio 1: **6 puntos.**
- Ejercicio 2: **2 puntos.**
- Ejercicio 3: **2 puntos (hasta 1 punto sólo una opción entre a), b) o c), y 2 puntos con dos o más opciones implementadas.**

Para obtener la máxima nota en cada apartado, se valorará:

- Implementar el *main.py* de la manera como se pide en la sección 5.
- La limpieza y organización del código en clases con un Diseño Orientado a Objetos razonable.
- El funcionamiento del código en las imágenes de test.
- Las ideas propias o técnicas pensadas por los alumnos para mejorar lo que se propone en el enunciado.

5 Referencias

1. Explicación del vector de características HoG:

   http://www.learnopencv.com/histogram-of-oriented-gradients/

2. Documentación de del descriptor HOG en OpenCV:

   http://docs.opencv.org/trunk/d5/d33/structcv_1_1HOGDescriptor.html

3. Ejemplo de uso de LDA:

   http://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html