Extractor de subtítulos incrustados: Trabajo Final de Procesamiento Digital de Imágenes

Mariángeles Paez, Leonardo N. Arato y Darién J. Ramírez
Trabajo práctico final de "Procesamiento Digital de imágenes", II-FICH-UNL.

***Resumen*— En el presente trabajo se propone un método automático de detección y extracción de subtítulos incrustados en videos utilizando técnicas de procesamiento digital de imágenes, las cuales brindan herramientas que hacen posible el tratamiento avanzado de los fotogramas que componen al video. Las imágenes procesadas son pasadas además por un reconocedor óptico de caracteres (OCR).**

***Palabras clave*— extractor, subtítulos, SubRip, tesseract, OCR.**

I. Introducción

SE vuelve cotidiano encontrar videos con subtítulos que no se pueden desactivar debido a que se encuentran incrustados en los fotogramas. Esta clase de subtítulos forman parte de la imagen y no se encuentran en formato texto. Por lo tanto, no pueden ser editados en caso de errores de escritura y no pueden ser utilizados en otros videos del mismo contenido pero de mejor calidad, dando lugar al trabajo tedioso de aproximar los tiempos manualmente para crear nuevos archivos *srt*.

El propósito de este trabajo es la producción de un método automático de extracción de subtítulos incrustados en videos, realizar la conversión de imagen a texto, calculando los tiempos en los que aparecen en el video, aplicando un *OCR* y generando un archivo en formato SubRip.

Para ello se realiza el tratamiento del video fotograma a fotograma. Se observa la región ubicada en la parte inferior del fotograma en una proporción de $\frac{1}{4}$ de la altura del mismo que es donde normalmente aparecen.

A partir de la región de interés especificada se utilizan técnicas de filtrado, binarizado, morfología y detección de líneas para encontrar letras presentes en cada fotograma.

Para la conversión de los subtítulos de formato imagen a texto se utiliza el reconocedor óptico de carateres *tesseract OCR*

Las herramientas que suministra opencv permiten obtener los tiempos en milisegundos. El procesamiento se realiza en dicha unidad pero luego son adaptados al formato en que deben aparecer en los archivos srt.

Finalmente se utilizan las medidas de solapamiento *Dice* (F1-Score) y *Jaccard* (Intersección sobre unión) para evaluar la calidad de la detección de los subtítulos.

II. Metodología

A. Detección de subtítulos

Se trabaja con un fotograma a la vez, se selecciona como área de trabajo la región inferior correspondiente al 25 % de la altura de la imagen, de manera que se puedan utilizar videos de cualquier resolución, teniendo en cuenta subtítulos incrustados en esa región.

Se convierte la imagen desde el modelo de color BGR al modelo HSV, se separa en canales, donde la componente de *valor* ofrece los mejores resultados en la detección en base a las pruebas realizadas anteriormente, para decidir qué modelo convenía utilizar.

Como el método no se basa en segmentación por color, ya que puede haber subtítulos con diferentes colores, si no en la detección de bordes, el fondo influye en la detección por lo que se aplica un filtro gaussiano que introduce un leve borroneado y homogeneiza la imagen. Luego se aplica un filtro pasa-altos de suma cero con énfasis en los bordes horizontales:

$$\begin{pmatrix} -1 & -2 & -1 \\ -1 & 10 & -1 \\ -1 & -2 & -1 \end{pmatrix}$$

Se limpian los contornos resultantes de la convolución y el resultado es posteriormente binarizado, siendo el umbral dependiente de la resolución del fotograma.

Para desaparecer posibles pixeles huérfanos se aplica un filtro de mediana pero previamente se aplica la operación morfológica de cierre para rellenar las letras obtenidas y evitar que el filtrado sea destructivo. Luego se emplea una apertura para reducir posibles bordes de tendencia vertical que hayan quedado y se dilata horizontalmente contemplando la presencia de como mínimo dos caracteres en preparación para el siguiente paso.

Ahora se pasa a detectar la presencia de caracteres en la región de trabajo. Para ello se utiliza la transformada de Hough para encontrar líneas horizontales que contengan varios puntos en la región. La existencia de líneas que cumplan con esa característica es la prueba de que existe texto en ese fotograma. En la figura 1 se pueden apreciar los resultados intermedios del proceso.

B. Confección del archivo SubRip

Los archivos SubRip tiene la extensión .srt y contienen texto plano con formato. Su estructura es la siguiente:

- Número de subtítulo (En orden secuencial empezando en 1)
- Tiempo inicial −−> Tiempo final (En formato horas:minutos:segundos,milisegundos)
- Texto del subtítulo (Puede incluir una o varias líneas separadas por un salto de línea)
- Línea en blanco.

Para la confección de cada subtítulo se debe tener en cuenta el momento en que aparece uno nuevo, la cantidad de fotogramas consecutivos en los que aparece y el momento en que desaparece.

Fig. 1. Pasos intermedios realizados en la detección de subtítulos.

Se trata de manera diferenciada el caso de fotogramas consecutivos que contienen subtítulos diferentes. Donde se finaliza el tiempo de un subtítulo para iniciar el tiempo de aparición del siguiente.

La igualdad de los subtítulos en los diferentes fotogramas se establece con el uso de la función de correlación, con un porcentaje de similaridad del 90 %.

C. *Utilización del OCR*

Para la conversión de imagen a texto se utiliza el motor OCR *tesseract*. Pero no se aplica directamente a cada fotograma que contiene subtítulos. Se realiza un promediado de las imágenes donde los subtítulos son iguales, de manera que en el resultado se destaca el texto y se pierden detalles del fondo. Luego, se aplica una transformación de potencia para realzar los caracteres encontrados de intensidades altas y oscurecer el fondo (Figura 2).

Por último, la imagen promedio se utiliza en la función de reconocimiento de caracteres, para obtener el texto *string*. El diccionario aplicado es el español y además se aplica una lista blanca en el reconocimiento para evitar encontrar caracteres indeseados que son poco probables en texto de esta índole.

Lista blanca:
a b c d e f g h i j k l m n ñ o p q r s t u v w x y z A B C D E F G H I J K L M N Ñ O P Q R S T U V W X Y Z á é í ó ú Á É Í Ó Ú ü Ü . , ¡ ! ¿ ?

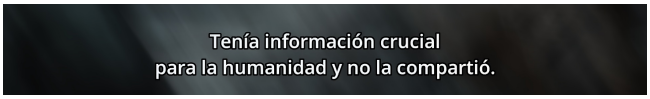

Fig. 2. Promedio y transformación de fotogramas con el mismo texto.

D. *Medidas de calidad de detección*

Para obtener una medida objetiva de la calidad de detección del algoritmo, coeficientes *Dice* (F1-Score) y *Jaccard* (intersección sobre la unión). Para calcularlos se creó una imagen que simula una línea de tiempo diferenciando bien la presencia (blanco) o falta (negro) de subtítulos en ella (predicción). De la misma forma, se creó una imagen con la misma característica, pero utilizando los subtítulos originales junto a sus tiempos (solución).

Con la obtención de estas imágenes se procedió a calcular el *Dice* y *Jaccard* de la siguiente manera:

$$Dice = \frac{2|solucion \cap prediccion|}{|solucion| + |prediccion|}$$

$$Jaccard = \frac{|solucion \cap prediccion|}{|solucion \cup prediccion|}$$

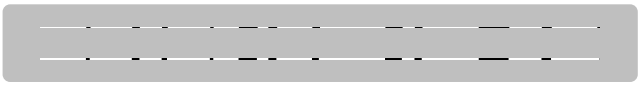

Fig. 3. Ejemplo de predicción y solución respectivamente.

III. Resultados

En la tabla I se muestran los resultados obtenidos con tres videos distintos, en distintas resoluciones y muestreados en fragmentos de 60 segundos.

Archivo	Resolución	Dice	Jaccard
snk[1]	720 (HD)	0,980735	0,962477
snk[11]	720 (HD)	0,994318	0,988724
snk[20]	720 (HD)	0,889938	0,805134
snk[1]	1080 (FullHD)	0,979918	0,960946
snk[11]	1080 (FullHD)	0,994318	0,988724
snk[20]	1080 (FullHD)	0,889938	0,805134
kf[6]	720 (HD)	0,981076	0,963177
kf[10]	720 (HD)	0.948524	0,903263
kf[12]	720 (HD)	0,953312	0,912434
kf[6]	1080 (FullHD)	0,982348	0,96558
kf[10]	1080 (FullHD)	0,961848	0,92728
kf[12]	1080 (FullHD)	0,963206	0,929911
y[1]	480 (SD NTSC)	0,974711	0,951023
y[6]	480 (SD NTSC)	0,990482	0,981217
y[20]	480 (SD NTSC)	0,984751	0,970172

TABLA I
RESULTADOS.

Fig. 4. Parte superior: OCR. Parte inferior: original.

IV. Conclusiones

Como se puede observar los resultados son dependientes de la resolución del archivo de video. Esto se debe a que el filtrado para la detección de bordes no utiliza un *kernel* proporcional al tamaño del fotograma. En su lugar se ajusta el valor del umbral mediante una función lineal en base a las dimensiones de la imagen. Este método está acotado a videos con resoluciones que van desde la definición estándar (SD) de la norma NTSC, 640×480 hasta la resolución *FullHD*, 1920×1080. El hecho de utilizar una función para determinar el valor del umbral no es un criterio fuerte pero funciona relativamente bien. Se realizaron pruebas con el algoritmo de *Otsu* pero resultaba imposible eliminar los bordes y se tenía una amplia cantidad de falsos positivos. Una posible mejora para determinar el umbral podría ser entrenar un perceptrón simple ya que el problema es linealmente separable, utilizando como entrada alguna característica que tenga relación con el umbral como por ejemplo la energía de la imagen de bordes y como salida el valor del umbral. Esto implica conseguir de alguna fuente esos datos o realizar la minería de datos pertinente.

Se utilizaron distintos métodos para la detección de bordes pero los de derivada segunda resultaron más útiles ya que permitían la fácil eliminación de los bordes mientras que los de derivada primera engrosaban mucho los bordes dificultando dicha tarea. *Canny* no otorgó buenos resultados.

Los resultados obtenidos son muy buenos, por lo general superiores al 90 %. El coeficiente *Jaccard* es normalmente inferior al *Dice* puesto que penaliza más la presencia de falsos positivos que dan su aparición en este método de detección. Un falso positivo aquí representa que se encontró un subtítulo donde en realidad no había debido a que el procesamiento del fotograma no llega a eliminar en algunos casos todo los bordes correspondientes al fondo. Sin embargo, esto no es tan grave como el caso de los falsos negativos (cuando existe subtítulo pero no se detecta) y con las pruebas realizadas parece ser que el método detecta correctamente o detecta de más pero no lo hace de menos, es decir, no hay falsos negativos. La razón principal de que no sea un problema es que al aplicar el *OCR*, este probablemente no retorne ningún texto a causa de que no hay carácter que se corresponda con lo que queda en el fotograma procesado. Por otro lado, se da también que los bordes en ocasiones poseen formas similares a algunas letras pudiendo ser detectadas como tal y pasadas a texto.

Los resultados obtenidos solo analizan la parte correspondiente a la detección de los subtítulos y dejan de lado que tan bien hace su tarea el *OCR*, el cual en algunos casos funciona muy bien mientras que en otros se aleja mucho de lo que debería obtener. Para mejorar esto se debería indagar más en el funcionamiento del *tesseract* y realizar un preprocesamiento mejor al efectuado de tal manera que la conversión de imagen a texto mejore y se pueda comparar correctamente.

Una de las contrapartes es el tiempo necesario para el procesamiento. La razón de la utilización de los archivos de 60 segundos se debe principalmente a esto. Hay que considerar que un video de ese tiempo con 24 fotogramas por segundo cuenta con aproximadamente 1440 fotogramas. El tiempo de procesamiento aumenta conforme se incrementa la resolución, por lo tanto, a futuro se debería encontrar alguna forma de optimización. Pueden variar y requieren de un análisis elaborado, por ejemplo, remuestrear en resoluciones más pequeñas para el procesamiento, evitar realizar demasiadas iteraciones en las operaciones morfológicas o directamente evitarlas, o incluir aspectos que combinen algo de segmentación por color con el método actual basado en la detección de bordes.

Referencias

[1] Rafael C. González and Richard E.Woods, *Digital Image Processing*. Prentice-Hall, 3rd. ed. (2008) - 2nd. ed. (2001).
[2] Enzo Ferrante, *Medidas De Calidad Segmentacion.* (2018).
[3] Tesseract OCR, *https://github.com/tesseract-ocr*