|  |  |  |  |
| --- | --- | --- | --- |
| **Minería de datos**  **“Método OCR”** | | | |
| **T3** | **#8. Misael Gustavo Bonilla Pacheco**  **#17.Humberto Gutiérrez Escobar** | **8A** |  |

**Introducción**

Cuando se dispone de información en forma de documento impreso y se desea procesarla mediante un computador, existen dos opciones: una primera consistiría en introducirla a través del teclado, labor larga y tediosa. Otra posibilidad es automatizar esta operación por medio de un sistema de OCR (Optical Character Recognition) compuesto de un software y hardware adecuado que reduciría considerablemente el tiempo de entrada de datos.

El OCR es una tecnología que trata de emular la capacidad del ojo humano para reconocer objetos. Es un software que permite el reconocimiento óptico de los caracteres contenidos en una imagen, de forma que estos se vuelven comprensibles o reconocibles para una computadora.

En el presente documento se muestra el proceso para obtener características de un cierto grupo de imágenes que van desde el 0 al 9 y así reconocer que número es y ciertas características que lo define.

**Requerimientos**

* Equipo de computo
* Sistema Operativo Windows 7, 8, 8.1 o 10
* Python Anaconda versión 3.6
* Imágenes segmentadas a partir del código en Matlab descargado de <http://watermarkero.blogspot.mx/>

**Pasos del algoritmo OCR**

1. Obtener carpetas de los números del 0 al 9 que contiene imágenes segmentadas y binarizadas a partir del código de Matlab descargado de <http://watermarkero.blogspot.mx/>.
2. En Spyder, Python, se recorren las carpetas con las imágenes segmentadas que se obtuvieron previamente y se obtienen las siguientes características de cada carpeta de números:
   * Relación de 1’s entre el tamaño de la imagen



* + Relación entre el número de filas y columnas (tamaño)



* + Cantidad de 1’s en la columna intermedia



* + Cantidad de 1’s en un cuarto de columna



* + Cantidad de 1’s en tres cuartos de columna



* + Cantidad de 1’s en la fila de en medio



* + Cantidad de 1’s en un cuarto de fila



* + Cantidad de 1’s en tres cuartos de fila



* + Cantidad de cortes entre 0´s y 1´s en la fila de en medio



* + Cantidad de cortes entre 0´s y 1´s en ¼ de fila



* + Cantidad de cortes entre 0´s y 1´s en ¾ fila



* + Cantidad de cortes entre 0´s y 1´s en la columna de en medio



* + Cantidad de cortes entre 0´s y 1´s en ¼ de columna



* + Cantidad de cortes entre 0´s y 1´s en ¾ de columna



1. Las características que se obtuvieron se almacenan en una matriz, dichas características son escritas en un archivo de texto csv.

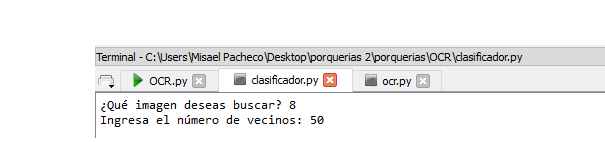
**KNN**

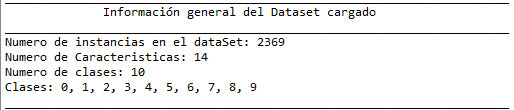
Después de obtener el archivo con la extensión *csv* se utiliza el método *knn* el cual facilita la búsqueda de vecinos más cercanos de cualquier imagen segmentada. Estos son los pasos para la utilización de este método.

1. Vaciar los datos del DataSet a una matriz.
2. Obtenemos las características de la imagen que vamos a identificar.
3. Se crea una nueva matriz la cual contendrá los datos de la imagen de entrada.
4. Se crea una nueva matriz donde se almacenara el número de vecinos más cercanos con la cual se van a comparar los datos con la matriz creada previamente.
5. Después de haber declarado las matrices, se obtienen filas y columnas de la imagen seleccionada.
6. La primera matriz declara se llenara con datos de menor importancia hasta llenar las 15 columnas, 14 de estas con las características que se necesitan para reconocer el número y una que identifica la clase a la que pertenece.
7. Lo mismo sucede con la segunda matriz, en esta matriz se almacena la posición, la distancia y la clase de la imagen.
8. A continuación se lee el archivo *csv,* previamente obtenido, y se almacena en la primera matriz con las 14 propiedades de cada imagen.
9. La segunda matriz de llena con la instancia obtenida del archivo *csv.*
10. Se realiza la formula euclidiana (el dato de la primer característica del dataset - el dato de la nueva instancia ^ 2 + la segunda, la tercera, hasta la característica 14) para obtener las distancias más cercanas del imagen comparándola con los características del DataSet y lo guardamos en una nueva matriz.
11. Finalmente regresamos el el dato encontrado para mostrar la posición, la distancia y la clase a la que pertenece la imagen que se seleccionó previamente.

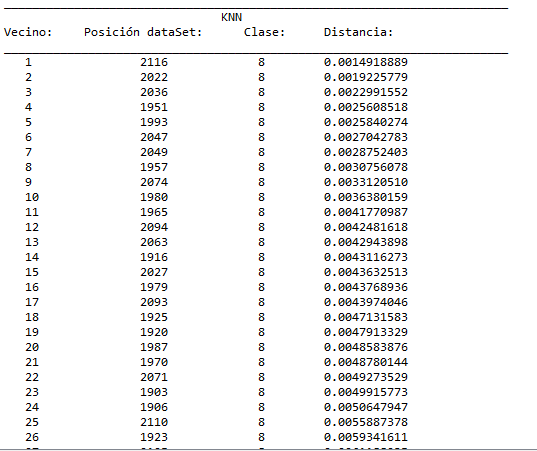
**Salida del programa**

El primer paso es ingresar la imagen que se desea analizar y el número de vecinos más cercanos que deseemos.

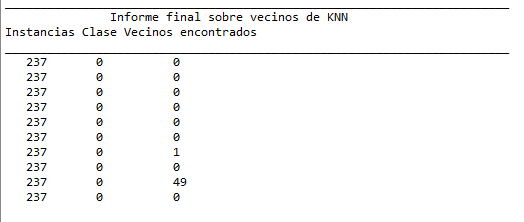
A continuación se muestra la información general del dataset seleccionado.



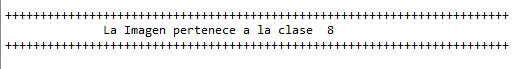
Después de ser procesada la imagen, el programa muestra la información con el número de vecinos, la posición donde se encuentra, la clase a la que pertenece y la distancia que tiene.



El programa muestra el informe final del proceso realizado con las isntancias, la clase y los vecinos encontrados.



Al final de todo el proceso el programa muestra la clase a la que pertenece la imagen que fue seleccionada previamente.



El código y los archivos que se necesitan para la implementación de este método se pueden encontrar en:

*Aqui\_va\_el\_link*