Detección de caracteres por medio de técnicas de análisis de imágenes

Tomás Higuera Viso[1], Miguel Antonio Nuñez Valle[2] y Álvaro Sánchez de Lucas[3]

Universidad Autónoma de Madrid - Escuela Politécnica Superior

Abstract. En este proyecto se nos pide reconocer todas las vocales, tanto mayúsculas como minúsculas a través de imágenes. Para abordar este proyecto, hemos hecho uso de técnicas que se utilizan en reconocimiento de imágenes, las cuales son la extracción de características mediante *hog* (histograma de gradientes orientados) y el método *bag of words* para crear modelos de las imágenes.
El clasificador que hemos utilizado es el *svm* (support vector machine), que creemos que es el más apropiado para este tipo de detección.

1 Introducción

Para la creación de nuestro modelo hemos hecho uso del conjunto de datos *letritas.zip*. La metodología que seguiremos para crear nuestro modelo será la siguiente: primero extraeremos características de las imágenes haciendo uso del método *hog*, a continuación crearemos un vocabulario a partir de las imágenes de entrenamiento. Una vez realizados estos pasos pasaremos a obtener histogramas que representen las imágenes mediante el método *bag of words*. Una vez obtenidas estas representaciones haremos uso del clasificador *svm* para predecir las etiquetas de las imágenes de test.
De los datos utilizados hemos hecho una partición simple, dedicando un 67% a train y un 33% a test.

2 Extracción de características

Para la extracción de características hemos hecho uso de de la técnica *hog*. Esta técnica consiste en crear un histograma de gradientes orientados a partir de cada imagen. Para la creación de estos histogramas no hay mas que tratar la imagen como un vector, hacer las derivadas parciales del mismo y de ahí obtener tanto magnitud como gradiente. Una vez obtenidas estas características pasaremos a realizar el histograma.
Al ser las imágenes en blanco y negro, nuestra tarea es mas sencilla ya que trataremos un único canal.
Para facilitar el análisis hemos creado carpetas que contienen cada tipo de letra y hemos cambiado el formato de las imágenes de tipo png a jpg, ya que las librerías que utilizamos de *skimage* tratan las imágenes solo en este formato.

2.1 Vocabulario para bag of words

Una vez obtenidos los histogramas de las imágenes, hemos creado un vocabulario a partir de los mismos, que no es mas que una concatencación de los distintos histogramas. Este vocabulario lo utilizaremos para que a la hora de evaluar cada imagen, podamos crear un histograma con las diferentes coincidencias de la imagen y el vocabulario descrito.

3 Bag of words

Una vez extraídas las características de las imágenes de entrenamiento, así como el vocabulario procedente de las mismas, pasamos a clasificar las distintas imágenes de test.
Para esta clasificación, lo primero que haremos será, mediante el método *bag of words*, crear un histograma de coincidencias entre las imágenes de test y las de entrenamiento, haciendo uso del vocabulario previamente creado.
Esta búsqueda de coincidencias se hará buscando la menor diferencia entre el histograma de gradientes orientados de la imagen de test con los del vocabulario y realizando una ponderación para obtener el histograma.
Este *bag of words* se realizara también con las imágenes de entrenamiento para que a la hora de hacer uso del clasificador *svm* podamos crear las diferentes regiones para etiquetar las imágenes.

4 Clasificador svm

Una vez obtenidos los *bag of words* de las imágenes de test y de entrenamiento, pasaremos a crear nuestro clasificador. El tipo de clasificador elegido es *svm* (support vector machine), que dado un conjunto de datos de entrenamiento podemos crear una serie de vectores que separen las diferentes etiquetas que queremos diferenciar, en nuestro caso las letras vocales tanto mayúsculas como minúsculas.

4.1 Desarrollo clasificador

Para la creación de este clasificador hemos hecho uso de la librería de python *opencv*. Las diferentes características con las que hemos definido este clasificador son las siguientes:
- El valor de SVM type lo hemos establecido a C_SVC, que permite la separación imperfecta de clases, esto es, que los vectores que se generaran para los diferentes tipos de clases, no necesariamente contendrán todos los valores de entrenamiento con la etiqueta correspondiente a ese área, sino que podrá haber valores situados en zonas en las que no corresponda. Esto lo hacemos porque al no haber generado nosotros los datos, contamos con que haya letras escrita de manera muy diferente a otras, y no queremos que estas afecten notablemente a nuestros resultados.
- El valor del Kernel lo hemos establecido a CHI2. Hemos escogido este tipo

de kernel, ya que tras buscar información hemos llegado a la conclusión de que este es el mas usado a la hora de tratar imágenes. A pesar de haber seleccionado este, también hicimos pruebas con el kernel RVF, que también se utiliza en el reconocimiento de imágenes y hemos obtenido valores peores, por lo que finalmente escogimos el CHI2.

5 Análisis de resultados

Tras realizar el clasificador obtenemos una precisión del 80%. Creemos que es una precisión bastante buena para un clasificador tan sencillo.
La mayor parte de los errores que se producen en el etiquetado se deben a la confusión de las letras *o* y las letras *a*. Esto se debe a que al usar *hog* como extractor de características, las diferencias entre ambas imágenes son mínimas. Creemos que a pesar de usar otro tipo de análisis de la imagen, los errores obtenidos serían similares a los nuestros, ya que este pequeño detalle es muy difícil diferenciar en gran parte de los casos.

References

[1] Libreria scikit-learn, https://scikit-image.org

[2] Moodle de la asignatura, https://moodle.uam.es

[3] Método bag of words, https://machinelearningmastery.com/gentle-introduction-bag-words-model

[4] Extractor de características de hog, https://www.learnopencv.com/histogram-of-oriented-gradients/