Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Tratamiento de Señales Visuales (TSV)

Grado en Ingeniería Informática

Tratamiento de Señales Multimedia I: señales visuales (TSMI)

Práctica 3
Reconocimiento de Escenas con modelos Bag-of-Words

Universidad Autónoma de Madrid

Escuela Politécnica Superior

Video Processing and Understanding Lab

OBJETIVO

- Reconocimiento de escenas mediante el uso de descriptores y clasificadores basados en bag of words

entrenamiento → Clasificador

Características
BoW
KNN/SVM
…

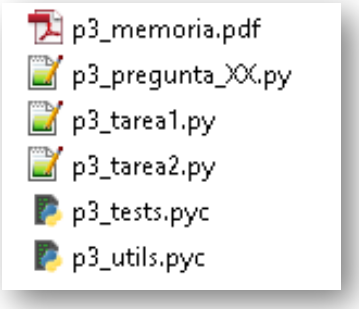
Imagen test

clasificación → ¿Categoría?

ENTREGA

- Descargue los ficheros de código Python disponibles en Moodle
  - Fichero `p3_bow_code.zip`

- Complete las tareas utilizando los ficheros *.py contenidos en el ZIP
- La entrega se **realiza por parejas** y constará de:
  - Memoria en PDF (1 fichero) + código Python adicional
  - Ficheros código Python de las tareas realizadas (hasta 2 ficheros)
  - Fichero p3_tests.pyc y p3_utils.pyc



  - **No entregar carpetas 'dataset' y 'test_data'**

No es necesario entregar todas las tareas

TAREAS

- Tarea 0 – Descargar material e incluir nombre en ficheros
- Tarea 1 – Implementar el modelo BOW
- Tarea 2 – Implementar descriptores HOG
- Responder a las preguntas en memoria

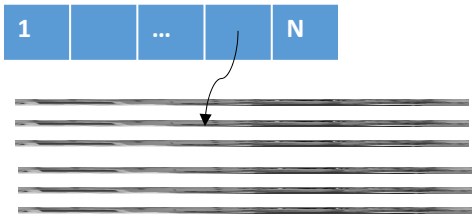
- Funciones prohibidas:
  - Cualquier función que no sea del paquete de `numpy` y `scipy`
  - Se pueden utilizar las funciones recomendadas de `skimage`
  - No se pueden utilizar las funciones *img_as_float64, img_as_float32, img_as_float* para normalización de imágenes. Se debe realizar la operación manualmente.

Aumento de complejidad con respecto a la práctica anterior y mayor número de preguntas en memoria
(se deben consultar los apuntes de teoría y artículos)

TAREA 1 1/2

- Implementar el modelo Bag-Of-Words (`p3_tarea1.py`)
  - Función **`construir_vocabulario(list_img_desc, vocab_size=10, max_iter=300)`**
  - Consideraciones:
    - Es resultado es el vocabulario en un array `numpy` con dimensión `[vocab_size,D]`
  - Funciones recomendadas:
    - Para agrupar, utilice la función `sklearn.cluster.Kmeans` con el parámetro `random_state=0`. En esta función utilice los parámetros `vocab_size` y `max_iter` de la función **`construir_vocabulario`**
    - Para convertir la lista en arrays numpy, utilice `numpy.concatenate`

Lista con descriptores para N imágenes



M descriptores de longitud D

Construir vocabulario

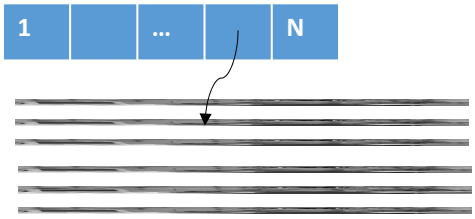
`vocab_size` centroides de longitud D (representantes grupos de descriptores)

TAREA 1 2/2

- Implementar el modelo Bag-Of-Words (`p3_tarea1.py`)
  - Función **`obtener_bag_of_words(list_img_desc, vocab)`**
  - Consideraciones:
    - El resultado es un array `numpy` con dimensión `[N, vocab_size]` donde cada fila es el histograma BOW (normalizado) de cada imagen
  - Función recomendadas:
    - Para calcular distancias utilice `scipy.spatial.distance.cdist` con valores por defecto.

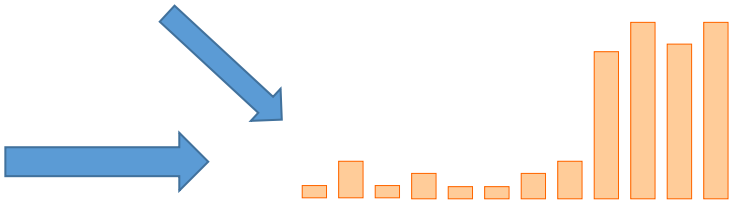
Repetir proceso para todas las imágenes de la lista

Lista con descriptores para N imágenes



M descriptores de longitud D

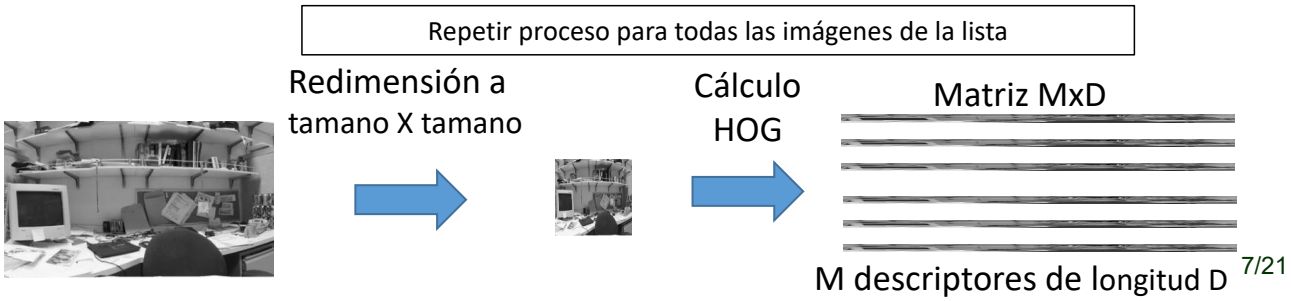
Vocabulario BOW (`vocab_size x D`)



descriptor de longitud `vocab_size`

TAREA 2

- **Implementar extracción de descriptores** (`p3_tarea1.py`)
  - Función `obtener_features_hog(path_imagenes,tamano=100, orientaciones=9, pixeles_por_celda=(8, 8),celdas_bloque=(4,4)`
  - Consideraciones:
    - Convierta cada imagen a gris, formato `float` y en el rango [0,1]
    - La salida de esta función es una lista 1xN, donde cada posición es un `numpy array` con los descriptores calculados para cada imagen
  - Funciones
    - Para calcular HOG, utilice `skimage.feature.hog` con los parámetros *orientations=orientaciones, pixels_per_cell=pixeles_por_celda, cells_per_block=celdas_bloque, feature_vector=False* (resto por defecto)
    - Recomendadas (`Skimage`): `io.imread, color.rgb2gray, transform.resize`



TAREA - MEMORIA

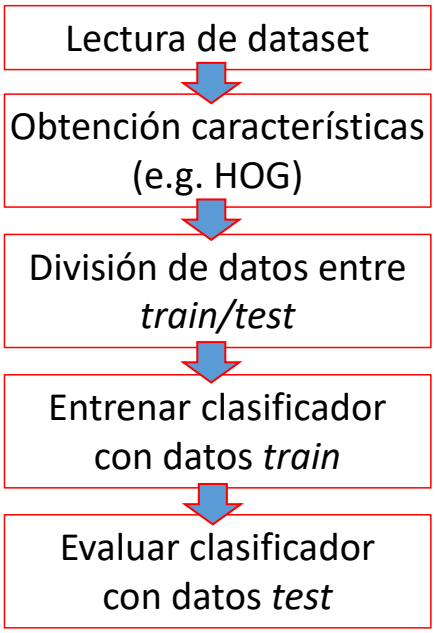
- Responda a las preguntas del fichero “preguntas_P3.docx”
  - No necesita realizar funciones nuevas, el trabajo se basa en realizar experimentos con la funcionalidad que ha generado
  - Puede necesitar repasar algunos conceptos teóricos
  - Utilice figuras y tablas para resumir experimentos

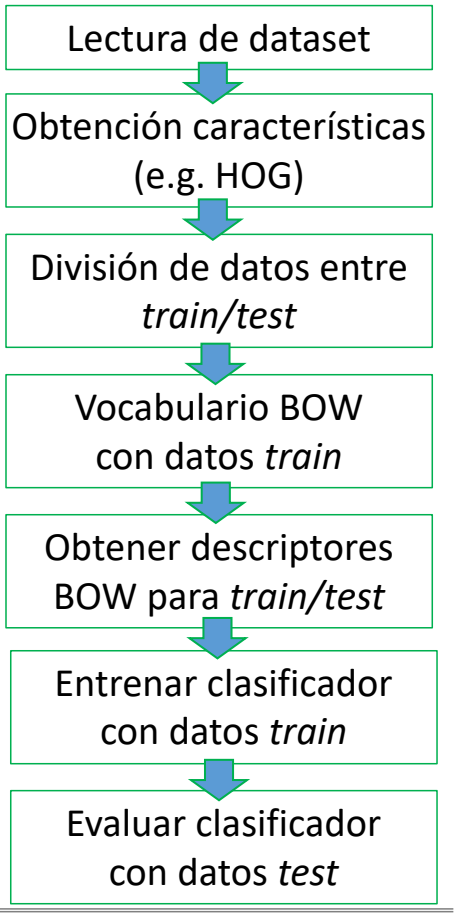
- Ficheros a generar
  - Fichero “preguntas_P3.pdf” en formato PDF con las respuestas
  - Genere tantos ficheros Python como necesite con el formato `p3_pregunta_XX.py` donde `XX` es el número de pregunta

- Criterios generales de evaluación: texto conciso y claro; relación con teoría; numeración de diagramas, figuras y tablas; referencias a figuras y tablas en texto; referencias a fuentes externas (si procede); errores ortográficos.

TAREA - MEMORIA

- Las preguntas requieren construir varios sistemas de clasificación del dataset con tareas 1/2, clasificadores KNN/SVM/RF y características
  - Opción 1: Raw/descriptores
  - Opción 2: BOW descriptores





TAREA - MEMORIA

- Funciones útiles: lectura del dataset (fichero `p3_utils.pyc`) 1/2

```
data = load_image_dataset(container_path, *, description=None,
categories=None,load_content=True, shuffle=True,random_state=0,
resize_shape=None, max_per_category=None,debug=False):
```

Datos de imágenes

Etiquetas

Etiquetas (nombres)

Rutas de imágenes

```
-------
data : :class:`~sklearn.utils.Bunch`
    Dictionary-like object, with the following attributes.

    data : list of str
        Only present when `load_content=True`.
        The raw image data to learn.
    target : ndarray
        The target labels (integer index).
    target_names : list
        The names of target classes.
    DESCR : str
        The full description of the dataset.
    filenames: ndarray
        The filenames holding the dataset
```

- Funciones útiles: lectura del dataset (fichero `p3_utils.pyc`) 2/2

```
data = load_image_dataset(container_path, *, description=None,
categories=None,load_content=True, shuffle=True,random_state=0,
resize_shape=None, max_per_category=None,debug=False):
```

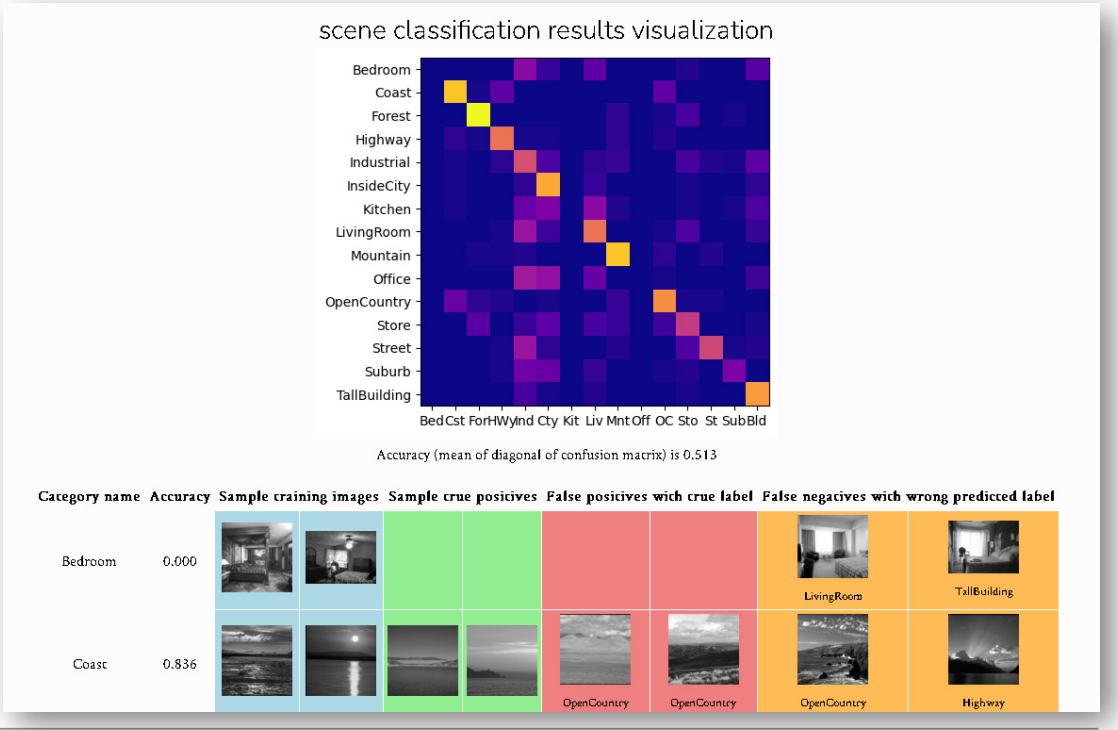
```
#example 1 - Load dataset
# 100 images per category
# resize all images to (200,200))
dataset_path = './datasets/scenes15'
data = load_image_dataset(container_path=dataset_path, shuffle=True
        load_content=True, resize_shape=(200,200),max_per_category=100)
```

```
#example 2 - Load dataset
# 200 images per category
# does not load images (only paths) - useful for computing features later
dataset_path = './datasets/scenes15'
data = load_image_dataset(container_path=dataset_path, shuffle=True,
                load_content=False, max_per_category=200)
```


- Funciones útiles: visualización resultados (fichero `p3_utils.pyc`) 1/3

```
create_results_webpage(...):
```

La función genera una página web html con resultados cualitativos y cuantitativos



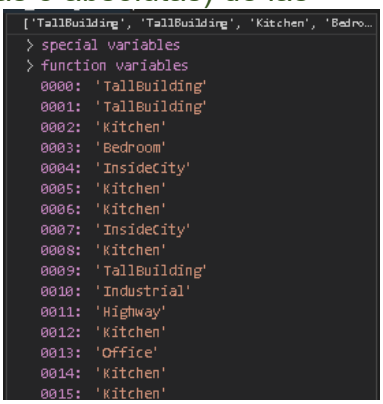
• Funciones útiles: visualización resultados (fichero `p3_utils.pyc`) 2/3

```
confusion_matrix = create_results_webpage
(train_image_paths, test_image_paths,
train_labels, test_labels,categories, abbr_categories, predic
ted_categories, name_experiment):
```

- **confusion_matrix** → matriz de confusión obtenida
- **train_image_paths** → Lista de Python con las rutas (relativas o absolutas) de las imágenes de entrenamiento
- **test_image_paths** → Lista de Python con las rutas (relativas o absolutas) de las imágenes de entrenamiento

- **train_labels,**
- **test_labels**
- **predicted_categories**

Lista de Python con las categorías reales para los datos de train/test y predichas solamente para los datos de test



• Funciones útiles: visualización resultados (fichero `p3_utils.pyc`) 3/3

```
confusion_matrix = create_results_webpage
(train_image_paths, test_image_paths,
train_labels, test_labels,categories, abbr_categories, predicted
_categories, name_experiment):

# This is the list of categories / directories to use.
# The categories are sorted by alphabetical order
categories = ['Bedroom', 'Coast', 'Forest', 'Highway', 'Industrial',
              'InsideCity', 'Kitchen', 'LivingRoom', 'Mountain', 'Office',
              'OpenCountry', 'Store', 'Street', 'Suburb', 'TallBuilding']

# This list of shortened category names is used later for visualization.
abbr_categories = ['Bed','Cst','For','HWy', 'Ind','Cty','Kit','Liv','Mnt',
                   'Off','OC','Sto','St','Sub','Bld']
```

name_experiment → string que describe el experimento realizado. Los resultados se guardan en la carpeta *./p3_results_webpage/ name_experiment* (e.g. name_experiment ='HOG_BOW_SVM', genera la página html de resultados en ./p3_results_webpage/ HOG_BOW_SVM/index.html)

TAREA - MEMORIA

- Funciones útiles:
  - Para dividir el dataset entre train/test (paquete `sklearn`)
    - `sklearn.model_selection.train_test_split`
  - Para clasificar, KNN y SVM del paquete `sklearn`
    - KNN `sklearn.neighbors.KNeighborsClassifier`
    - SVM `sklearn.svm`
    - RF `sklearn.ensemble.RandomForestClassifier`
  - Para evaluar el rendimiento el dataset
    - `sklearn.neighbors.KNeighborsClassifier.score`
    - `sklearn.svm.score`
    - `sklearn.ensemble.RandomForestClassifier.score`
  - Documentación:
    - https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC
    - https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
    - https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

TAREAS – AUTOEVALUACION

- ¿Cómo comprobar si la implementación es correcta?
  - Se proporciona funciones que se pueden ejecutar cuando se desee
    - **`test_p3_tarea1(…)`** en el fichero `p3_tarea1.py`

```
Practica 3 - Tarea 1
Realizando tests para las funciones 'construir_vocabulario' y 'obtener_bags_of_words' de la tarea 1
La funcion es correcta si los resultados obtenidos tienen una tolerancia de 2 decimales con respecto a la salida correcta.

 * Utilizando datos en fichero '.\p3_tarea1.data'
 * Tests para 'construir_vocabulario':
       Testeando con tamano de vocabulario 5 (20 descriptores Tiny de dimension 256)... OK
       Testeando con tamano de vocabulario 5 (1620 descriptores HOG de dimension 144)... OK
       Testeando con tamano de vocabulario 10 (20 descriptores Tiny de dimension 256)... OK
       Testeando con tamano de vocabulario 10 (1620 descriptores HOG de dimension 144)... OK
 * Tests para 'obtener_bags_of_words':
       Testeando con tamano de vocabulario 5 (20 descriptores tipo Tiny de dimension 256)... OK
       Testeando con tamano de vocabulario 5  (1620 descriptores tipo HOG de dimension 144)... OK
       Testeando con tamano de vocabulario 10 (20 descriptores tipo Tiny de dimension 256)... OK
       Testeando con tamano de vocabulario 10  (1620 descriptores tipo HOG de dimension 144)... OK
 * Finalizado en 2.435 secs
 * RESULTADO 'construir_vocabulario' con descriptor Tiny/HOG: 4/4 CORRECTOS ( 100.00% )
 * RESULTADO 'obtener_bags_of_words' con descriptor Tiny/HOG: 4/4 CORRECTOS ( 100.00% )
Tests completados = True
```

    - **`test_p3_tarea2(…)`** en el fichero `p3_tarea2.py`

```
Practica 3 - Tarea 2
Realizando tests para las funciones 'obtener_features_tiny' y 'obtener_features_hog' de la tarea 2
La funcion es correcta si los resultados obtenidos tienen una tolerancia de 2 decimales con respecto a la salida correcta.

 * Utilizando datos en fichero '.\p3_tarea2.data'

 * Tests para 'obtener_features_hog':
       Testeando con tamano 50... Imagen #0 #1 #2 #3 #4 #5 #6 #7 #8 #9
       Testeando con tamano 100... Imagen #0 #1 #2 #3 #4 #5 #6 #7 #8 #9
 * Finalizado en 1.469 secs
 * RESULTADO 'obtener_features_hog': 20/20 DESCRIPTORES CORRECTOS ( 100.00% )
Tests completados = True
```

TAREAS – AUTOEVALUACION

No hay funciones de auto-evaluación
para preguntas en la memoria pero…

- Accuracy/rendimiento de clasificación sobre datos de test (utilizando como entrenamiento 200 imágenes por categoría)
  - ~0% → funcionamiento erróneo
  - ~7% → rendimiento próximo a una clasificación aleatoria
  - ~20% → rendimiento para características Tiny* con KNN (*k=1*)
    Se puede aumentar ligeramente ajustando parámetro *k*
  - ~45% → rendimiento para características HOG-BOW y KNN
    Se puede aumentar ligeramente ajustando parámetro *k*
  - ~50% → rendimiento para características HOG-BOW y SVM (lineal)
  - ~55% → rendimiento para características HOG-BOW y RF (parámetros defecto)
  - ~70% → rendimiento para características HOG-BOW y SVM
    Ajuste de parámetros SVM, extracción HOG y diccionario BOW.

*Para cada imagen, la característica Tiny corresponde a (1) reducir sus dimensiones a un tamaño dado (e.g. imagen 8x8) y (2) transformarlo a un vector fila que será el descriptor Tiny de la imagen asociada (e.g. 64x1 valores)

CARGA DE TRABAJO

- 3 sesiones de prácticas (16 horas): 6h presenciales + 10h no presenciales
- Sugerencia de realización:

TAREA	Horas presenciales	Horas no presenciales	Horas TOTAL	Semana de prácticas
Explicación	0.5 h	0 h	0.5 h	1
Tarea 1	1.5 h	2 h	3.5 h	1
Tarea 2	2 h	0 h	2 h	2
Memoria	2 h	8 h	10 h	2-3
TOTAL	6h	10h	16h	

EVALUACION

- Evaluación de la práctica sobre 10 puntos

TAREA	Max nota	Criterio evaluado (ver rúbrica en Moodle)
Tarea 1	**2.5**	**Código: Ejecución (60%) + Diseño(40%)**
Tarea 2	**2**	**Código: Ejecución (60%) + Diseño(40%)**
Memoria	**5.5**	**Memoria: Claridad, exactitud de la respuesta y experimentos realizados por cada pregunta (100%)**
TOTAL	**10**	

- Penalizaciones:
  - Por entrega de ficheros no acorde a las especificaciones: -0.5 puntos
  - Por entrega de memoria con formato incorrecto: -0.5 puntos
  - Por uso de funciones prohibidas: -50% tarea
  - Por entrega tardía (tras considerar los 4 días disponibles para cada pareja):
    - -25% (un día), -50% (dos días), -75% (tres días), -100%(>= días)

REFERENCIAS

- Wikipedia
  https://en.wikipedia.org/wiki/Bag-of-words_model
- **Descriptor Histograma de Gradientes Orientados**
  - Richard Szeliski, "Computer Vision: Algorithms and Applications" 2020, **Sección 7.1.2**. http://szeliski.org/Book/2ndEdition.html
  - Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, *60*(2), 91-110
- **Modelo Bag-of-words**
  - Richard Szeliski, "Computer Vision: Algorithms and Applications" 2020, **Sección 6.2**. http://szeliski.org/Book/2ndEdition.html

SUGERENCIA

- Esta práctica presenta un incremento sustancial en complejidad respecto a prácticas anteriores.
- El trabajo efectivo y coordinado de la pareja es crítico para poder realizar todos los ejercicios satisfactoriamente.
- Sugerencia:
  - Tareas 1 y 2: realizar conjuntamente
  - Preguntas memoria:
    - Analizar que tareas son comunes a todas las preguntas (leer dataset, extraer características,…)
    - Repartir las tareas entre la pareja y realizar individualmente
    - Escribir los resultados en la memoria individualmente
    - Realizar un análisis y revisión conjunto de los resultados en memoria