Clasificador de Imágenes

Sistema Inteligente 25/26

Hecho por:

- Zixin Zheng

Índice

Introducción.....3

Solución propuesta.....3

Datos de entrada.....6

Estado del arte.....6

Conclusión.....7

Evolución experimentos.....7

Introducción

El objetivo de este trabajo consiste en desarrollar un sistema inteligente capaz de clasificar imágenes en categorías, Comidas/Personas, mediante técnicas de Visión Artificial y Aprendizaje Automático.

El reto consiste en generalizar correctamente imágenes no vistas antes en el entrenamiento de la máquina y que esta sea capaz de clasificarla correctamente, en Personas, que presenta una gran variabilidad en postura, ropa, fondo e iluminación.

Además, para el entrenamiento se ha creado un dataset propio con imágenes de comida personales del álbum de alumno y fotos de personas que se han obtenido en un banco de imágenes gratuito "pexels.com", porque no tenía suficientes fotos para el entrenamiento de personas.

Solución propuesta

Para resolver este problema se ha planteado utilizar la Percepción Computacional y el Aprendizaje Automático junto con una toma de decisiones híbrida para una solución más robusta.

El lenguaje de programación utilizado es Python y se han empleado librerías OpenCV, Numpy y Google Colab. Las imágenes antes de ser subidas para el entrenamiento se han recortado para adaptarse y se han asignado a cada imagen dependiendo de la categoría, Comida_x o Persona_x. Las imágenes se redimensionan en el programa a un tamaño de 64x64 píxeles que reduce la complejidad computacional y permite comparar imágenes de forma consistente.

```
import cv2   import numpy as np
from google.colab.patches import cv2_imshow
```

Para la parte de Percepción Computacional está la función de extracción de características, donde genera por cada imagen un vector de características, combinando información de color y forma. Además de usar un histograma de color en el espacio HSV que es más robusto que RGB frente a cambios de iluminación. También para capturar la estructura visual de las imágenes se emplea el HOG (Histograma de Gradientes Orientado), que analiza la distribución de gradientes de intensidad y es eficaz para detectar siluetas y formas (útil para Personas). Después se fusionan ambos descriptores en un único vector.

```python
#Extracción de características
def extraer_caracteristicas(ruta_imagen, alpha_color=3.0, beta_hog=1.0):
    img = cv2.imread(ruta_imagen)
    if img is None:
        return None

    img = cv2.resize(img, (64, 64), interpolation=cv2.INTER_AREA)

    #Histograma HSV
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
    hist = cv2.calcHist([hsv], [0, 1, 2], None, [12, 8, 8], [0, 180, 0, 256, 0, 256])
    hist = cv2.normalize(hist, None).flatten().astype(np.float32)

    #Forma/Textura: HOG
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    hog = cv2.HOGDescriptor(
        _winSize=(64, 64),
        _blockSize=(16, 16),
        _blockStride=(8, 8),
        _cellSize=(8, 8),
        _nbins=9
    )
    hog_desc = hog.compute(gray).flatten().astype(np.float32)

    #Fusion
    vector = np.concatenate([alpha_color * hist, beta_hog * hog_desc]).astype(np.float32)
    return vector, img
```

Extracción de características

Antes del entrenamiento se aplica una estandarización de tipo Z-score, calculando la media y desviación estándar sobre el conjunto de entrenamiento. Esto es fundamental para el uso de la distancia euclidiana y evita que unas características dominen sobre otras.

```python
#Distancia y estandarización
def distancia_euclidiana(v1, v2):
    diff = v1 - v2
    return np.sqrt(np.dot(diff, diff))

def fit_standardizer(X_train):
    mu = X_train.mean(axis=0)
    sigma = X_train.std(axis=0)
    sigma[sigma < 1e-8] = 1.0
    return mu, sigma

def transform_standardizer(X, mu, sigma):
    return (X - mu) / sigma
```

Estandarización Z-score

Los algoritmos empleados son el KNN balanceado, que hace uso de la distancia euclidiana como medida de similitud y el K se selecciona automáticamente mediante validación interna. El otro algoritmo es el MVN por centroides, que calcula un centroide para cada clase a partir de los datos de entrenamiento y esto ofrece una visión global del problema y resulta muy estable, gracias a esto las tasas de acierto han subido en comparación con utilizar el KNN como prioritario en la toma de decisiones en nuestro sistema híbrido ante conflicto de los algoritmos.

```python
#Algoritmo KNN balanceado
def obtener_voto_knn_balanceado(k, X_train, y_train, vector_nuevo, eps=1e-9):
    clases, counts = np.unique(y_train, return_counts=True)
    w_class = {c: 1.0 / counts[i] for i, c in enumerate(clases)}

    distancias = []
    for i in range(len(X_train)):
        d = distancia_euclidiana(vector_nuevo, X_train[i])
        distancias.append((d, y_train[i]))

    distancias.sort(key=lambda x: x[0])
    vecinos = distancias[:k]

    votos = {}
    for d, etiqueta in vecinos:
        w_dist = 1.0 / (d + eps)
        votos[etiqueta] = votos.get(etiqueta, 0.0) + w_dist * w_class[etiqueta]

    return max(votos.items(), key=lambda x: x[1])[0]
```

KNN balanceado

```python
#Algoritmo MVN
def entrenar_centroides(X_train, y_train):
    centroides = {}
    for c in np.unique(y_train):
        centroides[c] = X_train[y_train == c].mean(axis=0)
    return centroides

def obtener_voto_centroides(centroides, vector_nuevo):
    mejor_clase = None
    mejor_dist = float("inf")
    for c, centro in centroides.items():
        d = distancia_euclidiana(vector_nuevo, centro)
        if d < mejor_dist:
            mejor_dist = d
            mejor_clase = c
    return mejor_clase
```

MVN centroides

```python
#Sistema híbrido(Para esta versión se ha cogido en caso de conflicto MVN
#porque tiene más porcentaje de aciertos)
def clasificador_hibrido(vector, X_train, y_train, centroides, k=5):
    voto_knn = obtener_voto_knn_balanceado(k, X_train, y_train, vector)
    voto_cent = obtener_voto_centroides(centroides, vector)

    if voto_knn == voto_cent:
        return voto_knn, "Acuerdo"
    else:
        return voto_cent, "Conflicto -> Gana MVN"
```

Sistema de decisión híbrida

Datos de entrada

Se han creado dos carpetas, Personas y Comidas, en la carpeta de "/content" del Google Colab. En Comidas se han implementado 110 imágenes y en Personas 60 imágenes. Todas las imágenes solo han sido recortadas y presentan variabilidad de iluminación, fondo, resolución y ángulos.

Estado del arte

La clasificación automática de imágenes es un problema ampliamente estudiado dentro del campo de la Visión Artificial y el Aprendizaje Automático. Tradicionalmente, este tipo de problemas se ha abordado mediante la extracción manual de características visuales y el uso de clasificadores supervisados.

Entre los descriptores de color más utilizados se encuentran los histogramas de color, que permiten representar la distribución cromática de una imagen de forma compacta. En particular, el uso del HSV ha demostrado ser más robusto frente a cambios de iluminación que el espacio RGB, ya que separa la información de color de la intensidad luminosa.

En cuanto a la descripción de forma y textura, los Histogramas de Gradientes Orientados (HOG) constituyen una técnica clásica y eficaz para capturar la estructura y silueta de los objetos en una imagen. Este descriptor ha sido ampliamente utilizado en tareas de detección de personas y análisis de escenas, debido a su robustez frente a variaciones de iluminación y fondo.

Para la fase de clasificación, algoritmos como KNN y los clasificadores basados en centroides (MVN) han sido empleados con éxito en numerosos problemas de clasificación supervisada, especialmente cuando se dispone de conjuntos de datos de tamaño moderado y descriptores explícitos.

Conclusión

El sistema desarrollado implementa los bloques de Percepción Computacional y Aprendizaje automático, junto con un dataset propio se han obtenido resultados estables y aceptables con un porcentaje de exito de más de 93%, gracias a la combinación de clasificadores con un sistema híbrido que mejora la robustez del sistema.

Como mejoras futuras se podría ampliar el dataset para una mejor generalización, además de igualar los números de imágenes de Comidas con el de Personas. También implementar nuevas clases a clasificar o técnicas de aprendizaje más complejo que ayuden para clasificar mejor las imágenes.

Evolución experimentos

En la primera versión del sistema programado:

- Se cogen en total 160 imágenes, 110 de comida y 50 de persona, para el entrenamiento se cogen el 80% de las imágenes de forma aleatoria y el resto se usan para el testing (32 casos).

Caso de Prueba 1:

- Precisión Global: 90.62%
- Conflictos entre KNN y MVN: 5 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

Caso de Prueba 2:

- Precisión Global: 96.88%
- Conflictos entre KNN y MVN: 4 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

Caso de Prueba 3:

- Precisión Global: 87.50%
- Conflictos entre KNN y MVN: 2 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Comidas -> Sistema dice: Personas
   [Acuerdo] -> FALLO
```

```
Img: Comidas -> Sistema dice: Personas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

Caso de Prueba 4:

- Precisión Global: 93.75%
- Conflictos entre KNN y MVN: 5 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

Caso de Prueba 5:

- Precisión Global: 87.50%
- Conflictos entre KNN y MVN: 6 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

Caso de Prueba 6:

- Precisión Global: 87.50%
- Conflictos entre KNN y MVN: 6 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Comidas -> Sistema dice: Personas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

Caso de Prueba 7:

- Precisión Global: 90.62%
- Conflictos entre KNN y MVN: 1 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

Caso de Prueba 8:

- Precisión Global: 84.38%
- Conflictos entre KNN y MVN: 3 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Conflicto -> Gana KNN] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

```
Img: Comidas -> Sistema dice: Personas
   [Acuerdo] -> FALLO
```

```
Img: Comidas -> Sistema dice: Personas
   [Acuerdo] -> FALLO
```

Caso de Prueba 9:

- Precisión Global: 90.62%
- Conflictos entre KNN y MVN: 2 de 32 casos.

```
Img: Comidas -> Sistema dice: Personas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

Caso de Prueba 10:

- Precisión Global: 93.75%
- Conflictos entre KNN y MVN: 6 de 32 casos.

```
Img: Personas -> Sistema dice: Comidas
   [Acuerdo] -> FALLO
```

```
Img: Comidas -> Sistema dice: Personas
   [Conflicto -> Gana KNN] -> FALLO
```

En la Tercera versión del sistema programado:

- Se cogen en total 170 imágenes, 110 de comida y 60 de persona, para el entrenamiento se cogen el 80% de las imágenes de forma aleatoria y el resto se usan para el testing (33 casos).

Caso de prueba 1:

- Precisión Global: 100.00%
- Conflictos entre KNN y centroides: 9 de 33 casos.

Caso de prueba 2:

- Precisión Global: 96.97%
- Conflictos entre KNN y centroides: 10 de 33 casos.

```
Img real: Comidas -> Pred: Personas
   [Conflicto -> Gana MVN] -> FALLO
```

Caso de prueba 3:

- Precisión Global: 96.97%
- Conflictos entre KNN y centroides: 11 de 33 casos.

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

Caso de prueba 4:

- Precisión Global: 96.97%
- Conflictos entre KNN y centroides: 11 de 33 casos.

```
Img real: Comidas -> Pred: Personas
   [Conflicto -> Gana MVN] -> FALLO
```

Caso de prueba 5:

- Precisión Global: 87.88%
- Conflictos entre KNN y centroides: 7 de 33 casos.

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

Caso de prueba 6:

- Precisión Global: 96.97%
- Conflictos entre KNN y centroides: 10 de 33 casos.

```
Img real: Comidas -> Pred: Personas
   [Conflicto -> Gana MVN] -> FALLO
```

Caso de prueba 7:

- Precisión Global: 96.97%
- Conflictos entre KNN y centroides: 10 de 33 casos.

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

Caso de prueba 8:

- Precisión Global: 93.94%
- Conflictos entre KNN y centroides: 9 de 33 casos.

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

```
Img real: Personas -> Pred: Comidas
   [Acuerdo] -> FALLO
```

Caso de prueba 9:

- Precisión Global: 100.00%
- Conflictos entre KNN y centroides: 10 de 33 casos.

```
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Acuerdo] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Personas -> Pred: Personas
   [Conflicto -> Gana MVN] -> CORRECTO
```

```
--------------------
Img real: Comidas -> Pred: Comidas
   [Acuerdo] -> CORRECTO
```