PHOW Pyramid Histogram of visual Words

José María Campo Viñas
Universidad de los Andes
Código: 201412002
jm.campo11@uniandes.edu.co

Mariana Franky
Universidad de los Andes
Código: 201313944
m.franky10@uniandes.edu.co

Abstract

En este laboratorio se exploró PHOW como estrategía para el reconocimiento de palabras. Se utilizó la Biblioteca VLFeat desarrollado por Andrea Vedaldi y Brian Fulkerson, se utilizó como base de datos Caltech_101 y ImageNet. Se realizaron varios modelos cambiando parámetros como: número de palabras, valor de confianza y cantidad de particiones, con el objetivo de encontrar los mejores valores de estos y asi obtener un mayor ACA en la base de datos de ImageNet. Inicialmente se obtuvo un ACA, empleando los parámetros por defecto, de 12.75 %, finalmente se obtuvo un ACA de 19.80 % al optimizar dichos parámetros.

1.. Introducción

Durante las últimas décadas, se han desarrollado varios modelos para extraer características visuales de imágenes para tareas de reconocimiento de objetos. Los modelos son variados en varios aspectos: aquellos que son entrenados por supervisión, otros entrenados sin supervisión y modelos (por ejemplo, extractores de funciones) que están completamente cableados y no necesitan entrenamiento. Algunos de los modelos vienen con una estructura jerárquica profunda que consta de varias capas, y algunos otros son superficiales y vienen con sólo una o dos capas de procesamiento [1]. En éste laboratorio se utilizó pirámide histograma de palabras visuales (PHOW) como método para la extracción de características.

La pirámide histograma de las palabras visuales (PHOW) es una extensión del modelo bag-of words (BoW), en el cual las características de la imagen se tratan como palabras. Se ha aplicado al reconocimiento de imágenes en visión por ordenador. PHOW utiliza BOW como base donde una bolsa de palabras (características de la imagen) es un vector de frecuencias de las palabras repetidas en una imagen, el problema que presenta BOW es que se sabe que una característica particular existe en la imagen y además, se sabe con qué frecuencia , pero no se tiene certeza dónde está esa característica dentro de la imagen. Este problema lo aborda PHOW en donde divide la imagen en sub-regiones cada vez más finas, a las que llaman pirámides, calculando el histograma de palabras visuales en cada subregión local [1].

Para calcular las características de PHOW, se puede calcular los descriptores de SIFT para cada una de las imágenes de entrenamiento y luego cuantificarlo usando K-means lo que genera un vocabulario visual. El histograma de palabras visuales SIFT que se calcula para cada bin forma las características de PHOW [1].

2.. Métodos y materiales

2.1.. Materiales

Se utilizaron dos bases de datos que han sido esenciales en el campo de reconocimiento de objetos:

- **Caltech 101:** Esta base de datos contiene fotos de objetos pertenecientes a 101 categorias. Aproximadamente de 30 a 300 imágenes por categoría. La base de datos fue recogida en septiembre de 2003 por Fei-Fei Li, Marco Andreetto y Marc Aurelio Ranzato. El tamaño de cada imagen es aproximadamente 300x200 píxeles.
- **ImageNet:** Es una base de datos organizada de acuerdo con la jerarquía de WordNet (actualmente sólo los sustantivos), en la que cada nodo de la jerarquía está representado por cientos y miles de imágenes. Esta base de datos contiene alrededor de 14,192,122 imágenes, y 21841 categorias.

2.2.. Métodos

Se utilizó la Biblioteca VLFeat desarrollada por Andrea Vedaldi y Brian Fulkerson, de esta libreria se utilizó como base el script phow_caltech101 con el fin de explorar el método PHOW.

A continuación se resumirá el método de PHOW en la librería VlFeat.

1. Se descarga la base de datos de caltech101. Cabe destacar que se ajustó el código para descargar la base de datos de ImageNet.
2. Se entrena el vocabulario, para esto computa SIFT en las imágenes de entrenamiento, luego construye el diccionario visual de palabras con K-means donde cada centroide corresponde a las palabras visuales.
3. Se calcula la pirámide de histogramas de palabras visuales para cada imagen de entrenamiento.
4. Entrena un SVM para cada categoria.
5. En la sección de Test SVM y evaluar: Calcula la pirámide de histogramas de palabras visuales para cada imagen de prueba. Luego, se evalua todos los SVMs y selecciona la categoría que corresponda a la mayor confianza,

Hiperparámetros Los hiperparámetros (nombre en Cuadro y valor por defecto) de entrada para el algoritmo escogidos debido a su importancia fueron: número de palabras (numWords = 600), valor de confianza en la clasificación de SVM (svm-C = 10), cantidad de particiones realizadas en la pirámide de clasificación (numSpatialXY = [2 4]) y número de imágenes usadas para el entrenamiento y test (numImágenes = 10). El hiperparámetro de tamaño de imagen (tamaño = 480), a pesar de no ser declarado en la estructura de configuración, también se tuvo en cuenta.

El número de palabras se considera que influye en el resultado de clasificación final, ya que éste determina la cantidad de características a tener en cuenta en las imágenes, si se escoge un número pequeño no habrán características suficientes para clasificar las 200 categorías, mientras que, al escoger un número muy grande el programa puede comenzar a tener en cuenta rasgos considerados ruido y que pueden existir en las demás categorías.
Por otra parte, el valor de confianza determina cúan permisivo será el corte del hiperplano en los valores de entrada, si se escoge una confianza muy baja, el hiperplano tendrá en cuenta valores que pueden ser ruido y tratará de acomodarse a ellos, así se encuentren junto a los datos de otro conjunto, pero si se escoge una confianza muy alta, el hiperplano será demasiado simple como para clasificar las clases y será muy susceptible al ruido en los valores de prueba.
La cantidad de particiones realizadas en la pirámide de clasificación dependen del objeto a clasificar en la imagen, si el objeto es demasiado pequeño, se necesitarán muchas particiones para que el algoritmo de SIFT genere una ventana adecuada a sus características, pero pocas particiones no darán las características necesarias para su clasificación. En casos de objetos grandes, ocurrirá los efectos contrarios a los objetos pequeños.
El número de imágenes empleadas tanto para entrenamiento como prueba, influyen en el resultado de clasificación, como se ha venido presentando en informes pasados, una mayor cantidad de imágenes en entrenamiento, en teoría conllevarán a una mejor clasificación de datos.
Por último, el tamaño de imagen corresponde a la cantidad de información a evaluar en cada partición realizada por la pirámide, entre mayor sea el tamaño de una imagen, mayor será la cantidad de información a computar con SIFT.

Se optó por cambiar cada hiperparámetro a valores menores y mayores cercanos a la magnitud respectiva, de manera que se pudiese denotar si ocurría underfitting o overfitting en los datos.
Finalmente, teniendo para cada modelo un ACA, se realizó una función con el objetivo de escoger los mejores hiperparámetros y asi obtener un mejor ACA en la base de datos ImageNet.

3.. Resultados

Se variaron los hiperparámetros mencionados anteriormente para las bases de datos de Caltech-101 e ImageNet200, los resultados de estos se muestran en el Cuadro a continuación.

Cuadro 1. ACA en distintos hiperparámetros

Hiperparámetros	Caltech-101 (ACA)	ImageNet (ACA)
Valores defecto	67.45 %	12.75 %
numWords = 500	67.52 %	12.75 %
numWords = 700		12.95 %
numWords = 800	68.30 %	
svm-C = 0.1	67.45 %	12.15 %
svm-C = 100	67.71 %	12.75 %
tamaño = 430	67.45 %	12.75 %
tamaño = 530	67.45 %	12.75 %
numSpatialXY = [2 4 6 8]	65.56 %	11.80 %
numImágenes = 25		18.38 %

Se procede a volver a correr el algoritmo con la base de datos de imageNet200, ésta vez con los mejores parámetros dados en el Cuadro 1. Se obtuvo un ACA de 19.80 %.

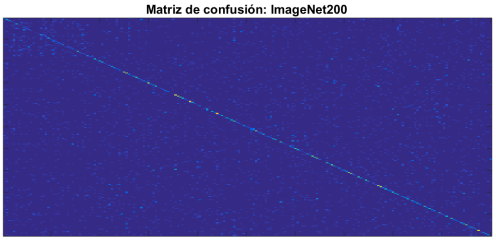

Figura 1. Matriz de confusión resultante de la base de datos de ImageNet200 bajo los mejores parámetros.

En la base de datos de imageNet200 la clase con mejor puntaje de clasificación fue web_site con 22 de 25 aciertos. Las clases con peores puntajes fueron Bedlington_terrier, Chesapeake_Bay_retriever, Labrador_retriever, alligator_lizard, combination_lock, feather_boa, fur_coat, lighter, pencil_box, piggy_bank, saltshaker y thunder_snake con 0 de 25 aciertos.

4.. Discusión y Conclusiones

Los hiperparámetros más relevantes para los resultados, son aquellos que logran un cambio en el puntaje de clasificación media en las clases. Al observar el cuadro 1, se denota que los hiperparámetros de mayor relevancia son: número de palabras, valor de confianza en la clasificación de SVM, cantidad de particiones realizadas en la pirámide de clasificación y el número de imágenes empleadas para el entrenamiento y test.
El único hiperparámetro que no genera cambio en el resultado fue aquel de ajustar el tamaño de la imagen, esto se debe a que el algoritmo únicamente reacomodaba el tamaño vertical de la imagen en caso de ser mayor al estipulado por el hiperparámetro.

Los resultados que se tomaron en cuenta para comparar las dos bases de datos , ImageNet y Caltech_101 fueron los que tenian los parametros iniciales del código. Estos resultados se muestran en el cuadro 1. Para Caltech se obtuvo un ACA de 67.45 % y para ImageNet se obtuvo un ACA de 12.75 %, la gran diferencia en los resultados puede deberse a la cantidad de clases e imágenes en la base de datos. Caltech_101 cuenta solo con 101 categorias, mientras que ImageNet cuenta con aproximadamente 21841 categorias, esto hace que el problema sea más complejo debido a que algunas de las categorias pueden ser parecidas entre ellas lo que dificulta que se clasifiquen correctamente.

La clase web_site es aquella de mejor clasificación, porque se compone únicamente de bordes verticales u horizontales como se observa en la Figura 2, esto se debe a que las distintas capas bajo las cuales se maneja la interacción con el usuario, manejan una geometría uniforme y cuadrada, dichas páginas no poseen una interfaz compleja debido que el programa donde se desarrollaban no tenía la capacidad de superponer múltiples capas y manejar distintas formas en ellas, como suele ocurrir actualmente en la mayoría de sitios web.

Por otro lado, en las clases de peor puntaje, una de sus fallas se puede atribuir a figuras estéticas internas del objeto, las cuales contribuyen ruido al momento de clasificar, esto se denota en la Figura 3 de la clase pencil_box, en la cual se podría reconocer la caja de lápices con HOG de manera simple si sólo se tuviese en cuenta la forma global del objeto e ignorar sus estructuras internas. Otra falla en dichas clases consta de la posición del objeto principal y las estructuras de fondo, donde en la Figura 4 se muestra que el perro perteneciente a la clase Chesapeake_Bay_retriever, se encuentra en una posición donde sólo se puede distinguir su rostro, mientras que el cuerpo está ocluido y en la misma imagen hay múltiples objetos que no hacen referencia a la clase principal.

Figura 2. Muestra de la clase web_site.

Figura 3. Muestra de la clase pencil_box.

Figura 4. Muestra de la clase Chesapeake_Bay_retriever.

Si bien en éste laboratorio, se comprobó que modificando los hiperparámetros y aumentando el número de imágenes en la etapa de entrenamiento se puede obtener un mayor puntaje ACA. Se pueden considerar otros factores para aumentar nuestro resultado tales como: preservar información de texturas e intensidades en determinada localidad de píxeles como se ha venido trabajando en los anteriores laboratorios. Otra forma donde podría aumentar los resultados sería, cambiar el método de clasificación, ya que en éste se emplea SVM, el cual es un clasificador binario. Se puede implementar el uso de clasificadores multiclase como bosques aleatorios con el propósito de mejorar el tiempo de ejecución y evitar crear clasificadores específicos a las respectivas clases.

Referencias

[1] S.-M. Khaligh-Razavi, "What you need to know about the state-of-the-art computational models of object-vision: A tour through the models," *arXiv preprint arXiv:1407.2776*, 2014.