

Procesamiento de texto manuscrito usando conjuntos de clasificadores

Emilio Samuel Aced Fuentes

Roberto Alcober Couso

Arturo Blázquez Pérez

Nicolás Trejo Moya

29 de diciembre de 2018

1. Introducción

En este proyecto vamos a clasificar imágenes de letras manuscritas intentando predecir de forma correcta el símbolo que representan. Utilizaremos varios algoritmos de clasificación y los compararemos para encontrar los mejores resultados posibles, viendo diferencias de tiempos y tasa de acierto.

2. Análisis de los datos

En el dataset provisto, están representadas las 10 primeras letras del alfabeto (A-J), por tanto 10 clases las cuales hemos etiquetado en nuestra base de datos etiquetadas de 0 a 10 respectivamente. Las imágenes nos vienen en un tamaño de 206×150 en escala de grises.

Como podemos observar las imágenes contienen ruido, el cual podría dificultar la tarea de clasificación gravemente. Este problema lo hemos solucionado sometiendo a la imagen a diferentes tratamientos.

2.1. Tratamientos

Para todos los experimentos de este proyecto hemos umbralizado la imagen mediante otsu y posteriormente hemos realizado un filtro de mediana con un kernel cuadrado de tamaño 3. Esto nos ha dado el siguiente resultado:

A continuación explicaremos los tratamientos que le hemos aplicado a las imágenes para reducir la complejidad del problema

2.2. Atributos

Teniendo en cuenta que las imágenes son matrices 206×150 lo cual nos da un total de 30900 atributos, lo cual es una cantidad de atributos desorbitada, por ese motivo hemos decidido ver como afecta reducir la dimensión. Las principales modificaciones que realizaremos a las imágenes para reducir su dimensionalidad son las siguientes:

Figura 1: Ejemplo de una A

Figura 2: Ejemplo de una A tratada

Eliminar filas y columnas poco importantes Esto lo realizaremos mediante la selección de un umbral, con el cual consideraremos que las filas y columnas que tengan una cantidad menor de zona pintada que el umbral especificado no son de relevancia y por tanto las despreciaremos del problema

Figura 3: Ejemplo de una A recortada

Redimensionar la imagen interpolando Con el fin de evitar el aliasing (meter bordes que no existían al reducir una imagen) hemos redimensionado la imagen siempre aplicando un kernel gaussiano previamente.

A continuación mostraremos distintos valores de los tamaños de las imágenes que compararemos en nuestro estudio:

3. Modelos

Usaremos los siguientes clasificadores en este trabajo:

1. Random Forest
2. SVC
3. KNN

en los cuales variaremos los tamaños de las imágenes y compararemos tiempos y tasa de aciertos para poder decidirnos por uno.

Para todos ellos destinaremos la mitad de los datos para train y la otra mitad para test.

3.1. Random Forest

Figura 4: Varios tamaños de una A que usaremos