



Ciencias de la
Computación
FACULTAD DE CIENCIAS
FÍSICAS Y MATEMÁTICAS
UNIVERSIDAD DE CHILE

Tarea 2 - Comerciales

CC5204 - Búsqueda por contenido de Imágenes y Videos

Alumnos: Patricio Isbej
Elisa Kauffmann
Profesor: Juan Manuel Barrios
Ayudante: Sebastián Ferrada
Fecha: 2 de mayo de 2016

Índice

1. Introducción	3
2. Resumen	3
3. Hipótesis	3
4. Diseño e implementación	3
4.1. Descripción del contenido	3
4.2. Búsqueda por similitud	4
4.3. Detección	6
5. Experimentos y resultados	6
6. Análisis y conclusiones	6
7. Anexos	6

Índice de figuras

Índice de tablas

1. Bosquejo del diccionario creado para la configuración de 1 descriptor cada 10 frames.	5
2. Bosquejo de la matriz de identificadores y descriptores de los comerciales	6
3. My caption	7
4. My caption	7
5. My caption	8

1. Introducción

2. Resumen

3. Hipótesis

4. Diseño e implementación

Se implementa un detector de comerciales que señala cuántas veces fue emitido cada comercial durante las 4 horas de programación, determinando exactamente el inicio y fin de cada aparición. Para ello se divide el trabajo en 3 pasos:

1. Descripción del contenido (extracción de características)
2. Búsqueda por similitud (comparación de descriptores)
3. Detección (encontrar secuencias parecidas)

Se usa el lenguaje **Python** utilizando las librerías **OpenCV** para procesar imágenes y videos, **numpy** para manejo de vectores y matrices y **matplotlib** para graficar.

El reporte de detecciones se guarda en un archivo **.m3u** que por cada aparición de un comercial, contiene 3 líneas con el siguiente formato:

```
#EXTVLCOPT:start-time= [seconds]
#EXTVLCOPT:stop-time= [seconds]
.../.../base/mega-2014_04_25T22_00_07.mp4
```

donde la primera y la segunda línea indican el momento en que empieza y termina el comercial respectivamente (en segundos), y la tercera línea indica el video a reproducir, en este caso es el programa de 4 horas.

4.1. Descripción del contenido

A rasgos generales, en esta etapa se utiliza el descriptor de **histogramas por zonas** para caracterizar los frames de los videos guardando el descriptor de cada

video en un archivo `.npy`, el cual será utilizado en el paso siguiente. Es importante mencionar que los frames de todos los videos son escalados al mismo tamaño (720×400 px) para normalizar y facilitar la división en zonas. Además, previo al desarrollo de la tarea se verificó que todos los videos tuvieran el mismo *framerate* para mantener consistencia (~ 29.97 fps).

En esta etapa, los parámetros que varían en cada configuración son:

- frecuencia de muestreo de frames
- cantidad de zonas en las que se divide cada frame
- cantidad de bins de los histogramas

En detalle, el programa ejecuta los siguientes pasos:

1. Para cada archivo de video, se toma un frame cada cierta frecuencia, lo escala a un tamaño de 720×400 px y lo transforma a escala de grises.
2. Se divide cada uno de estos frames en la cantidad de zonas especificada por el tipo de configuración, y usando la función `calcHist` de `OpenCV` se obtiene un histograma por cada zona en forma de un arreglo.
3. Se crea el descriptor completo del frame: un arreglo que es la concatenación de los vectores de los histogramas obtenidos en el paso anterior.
4. Se crea el descriptor del video completo construyendo un arreglo `numpy` de todos los arreglos que describen cada frame del video.

El código se encuentra en el archivo `describe.py`

4.2. Búsqueda por similitud

Durante esta etapa, usando los descriptores creados anteriormente, por cada frame del video de 4 horas se encuentra el frame del comercial que más se le parece. Para ello se comparan todos los vectores descriptores de cada comercial contra cada frame del video de 4 horas, utilizando distancia Manhattan. Para facilitar el procesamiento se crea un diccionario que guarda, por cada comercial, su título, un número identificador y la cantidad de frames procesados. Un ejemplo de uno de

Id	Nombre	Frames
0	mr big (2).npy	95
1	scotiabank.npy	76
2	ripley dias r.npy	63
3	sodimac homecenter.npy	62
4	donnasept.npy	61
5	dove desodorante hombre.npy	45
6	entel 4g corto.npy	81
7	johnson mundial.npy	60
8	limon soda.npy	96
9	bilz y pap.npy	105

Cuadro 1: Bosquejo del diccionario creado para la configuración de 1 descriptor cada 10 frames.

los diccionarios creados se puede observar en el cuadro 1.

Específicamente, el programa realiza los siguientes pasos:

1. Se crea un diccionario que mapea los títulos de los comerciales a un identificador y a la cantidad de frames obtenidos en la etapa anterior.
2. Se cargan en memoria los descriptores de video de todos los 10 comerciales y usando el diccionario se construye una matriz de dimensiones 10×3 . La matriz contiene, para cada comercial, una fila con su identificador, su título y su descriptor. (Ver cuadro 2).
3. Se carga el descriptor del programa de 4 horas y se compara cada frame contra todos los frames de los 10 comerciales. Para medir la similitud entre vectores descriptores se utiliza distancia Manhattan (utilizando la función `norm` de `OpenCV` con el parámetro `NORM_L1`). Así se va construyendo un vector de “calces” con tantas filas como frames del programa de 4 horas procesados. Cada fila contiene el índice del frame y el identificador del comercial que más se parece al frame del programa.
4. El vector de “calces” se guarda en un archivo `.npy` para ser usado en el paso siguiente

com_id	nombre archivo	descriptor comercial
0	mr big (2).npy	$[[[], [], \dots, [], []]]$
1	scotiabank.npy	$[[[], [], \dots, [], []]]$
2	ripley dias r.npy	$[[[], [], \dots, [], []]]$
3	sodimac homecenter.npy	$[[[], [], \dots, [], []]]$
4	donnasept.npy	$[[[], [], \dots, [], []]]$
5	dove desodorante hombre.npy	$[[[], [], \dots, [], []]]$
6	entel 4g corto.npy	$[[[], [], \dots, [], []]]$
7	johnson mundial.npy	$[[[], [], \dots, [], []]]$
8	limon soda.npy	$[[[], [], \dots, [], []]]$
9	bilz y pap.npy	$[[[], [], \dots, [], []]]$

Cuadro 2: Bosquejo de la matriz de identificadores y descriptores de los comerciales

El código se encuentra en el archivo `compare.py`

4.3. Detección

En esta etapa se detecta la aparición de los comerciales en el video de 4 horas haciendo uso del diccionario y el vector de “calces” creados en las etapas anteriores. La idea general es usar máscaras, una por cada comercial, para ir recorriendo el vector de “calces” y comparar segmentos de éste con las máscaras usando distancia de Hamming. La idea se ilustra en la figura 5.

Detalladamente, la implementación lleva a cabo los siguientes pasos:

5. Experimentos y resultados

6. Análisis y conclusiones

7. Anexos

Cuadro 3: My caption

Division	Frecuencia	30		20		10	
	Bins\Tolerancia	50 %	80 %	50 %	80 %	50 %	80 %
1x1	16	0.75	0.98	0.81	1	0.84	1
	32	0.75	1	0.78	1	0.89	1
	64	0.78	1	0.81	1	0.89	1
2x2	16	0.86	1	0.94	1	0.94	1
	32	0.89	1	0.86	1	0.94	1
	64	0.89	0.98	0.91	1	0.94	1
4x4	16	0.91	0.96	0.89	1	0.96	1
	32	0.91	0.96	0.86	1	0.94	1
	64	0.91	0.94	0.89	1	0.96	1

Cuadro 4: My caption

Division	Frecuencia	30		20		10	
	Bins\Tolerancia	50 %	80 %	50 %	80 %	50 %	80 %
1x1	16	15/0/10	24/0/1	17/0/8	25/0/0	18/0/7	25/0/0
	32	15/0/10	25/0/0	16/0/9	25/0/0	20/0/5	25/0/0
	64	16/0/9	25/0/0	20/0/5	25/0/0	20/0/5	25/0/0
2x2	16	19/0/6	25/0/0	22/0/3	25/0/0	22/0/3	25/0/0
	32	20/0/5	25/0/0	19/0/6	25/0/0	22/0/3	25/0/0
	64	20/0/5	25/1/0	21/0/4	25/0/0	22/0/3	25/0/0
4x4	16	21/0/4	25/2/0	20/0/5	25/0/0	23/0/2	25/0/0
	32	21/0/4	25/2/0	19/0/6	25/0/0	22/0/3	25/0/0
	64	21/0/4	25/3/0	20/0/5	25/0/0	23/0/2	25/0/0

Cuadro 5: My caption

Frecuencia		30		20		10	
Division	Bins\Tolerancia	50 %	80 %	50 %	80 %	50 %	80 %
1x1	16	15/0/10	24/0/1	17/0/8	25/0/0	18/0/7	25/0/0
	32	15/0/10	25/0/0	16/0/9	25/0/0	20/0/5	25/0/0
	64	16/0/9	25/0/0	20/0/5	25/0/0	20/0/5	25/0/0
2x2	16	19/0/6	25/0/0	22/0/3	25/0/0	22/0/3	25/0/0
	32	20/0/5	25/0/0	19/0/6	25/0/0	22/0/3	25/0/0
	64	20/0/5	25/1/0	21/0/4	25/0/0	22/0/3	25/0/0
4x4	16	21/0/4	25/2/0	20/0/5	25/0/0	23/0/2	25/0/0
	32	21/0/4	25/2/0	19/0/6	25/0/0	22/0/3	25/0/0
	64	21/0/4	25/3/0	20/0/5	25/0/0	23/0/2	25/0/0

Cuadro 6: My caption

2 0
2 1
2 2
2 3
2 4
2 5
2 6
2 7
2 8
2 9
2 10
2 11
2 12
2 13
2 14

Cuadro 7: My caption

8	9
4	16
5	29
5	29
5	29
4	16
4	16
2	1
2	5
2	3
2	3
2	5
2	5
2	7
2	7
2	8
2	9
2	11
2	11
2	12
2	14
2	14
4	6
4	6
4	6
4	6
6	23
4	15
4	7
5	25
2	0
4	3