UNIVERSIDAD PÚBLICA DE NAVARRA

DEPARTAMENTO DE ESTADÍSTICA, INFORMÁTICA Y MATEMÁTICAS

Nuevos métodos para la combinación de características en procesamiento de imágenes

Juan Ignacio Forcén Carvalho

TESIS DOCTORAL

Pamplona, Diciembre de 2020

https://doi.org/10.48035/Tesis/2454/39683

UNIVERSIDAD PÚBLICA DE NAVARRA

DEPARTAMENTO DE ESTADÍSTICA, INFORMÁTICA Y MATEMÁTICAS

Nuevos métodos para la combinación de características en procesamiento de imágenes

Juan Ignacio Forcén Carvalho

TESIS DOCTORAL

Pamplona, Diciembre de 2020

Autorización

Dr. Miguel Pagola Barrio: Profesor titular de la Universidad Pública de Navarra en el Área de Conocimiento de Ciencias de la Computación e Inteligencia Artificial.

Dra. Edurne Barrenechea Tartas: Profesora titular de la Universidad Pública de Navarra en el Área de Conocimiento de Ciencias de la Computación e Inteligencia Artificial.

HACEN CONSTAR que el presente trabajo titulado "*Nuevos métodos para la combinación de características en procesamiento de imágenes*" ha sido realizado bajo su dirección por D. Juan Ignacio Forcén Carvalho.

Autorizándole a presentarlo como memoria para optar al grado de Doctor por la Universidad Pública de Navarra.

(Director) Dr. Miguel Pagola Barrio

(Director) Dra. Edurne Barrenechea Tartas

(Doctorando) Juan Ignacio Forcén Carvalho

Pamplona, Diciembre de 2020

Agradecimientos

Quiero agradecer esta tesis doctoral a todas las personas que la han hecho posible, aquellos que han colaborado conmigo o me han ayudado de manera puntual, aquellos que han sido parte activa del trabajo como mis tutores o aquellos como mi familia que me han apoyado y estimulado siempre de manera incondicional.

Me gustaría dar las gracias a mis tutores Miguel y Edurne sin los que esta tesis no habría sido posible. Ellos han supervisado mi trabajo, me han guiado y me han aconsejado, pero también me han inculcado su buen hacer, su entusiasmo y su pasión por la investigación. No podría agradecer de manera suficiente su apoyo ni las horas de discusión con Miguel, sus ideas, sugerencias y motivación.

Quiero también agradecer al Grupo de Investigación de Inteligencia Artificial y Razonamiento Aproximado (GIARA) de la Universidad Pública de Navarra, mostrando mi gratitud especialmente a su director y creador Humberto Bustince, quien me abrió las puertas del grupo. De la misma manera quiero agradecer a todos y cada uno de los miembros de GIARA, que pese a que no he pasado muchas horas en el departamento por mi trabajo, todos ellos siempre se han mostrado dispuestos conmigo, ayudándome y colaborando en todo lo que ha sido necesario.

Finalmente, quiero dedicar esta tesis a mi familia a la vez agradecerles todo lo que han hecho por mi, incluso mucho antes del comienzo de la misma. A mis padres que siempre me han apoyado en esta etapa como en otras, ellos me han dado todo lo que unos padres pueden dar a un hijo como son la educación y los valores. A mi hermana que siempre me ha ayudado desde que éramos pequeños. Y a mi mujer por su apoyo, paciencia y comprensión tras mis jornadas interminables de tesis y trabajo.

Muchas gracias por todo.

Índice general

Índice de figuras

Capítulo I

Memoria

1. Introducción

La irrupción de la digitalización y automatización en nuestra sociedad ha provocado que muchas de las tareas que los humanos han llevado a cabo durante miles de años estén siendo realizadas por computadoras o máquinas. Una característica esencial que tenemos las personas es la capacidad para percibir el entorno a través de los sentidos; entre ellos, la vista o la percepción visual es fundamental en nuestra manera de interactuar con el mundo.

La visión por computador es el área de investigación encargada de dotar a los ordenadores de la capacidad de hacer tareas como las que el ser humano lleva a cabo a partir de su sistema visual, de manera similar a cuando una persona percibe determinada información con la vista y esta es analizada para comprender el entorno, la visión por computador lo hace a través de una cámara y una aplicación o sistema informático.

Tradicionalmente el campo de la visión por computador [116] ha estado centrado en resolver problemas concretos a partir de una o varias imágenes, como la localización y detección de objetos, la clasificación, el reconocimiento de texto o la segmentación de objetos, entre otros. Todos estos problemas son clave en los denominados sistemas inteligentes, como por ejemplo un coche autónomo, una aeronave no tripulada o cualquier sistema informatizado en el que sea necesario tomar una decisión de alto nivel a partir de una imagen. Esta memoria está centrada en dos problemas comunes de la visión por computador como son la **clasificación de imágenes** y la **recuperación de imágenes** (conocido en la literatura como *Instance Image Retrieval*[1]).

La clasificación de imágenes es el problema en el que un sistema debe ser capaz de determinar la categoría, o categoría del objeto principal, de una imagen dada. Algunos ejemplos de clasificación de imágenes pueden ser un sistema que a partir de una imagen médica determine si un paciente

[1] En la memoria haremos referencia al problema de *Instance Image Retrieval* en su denominación en castellano. (*Recuperación de Imágenes*).

tiene o no cáncer [57], el categorizar un tipo de melanoma [14], indicar la especie de un pájaro [128] o la identificación de peatones u otros obstáculos en un vehículo autónomo [100]. En cuanto al segundo problema que abordamos en esta tesis, la recuperación de imágenes consiste en ser capaces de encontrar en una base de datos extensa de imágenes todas aquellas que sean similares a una imagen dada, como ejemplos tendríamos la búsqueda de productos comerciales a partir de uno dado [4] o las búsquedas habituales que realizamos en *Google Images*[2] o *Pinterest*[3] [66] considerando la similitud entre imágenes.

Estos dos problemas, tanto la clasificación de imágenes como la recuperación de imágenes, suelen ser abordados por el subcampo de la IA denominado aprendizaje automático (en inglés *Machine Learning*), y en concreto mediante técnicas de aprendizaje supervisado [83, 74], es decir, los algoritmos o modelos basan su aprendizaje en conjuntos etiquetados de entradas con sus respectivas salidas. Un problema de clasificación o recuperación de imágenes se puede dividir en dos fases, la primera debe ser capaz de caracterizar o extraer información de una imagen, y la segunda de tomar una decisión de nivel superior en base a las características de la imagen obtenidas en la primera fase.

La extracción de información visual es un proceso complejo realizado de manera innata por las personas. Este proceso implica analizar y caracterizar las propiedades de una imagen como la forma, la distribución de color, la iluminación o la textura de los diferentes elementos, para luego relacionarlas jerarquicamente. Por ejemplo, un perro tendrá características similares a las de un gato, pero diferentes a las de un pez.

En visión por computador esta fase de extracción de características relevantes en la imagen es clave para la resolución de los problemas planteados. Las características que encontramos en una imagen pueden ser de diferentes niveles de complejidad, desde bordes o esquinas, hasta características más complejas, como texturas o formas. En definitiva, la capacidad que tiene un sistema para resolver un problema dado depende en gran medida de su capacidad para extraer e identificar las características mas relevantes. Por ello, cuanto más discriminatorias y robustas sean ante variaciones las características o descriptores extraídos por el sistema, mejor será su desempeño.

Durante los últimos años la comunidad científica ha estado centrada principalmente en dos grandes líneas de trabajo, la denominada visión por computador tradicional y la línea más reciente basada en aprendizaje profundo o *Deep learning* [80]. La línea de visión por computador tradicional se basa en la creación de descriptores a partir de características de bajo nivel (texturas, bordes, formas o color) para caracterizar las imágenes y posteriormente, utilizar técnicas de aprendizaje automático sobre los descriptores generados. Por ejemplo, para el problema de clasificación es habitual utilizar regresión logística [71]. Por otro lado, las técnicas de *Deep learning* han cobrado mucha relevancia en las líneas de investigación actuales debido al éxito obtenido en una gran variedad de aplicaciones, en esta técnica, las fases de extracción de características y de clasificación son aprendidas de manera conjunta por un modelo. En los modelos de Deep Learning se realiza un aprendizaje de múltiples

[2] *Google Images*. https://www.google.com/imghp
[3] *Pinterest*. https://www.pinterest.es/

niveles de características, donde las características de más alto nivel se derivan de las características de nivel inferior para formar una representación jerárquica.

Evidentemente, la cantidad de información o características que hay en una imagen es muy elevada. Si se generan características de bajo nivel, como los bordes o el color, se obtienen miles o decenas de miles de descriptores locales en una sola imagen, siendo además dichos descriptores por si solos muy poco representativos. Por ello, estos descriptores locales son agregados para obtener unos pocos descriptores o un descriptor único de la imagen de dimensionalidad reducida (llamado representación compacta). La fusión de la información de varios descriptores en uno solo es utilizada tanto en la visión por computador tradicional [25, 64, 140] como en *Deep learning* [79].

El objetivo de la fusión o agregación de información (descriptores o características) de una imagen, es preservar la información importante y, a su vez, descartar la no relevante. Para ello, existen diferentes estrategias, como por ejemplo tener en cuenta la magnitud de las características descartando aquellas que sean poco representativas, analizando su posición en la imagen o dependiendo de la posición relativa entre ellas. En esta memoria estudiamos diferentes problemas abiertos entorno a la agregación de información y presentamos el estudio de nuevas técnicas de agregación de características de imágenes para mejorar el rendimiento en los problemas de clasificación y recuperación de imágenes, realizando propuestas para algoritmos de visión por computador tradicional y para modelos de *Deep Learning*. Concretamente, hemos propuesto un método para agregar información en problemas de clasificación multi-clase a través de ensembles ponderados. También presentamos una nueva función de *pooling* capaz de auto aprenderse para redes neuronales convolucionales. Centrándonos en el problema de recuperación de imágenes proponemos un esquema de agregación de características donde damos mayor importancia a los objetos relevantes de la imagen y finalmente introducimos una nueva representación de *Co-Ocurrencias* auto aprendible para redes neuronales convolucionales capaz de capturar la correlación espacial entre diferentes características.

La memoria está dividida en dos partes:

- Parte I. Dedicada al planteamiento del problema, la discusión de los métodos propuestos, los resultados y las conclusiones obtenidas.
- Parte II. Contiene las publicaciones asociadas al estudio realizado.

En la Parte I, comenzamos planteando formalmente el problema y presentando el contexto necesario (Sección 2), la motivación concreta que nos ha llevado a la elaboración de esta memoria (Sección 3), y los objetivos propuestos de la misma (Sección 4). Posteriormente, resumimos los trabajos realizados y discutimos los resultados más relevantes obtenidos junto a sus conclusiones (Sección 5). Finalmente, presentamos las conclusiones de la memoria (Sección 6) y las líneas futuras de trabajo que surgen a partir de los estudios realizados (Sección 7).

La Parte II presenta un compendio de cuatro publicaciones asociadas a esta memoria:

- Combinación de características a través de ensembles ponderados para clasificación de imágenes – *Combination of features through weighted ensembles for image classification*
- Aprendiendo operaciones de *pooling* basadas en medias ponderadas ordenadas para clasificación de imágenes – *Learning ordered pooling weights in image classification*
- Agregación de características profundas en recuperación de imágenes basadas en la detección de objetos – *Aggregation of deep features for image retrieval based on object detection*
- *Co-Ocurrencia* de características de redes neurales convolucionales aplicadas a recuperación de imágenes – *Co-occurrence of deep convolutional features for image search*

2. Planteamiento del problema

En esta sección describimos el contexto necesario para entender el trabajo realizado en esta memoria. Las Secciones 2.1 y 2.2 repasan dos de las tareas principales de la visión por computador, la clasificación de imágenes y la recuperación de imágenes respectivamente. La Sección 2.3 profundiza en las aproximaciones clásicas de visión por computador. La Sección 2.4 revisa las nuevas técnicas de aprendizaje profundo o *Deep Learning*. Finalmente, la Sección 2.5 está centrada en la agregación de características.

2.1. Clasificación de imágenes

La clasificación de imágenes es un problema común en visión por computador, consiste en identificar el objeto principal de una imagen dado un conjunto de posibles categorías predefinidas [30]. A la vez que común, la clasificación de imágenes es clave en visión por computador, puesto que otros problemas como la segmentación de imágenes o la detección de objetos pueden ser reducidos o tratados como problemas de clasificación [44].

Un sistema de clasificación es un algoritmo que a partir de una entrada (imagen) genera una predicción de categoría, en forma de probabilidades para las diferentes clases posibles del problema. Partiendo de que una imagen es un conjunto de valores numéricos, que representan las características de la misma, dicha representación es transformada a través de un modelo a un vector de probabilidades que indica la probabilidad de pertenencia a cada una de las posibles clases (p.ej. 90 % Perro, 7 % Gato, 3 % Oso, 0 % Pájaro, 0 % Mono), siendo la clase con mayor probabilidad la asignada por el sistema. (Ver Figura 1.)

En clasificación, las categorías posibles son discretas y conocidas de antemano; si solo hay dos clases posibles de imágenes (p.ej. Perro y Gato) será un problema de clasificación binario, mientras que si hay más de dos clases posibles (p.ej. Perro, Gato, Oso, Pájaro, Mono), entonces se trata de un problema de clasificación multi-clase.

Figura 1: Sistema de clasificación de imágenes.

Este problema de clasificación de imágenes puede ser relativamente sencillo para los humanos (siempre que el dominio sea conocido), sin embargo, los algoritmos de clasificación de visión por computador se enfrentan a los siguientes retos o dificultades:

Figura 2: Dificultades habituales en el problema de clasificación de imágenes.

- **Variación intra-clase**. La apariencia visual para objetos de una misma clase puede ser muy diferente. Esto se denomina variación intra-clase y dificulta el acotar, identificar o categorizar las propiedades o características de una clase dada. (Figura. 2 (a)).
- **Punto de observación**. La posición desde la cual un mismo objeto es capturado u observado provoca que su apariencia visual pueda ser muy diferente aun siendo el mismo objeto. (Figura. 2 (b)).
- **Escala**. La escala de los objetos puede ser muy diferente entre instancias de una misma clase, o incluso con respecto a otros objetos de su entorno. (Figura. 2 (c)).

- **Deformaciones**. Las deformaciones es otro elemento habitual, este caso puede ocurrir por no ser el objeto a capturar rígido o por razones como la óptica de la cámara. (Figura. 2 (d)).
- **Oclusiones**. Ocurre cuando el objeto a identificar este parcialmente tapado por otro objeto dificultando su identificación. (Figura. 2 (e)).
- **Condiciones de iluminación**. Los cambios en las condiciones de iluminación pueden provocar variaciones muy grandes en la apariencia visual de un mismo objeto. (Figura. 2 (f)).
- **Fondo**. El fondo o los objetos que se encuentran cercanos al objeto a detectar pueden dificultar la identificación del objeto de interés, por ejemplo por su similitud. (Figura. 2 (g)).

Cuanto más robusto sea un sistema de clasificación a la combinación de las dificultades presentadas anteriormente y más sensible sea a las variaciones entre clases mayor será su capacidad de clasificación; es decir, la capacidad de clasificación de un modelo dependerá de su capacidad de minimizar la variación entre las imágenes de una misma clase o intra-clase, y maximizar la variación entre clase o inter-clase.

En el ambito científico, los problemas de clasificación son estudiados en base a conjuntos de referencia o *Datasets*, que contienen imágenes categorizadas en las posibles clases a clasificar. Tres ejemplos estándar de conjuntos de datos o *Datasets* utilizados en la experimentación de clasificación de imágenes son:

- **CIFAR-10** [1]: contiene 60000 imágenes en color de 32x32 pixels que pertenecen a 10 categorías diferentes con 6000 imágenes por clase. 50000 imágenes son utilizadas para propósito de entrenamiento y 10000 para evaluación o test.
- **STL-10** [2]: es un conjunto inspirado en CIFAR-10, con 10 clases diferentes, imágenes de 96x96 pixels y enfocado a aprendizaje no supervisado, por ello contiene 800 imágenes por clase de test, 500 imágenes etiquetadas por clase para entrenar y 100000 imágenes no etiquetadas. En la Figura 3 mostramos ejemplos de imágenes de las diferentes clases de este conjunto.
- **ImageNet** [109]: Se trata de un conjunto de clasificación masivo que contiene imágenes descargadas de motores de búsqueda como *flicker* y otros. Este conjunto tiene diferentes versiones, por ejemplo la versión ILSVRC2012 consta 1.2 millones de imágenes de entrenamiento y 150000 para validación y test, todo ello para 1000 categorías diferentes.

El objetivo de un sistema de clasificación es predecir de manera correcta el máximo número de casos, en general para ello se utiliza la métrica de *Accuracy* Ec. (I.1). Aunque no siempre es la mejor opción y pueden ser utilizadas diferentes métricas dependiendo de cada caso, como $Precision$, $Recall$ o $F1Score$. Por ejemplo en el caso de trabajar con clases desbalanceadas la métrica de *Accuracy*

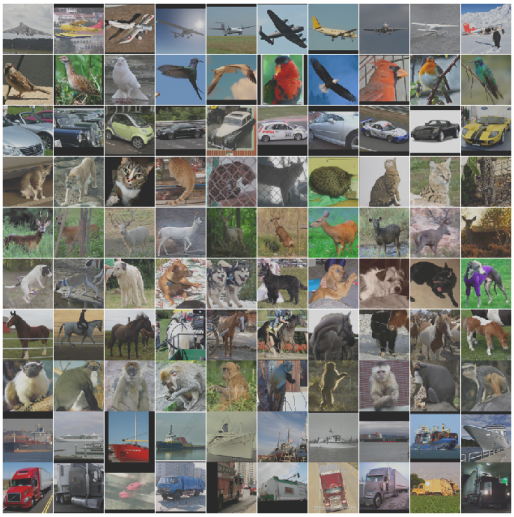


Figura 3: Imágenes del conjunto STL-10.

puede producir un valor alto con un modelo sesgado a la clase dominante, en este caso es preferible utilizar otra métrica.

$$Accuracy = \frac{Predicciones\ Correctas}{Predicciones\ Totales} \tag{I.1}$$

La manera de abordar el problema de clasificación de imágenes está inspirada en cómo realizamos los humanos esta misma tarea y consta de dos fases claramente diferenciadas. La primera fase es la obtención de información de una imagen dada (fase de *extracción de características*), y la segunda fase (llamada fase de *clasificación*) es la encargada de determinar la categoría (clasificar) la imagen, en base a la información o características obtenidas en la primera fase.

La fase de **extracción de características** consiste en transformar una imagen en un conjunto de características. De manera abstracta los humanos obtenemos visualmente información de las imágenes en forma de líneas, bordes, o colores, con las que componemos objetos (por ejemplo ojos, boca o nariz) que una vez agregados pueden dar lugar a un objeto de mayor nivel jerárquico ("cara"). Los sistemas de visión por computador son capaces de extraer información de bajo nivel, como color o líneas, y también son capaces de crear representaciones jerárquicas como la del ejemplo anterior. En general, cuanto más descriptivas sean las características extraídas en la primera fase, mejor será la precisión de clasificación del sistema. Por ejemplo, la información de color seguramente no será suficiente para diferenciar entre gatos y perros, pero sí podría ser suficiente para diferenciar entre otras dos clases diferentes. En la fase de **clasificación** el objetivo es determinar la categoría de la imagen a partir de las características previamente extraídas, en esta fase se utiliza un algoritmo de clasificación.

Actualmente en visión por computador el problema de clasificación de imágenes puede ser abordado desde dos aproximaciones diferentes; un enfoque de aprendizaje máquina tradicional o un enfoque basado en el paradigma de *Deep learning* donde se utilizan las Redes Neuronales Convolucionales (*Convolutional Neural Networks*) CNNs. En las **técnicas tradicionales** de visión por computador, la fase de extracción de características es realizada en base a descriptores que transforman las imágenes de entrada en codificaciones, estos descriptores han sido previamente definidos por expertos en la materia para resolver problemas de visión por computador genéricos o específicos, a esto se le llama ingeniería de características o *feature engineering*. Sin embargo, en **Deep learning** a partir de los datos etiquetados, se realiza simultáneamente el aprendizaje de la fase de extracción de características y la fase de clasificación de manera conjunta[4]. En la Figura 4 observamos las diferencias entre las dos aproximaciones mencionadas previamente.

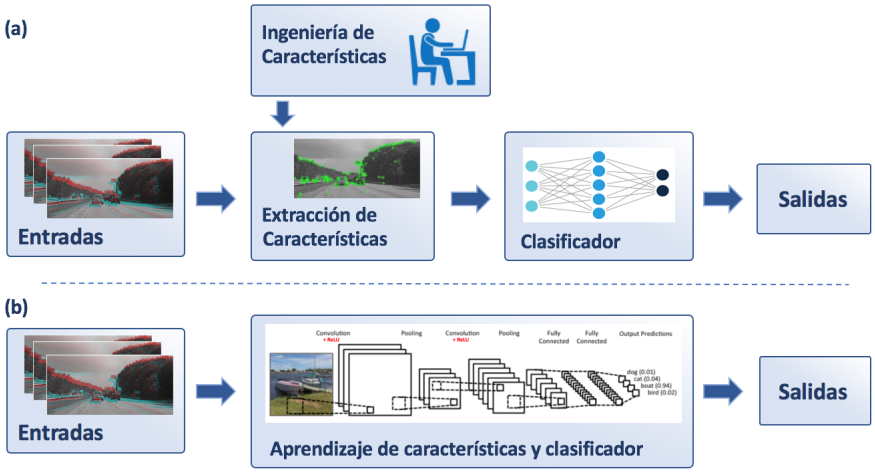


Figura 4: Flujo en visión por computador tradicional (a) y en *Deep Learning* (b).

El *Deep learning* es una rama de *Machine learning* basada en redes neuronales artificiales (*Artificial Neural Networks* ANN), e inspirado en el funcionamiento del cerebro humano, es decir está compuesto por un conjunto de neuronas que realizan operaciones simples e interactúan entre ellas para tomar decisiones complejas. En general, los modelos que utilizan soluciones basadas en *Deep learning* presentan los mejores resultados en la mayoría de las disciplinas de visión por computador [127, 70, 75]. Esto se debe principalmente a la capacidad que tiene el sistema de aprender, durante la fase de entrenamiento, ciertos patrones o relaciones complejas implícitas en los datos, permitiendo resolver problemas con muchas clases donde las técnicas tradicionales, basadas en la extracción de características, no son capaces de hacerlo por la dificultad que tienen para caracterizar problemas muy complejos.

No obstante, también hay problemas que son resueltos de manera más eficiente con técnicas

[4]El aprendizaje completo del modelo se denomina en Inglés aprendizaje *end-to-end*

tradicionales de visión por computador [94]. Además, en algunas ocasiones las soluciones basadas en *Deep learning* tienden a sobre-ajustarse[5] a los conjuntos de datos que han sido utilizados durante la fase de entrenamiento, lo que conlleva a una peor generalización del modelo. Para que no suceda este efecto de sobre-ajuste, es importante que el conjunto de datos de entrenamiento sea lo suficientemente grande y amplio en cuanto a variabilidad y riqueza en la muestra.

2.2. Recuperación de Imágenes

La recuperación de imágenes [108], al igual que la clasificación de imágenes, es un problema clave en visión por computador. Este problema consiste en ser capaces de buscar en una base de datos extensa, de imágenes no clasificadas, las imágenes más similares a una imagen de consulta[6] dada, siendo las más similares las de la misma instancia que la imagen de consulta. (Ver Figura 5). Algunos ejemplos de recuperación de imágenes son: la búsqueda de productos comerciales, como hace *Amazon* para facilitar a los usuarios encontrar el producto deseado, la búsqueda de imágenes similares a una dada de *Google Images* [7] y *Bing Visual Search*[8] o el reconocimiento de la localización visual a partir de una foto.

Una de las principales diferencias entre recuperación de imágenes y la clasificación de imágenes, es que en este caso el número de categorías no está acotado, por ello las aproximaciones al problema son diferentes a las de clasificación. Los problemas a los que se enfrenta la recuperación de imágenes se corresponden con la necesidad de describir un mismo objeto de manera uniforme en condiciones muy diferentes:

- **Punto de observación**, **Escala**, **Oclusiones**, **Condiciones de iluminación**. Estas dificultades son compartidas con la clasificación de imágenes y que han sido presentados anteriormente en la Sección 2.1.
- **Objetos similares deben ser diferenciados**. Este problema responde a la necesidad de poder diferenciar objetos similares en recuperación de imágenes, aunque visualmente sean muy parecidos. (Figura 6).
- **Eficiencia**. Los sistemas de recuperación de imágenes deben ser capaces de manejar millones o billones de imágenes, con las implicaciones que ello tiene a nivel de memoria, procesamiento y tiempo de búsqueda; esto hace que la eficiencia sea clave en este problema, condicionando las posibles soluciones y razón por la cual la mayoría de métodos del estado del arte transforman cada imagen en una representación compacta.

[5] Sobre-ajuste, del término en ingles *Overfitting* y relativo al hecho de que un un modelo se ajuste en exceso a los datos de entrenamiento, provocando que no generalice bien.
[6] En recuperación de imágenes, la consulta es una imagen denominada imagen de consulta, del inglés *Query Image*
[7] *Google Images*. https://www.google.com/imghp
[8] *Bing Visual Search*. https://www.bing.com/visualsearch

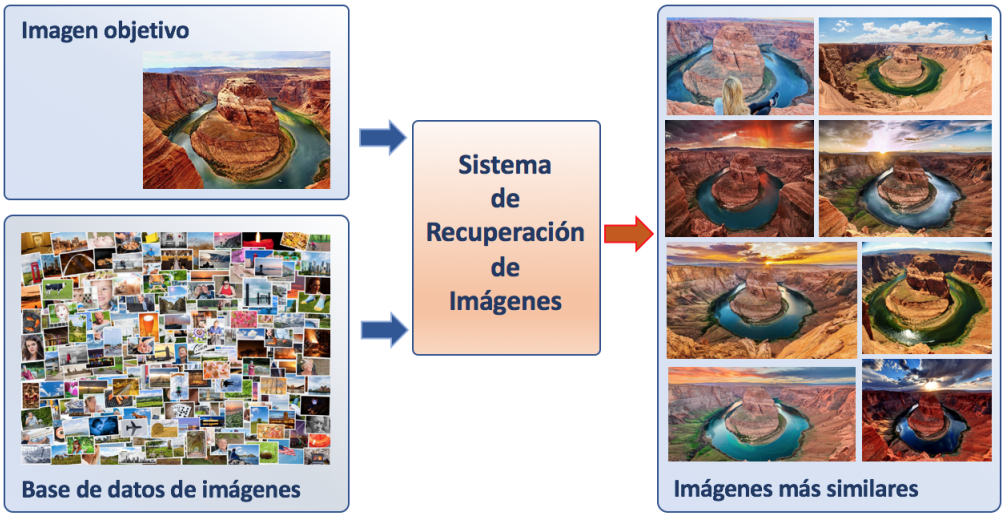


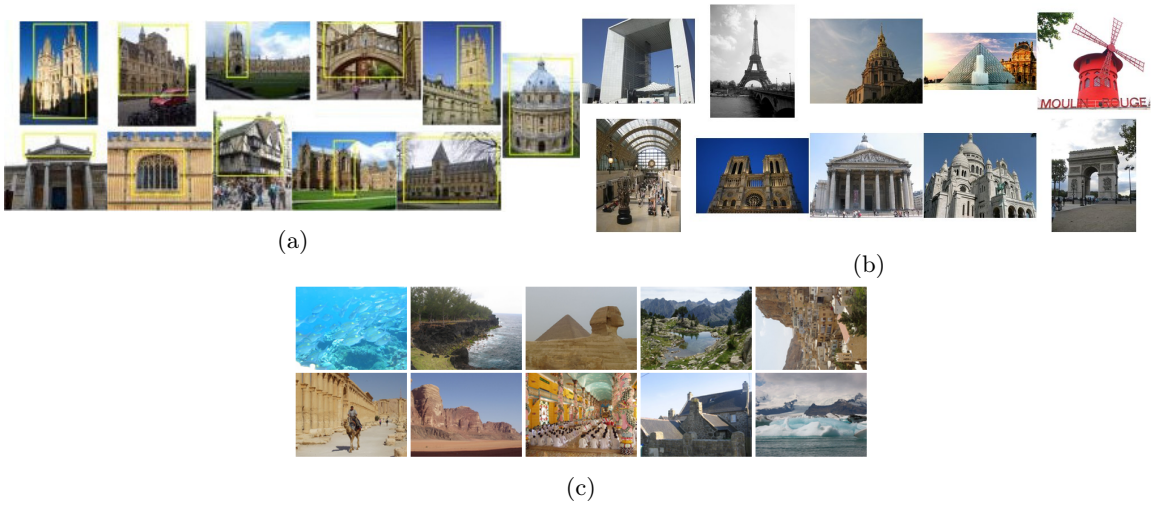
Figura 5: Sistema de recuperación de imágenes.

Figura 6: Ejemplo del problema de objetos similares en recuperación de imágenes.

De manera similar a los problemas de clasificación, algunos de los conjuntos de referencia utilizados en recuperación de imágenes son: Oxford-5k [96], Paris-6k [97], INRIA Holidays [61], ROxford [103] y RParis [103]. A continuación describimos brevemente las características de cada uno de estos conjuntos:

- **Oxford-5k** [96]. Oxford Buildings Dataset, contiene 5062 imágenes de edificios de la ciudad de Oxford, obtenidas a partir del motor de búsqueda *Flickr* y manualmente anotadas en 11 lugares o edificios diferentes, cada uno representado con 5 posibles consultas, resultando en total 55 consultas para evaluar el problema de recuperación de imágenes. En la Figura 7 se muestra una imagen consulta ejemplo para cada uno de los diferentes lugares o edificios (del inglés *Landmarks*).

- **Paris-6k** [97]. Este dataset contiene 6412 imágenes obtenidas de manera similar al conjunto Oxford-5k pero en este caso de la ciudad de Paris. De la misma manera hay 11 lugares de interés o edificios con 5 consultas cada uno, generando un total de 55 consultas. En la Figura 7 se muestra una imagen consulta ejemplo para cada uno de los diferentes lugares de interés o *Landmarks*.



(a) (b)

(c)

Figura 7: Imágenes de los conjuntos *Oxford Buildings* (a), *Paris* (b) y *Holidays* (c).

- **INRIA Holidays** [61]. Es un dataset de 1491 imágenes procedentes de lugares vacacionales donde se busca la robustez en cuanto a rotación, puntos de observación, iluminación o desenfoque. A diferencia de Oxford5k y Paris6K, INRIA Holidays contiene un número mucho mayor de posibles lugares o escenas (500 consultas) dificultando el problema, entre ellas hay imágenes de escenas de naturaleza, agua o fuegos artificiales. En la Figura 7 observamos algunas de las consultas ejemplo.

- **ROxford** y **RParis** [103]. Son conjuntos creados a partir de las imágenes de Oxford-5k y Paris-6k, pero donde ha sido revisada la anotación de las imágenes y se han propuesto nuevas consultas *Queries* más complejas, así como subconjuntos de evaluación de diferentes niveles de dificultad.

Los sistemas de recuperación de imágenes devuelven ordenadas por similitud las imágenes contenidas en una base de datos para una imagen consulta dada, siendo las imágenes de la misma instancia a la consulta *Query* las más similares. Para evaluar estos sistemas se utiliza la métrica *AP Average Precision* que puede ser formulada como el área bajo la curva *Precisión Recall* Ec. (I.2), o en la práctica como la suma en cada posición de la secuencia ordenada Ec. (I.3).

$$AP = \int_0^1 p(r)dr \tag{I.2}$$

$$AP = \sum_{k=1}^{n} P(k)\Delta r(k) \tag{I.3}$$

Donde $P(k)$ es la precisión teniendo en cuenta los primeros k elementos y $\Delta r(k)$ es el incremento

de *Recall* entre $k-1$ y k. Dado que en los problemas de recuperación de imágenes es común tener más de una imagen consulta la métrica final utilizada es la media de todos los AP de las diferentes consultas, la medida denominada como MAP *Mean Average Precisión*, Ec. (I.4).

$$MAP = \frac{\sum_{i=1}^{m} AP_i}{m} \tag{I.4}$$

A continuación se explican los procesos clave que definen el flujo del problema de recuperación de imágenes. Estos son la generación de la representación comparable de la imagen, denominado Representación compacta de la imagen (*Image Representation*), búsqueda por similitud (*Image Search*) y Expansión de la consulta (*Query Expansion*).

- **Representación compacta de la imagen**

  Este paso corresponde a la necesidad de generar una representación compacta de cada imagen, siendo la representación compacta de una imagen un vector de características n-dimensional capaz de codificar cualquier imagen del sistema; además, cuanto más descriptivas y discriminatorias sean las representaciones compactas generadas mejor será su funcionamiento. (Siendo idealmente invariante a los retos presentados anteriormente, Punto de observación, escala, oclusiones, iluminación...). El hecho de obtener una representación compacta de una imagen permite comparar la imagen consulta con las imágenes de la base de datos entre sí y además de manera eficiente, esto es especialmente importante si tenemos en cuenta que en el problema de recuperación de imágenes puede haber millones de imágenes en la base de datos. En la práctica, habitualmente estas representaciones son generadas a partir de una Red Neuronal Convolucional (CNN). (Figura 8)

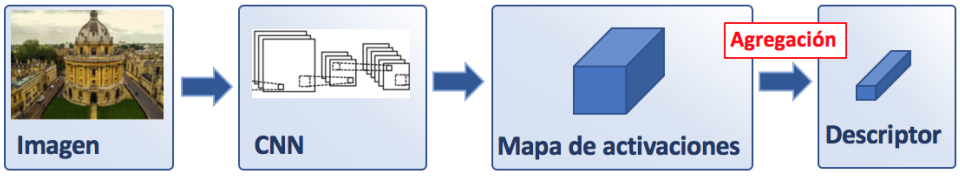


Figura 8: Representación compacta en recuperación de imágenes a partir de CNNs.

- **Similitud entre imágenes**

  Una vez tenemos las representaciones de todas las imágenes de búsqueda y de la imagen consulta, el siguiente paso es averiguar qué imágenes de la base de datos son más similares a la consulta. La distancia coseno es la distancia habitualmente utilizada para comparar las representaciones de dos imágenes, con la particularidad de que si el conjunto de descriptores están normalizados siguiendo la l2-norma [82], la búsqueda coseno es equivalente a la obtenida utilizando la distancia euclidea, siendo esta más simple. Dados dos vectores $x, y \in \mathbb{R}^D$ la

similitud entre x e y se mide con la distancia coseno medienta la siguiente expresión Ec. (I.5), (siendo $||x||$ el vector x l2-normalizado):

$$cos(x, y) = \frac{\sum_{i=1}^{D} x_i y_i}{||x||||y||}, \tag{I.5}$$

La respuesta del sistema es el listado de imágenes ordenado a partir de esta distancia, es decir, tendremos una lista ordenada de mayor a menor similitud de las imágenes de la base de datos con respecto a la imagen consulta.

- **Expansión de la consulta**

  El proceso de expansión de la consulta, (*Query Expansion*) (QE) consiste en realizar una segunda consulta a partir de los descriptores más similares resultantes de la primera consulta. Este proceso produce resultados mejorados al integrar en la segunda consulta información o características de las imágenes más similares a la primera y que pudiesen no estar presentes en ella. El método más simple es hacer la media de los n descriptores más similares, (llamado en inglés *Average Query Expasion* AQE) aunque hay diferentes alternativas en la literatura [20, 21, 132].

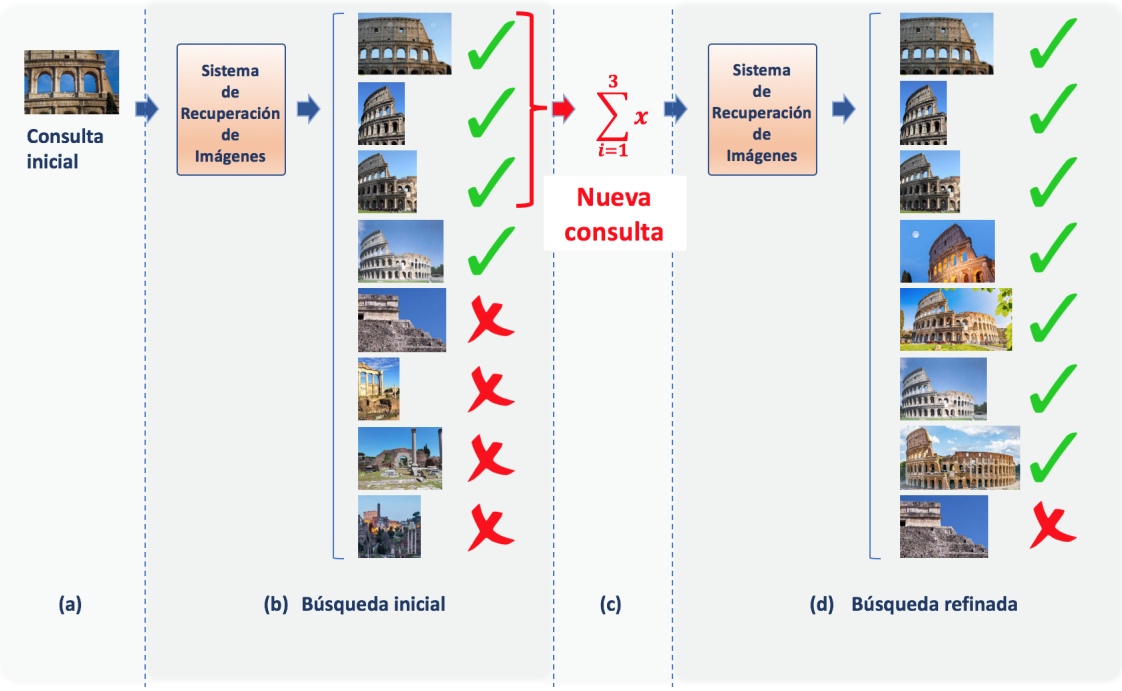


Figura 9: Ejemplo gráfico de la operación de expansión de la consulta. (a) Consulta inicial. (b) Resultado de dicha consulta. (c) Nueva consulta combinando los descriptores más similares resultantes de la primera búsqueda (*Expansión de la consulta*). (d) Resultados de la segunda consulta.

En la Figura 9, mostramos un ejemplo del concepto de expansión de la consulta. En (a) se realiza una consulta con una imagen dada al sistema de recuperación de imágenes, dicha consulta devuelve las imágenes más similares en la base de datos en (b), como podemos ver algunas imágenes son visualmente similares pero son de otra instancia diferente. En (c) se genera una nueva consulta combinando los descriptores de las imágenes más similares de la primera búsqueda, suponiendo que las primeras respuestas son correctas, con este proceso se consigue añadir al nuevo descriptor información adicional de imágenes de la misma instancia que la consulta hecha inicialmente. Finalmente en (d) podemos ver los resultados mejorados de la nueva consulta, al incluir esta más información que la inicial.

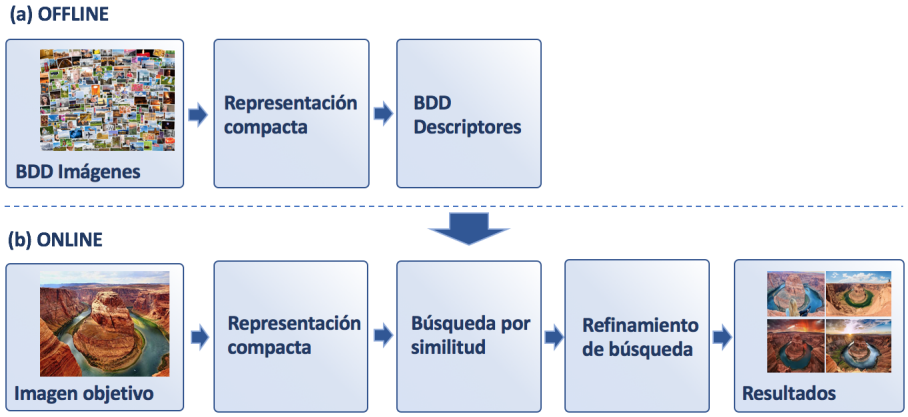


Figura 10: Procesos en recuperación de imágenes. (a) Procesos que pueden ser realizados previos a la consulta. (b) Procesos que deben ser realizados en tiempo de ejecución.

Como se ha introducido, el paso previo a encontrar las imágenes más similares a la consulta, es necesario transformarlas a una representación que permita dicha comparación. Esta transformación de la imagen consulta tiene que hacerse en tiempo de ejecución de la consulta, sin embargo la transformación de todas la imágenes a comparar puede haberse realizado previamente conforme se crean o añaden imágenes a la base de datos. De esta manera se consideran dos fases en los sistemas de recuperación de imágenes, la fase que se realiza en tiempo de ejecución, y la fase *offline*. (Ver Figura 10).

En la fase realizada en tiempo de ejecución, después de transformada la imagen consulta en una representación comparable, se realiza la búsqueda de las imágenes más similares en la base de datos para la generación de resultados de la consulta de manera ordenada. Posteriormente, aunque no sea estrictamente necesario, es habitual hacer diferentes tipos de refinamiento; refinamiento de búsqueda *Re-Ranking*), Verificación espacial, o Expansión de la consulta *Query expansion*.

Una vez presentados los problemas de clasificación de imágenes y de recuperación de imágenes, en los siguientes apartados se exponen las técnicas más comunes con las que son abordados.

2.3. Visión por computador tradicional

2.3.1. Extracción de características

Tradicionalmente la extracción de características, o la extracción de descriptores, en visión por computador ha estado centrada en analizar, caracterizar y entender la información más relevante que permite representar y describir a una imagen. En la literatura podemos encontrar gran cantidad de características obtenidas o diseñadas de manera manual, también denominadas en ingles *Handcrafted Features* [91]. Estas características manuales han sido utilizadas en innumerables aplicaciones de la visión por computador como clasificación de imágenes, detección de instancias o localización de objetos. Durante años la literatura ha estado centrada en el estudio, la comprensión, la creación y la mejora de las características que pueden obtenerse a partir de una imagen.

A finales de los años 1980 y principios de los 1990 la extracción de características de imágenes estuvo centrada en aproximaciones basadas en filtros. En estos casos la imagen era convolucionada con un conjunto de filtros seguido de una función de activación no lineal [77], o como los conocidos *Gabor filters* [13, 88, 59], *Gabor Wavelets* [88], *Wavelet Pyramids* [35, 87], o filtros lineales simples [86]. (Ver Figura 11)

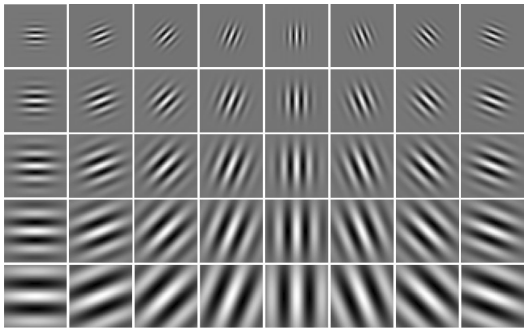


Figura 11: Filtros de *Gabor*, diferentes orientaciones y frecuencias.

También se realizaron caracterizaciones basadas en distribuciones de probabilidad como *Markov Random Field* (MRF) [24, 89]. A continuación en los años 1990s, se identificó la necesidad de tener descriptores invariantes, para ser más robustos a variaciones de iluminación, escala, traslaciones, rotaciones o transformaciones, dando lugar a los descriptores locales invariantes, algunos de los más conocidos son *Scale Invariant Feature Transform* (SIFT) [85], *Speeded Up Robust Features* (SURF) [9] y *Local Binary Patterns* (LBP) [92]. Posteriormente llegaría el método llamado *Bag-of-Features* (BoF) [26], [125] donde se aprende un diccionario de características y las imágenes son representadas como un histograma del propio diccionario, de forma similar a como se realiza la clasificación de textos. Gran parte de estas características han sido creadas para solventar o mejorar casos concretos, como variaciones en escala, rotación, iluminación u oclusiones. Esto proporciona características, o extractores de características, de diferente naturaleza y con mayor capacidad de adaptación a

problemas generales de visión por computador. Aunque como es evidente, también existe la necesidad de obtener descriptores enfocados a problemas particulares o a un dominio concreto ante la imposibilidad de obtener un descriptor único como solución global.

Una imagen puede tener cientos o miles de características de diferente nivel de abstracción, que normalmente son agregados para conseguir una mejor representación de la imagen. Los descriptores agregados suelen considerarse descriptores locales, si corresponden a zonas parciales de la imagen, o globales si describen a toda la imagen. En este trabajo profundizamos en la mejora de estas fases de agregación de características, así como en la combinación o agregación de características de diferente naturaleza, para conseguir una representación mejorada de la imagen, que proporcione mejoras en la resolución de los diferentes problemas de visión por computador.

A continuación se describen los métodos descriptores de características que han sido utilizadas en esta memoria; histogramas de gradientes orientados (HOG) [27], histogramas de color [115] y el método de bolsa de características (*Bag of Features* (BoF)) [114].

Los histogramas (HOG) [27] consideran que un objeto y su forma pueden ser caracterizados por la distribución de sus gradientes locales, es decir capturando la información relativa a formas o bordes de las imágenes.

En la Figura 12 (b) vemos una representación del HOG de la imagen original Figura 12 (a), donde en cada celda están dibujadas las direcciones del histograma.

(a) (b)

Figura 12: Histogramas de gradientes orientados: (a) Imagen original, (b) Representación HOG superpuesta.

El proceso de obtención de los descriptores HOG a partir de una imagen es el siguiente:

1. Se divide la imagen original en diferentes celdas.
2. Para cada celda se crea un histograma de orientaciones que acumula las direcciones de los gradientes de todos los píxeles de esa celda. El histograma suele tener 8 o 9 bins [27].
3. Se divide la imagen en diferentes bloques, que contienen varias celdas, y se normaliza el contraste dentro de cada bloque. Este paso reduce la dependencia debida a diferente iluminación.

4. Se concatenan los histogramas de todas la celdas de la imagen para forma el vector de características de la imagen a partir del cual generalmente se entrena un clasificador [27]. Las aplicaciones de estos vectores de características son muy variadas. (Vease [68, 117]).

Otro tipo de descriptor de características son los histogramas de color. Los histogramas, en general, son una manera común y simple de cuantificar la cantidad de veces que aparece una característica en una imagen. Aunque es una característica denominada de bajo nivel, ha obtenido resultados muy positivos (Ver [115] y [19]). Es importante destacar que en este tipo de características la información relativa a las relaciones espaciales se pierden, esto tiene como ventaja la invarianza ante rotaciones y traslaciones. Sin embargo, no permite capturar la posible dependencia espacial entre características.

En general, en procesamiento de imagen, un color es representado como un vector de tres dimensiones para cada pixel de la imagen. La codificación de color dependerá del espacio de colores en el que ha sido codificado e influye en la representación final. Por ejemplo, una representación de color invariante a la iluminación permitira que dos objetos iguales en una situación de iluminación diferente tengan descriptores de color similares. Si utilizamos la representación de color (*Hue, Saturation, Lightness*) HSL la componente L es la encargada de caracterizar la luminancia, esto hace que el tono (H) y la saturación (S) se mantengan invariantes ante cambios de iluminación.

Chamorro et. al. [18] presentaron los conceptos de una nueva representación de color basada en los principios de la lógica difusa [137] creando un espacio de colores fuzzy (difusos). Esta representación, pretende cubrir el salto entre las representaciones de colores de los ordenadores y la percepción humana. En la representación básica surgen 13 colores fuzzy basados en el sistema ISCC-NBS [69] y en el trabajo de sobre definicción de colores y testeado en personas [10]. Dichos colores son 10 colores básicos (rosa, rojo, naranja, amarillo, marrón, oliva, verde, amarillo-verdoso, azul y púrpura), además de 3 colores neutros (blanco, gris y negro).

Los pasos para obtener un histograma de color fuzzy son los siguientes:

1. Dividir las imágenes en regiones o celdas.

2. Calcular el histograma de color fuzzy para cada bloque añadiendo los grados de pertenencia para cada pixel a cada color fuzzy.

3. Concatener todos los histogramas para obtener el descriptor *Fuzzy Color Histogram (FCH)* que describa la imagen.

Esta representación del color permite una caracterización difusa, esto es especialmente interesante si tenemos en cuenta la variabilidad en la percepción de los colores de un mismo objeto, al depender de condiciones como la iluminación.

El último descriptor de características que describimos es el método de bolsa de características o *Bag-of-Features* (BoF) [114]. Su origen es el análisis de texto, donde un documento es caracterizado y clasificado por la frecuencia de las palabras que contiene, independientemente de su orden. Cuando

este método es aplicado a procesamiento de imagen, las características locales de las imágenes se tratan como si fuesen palabras, de esta manera a partir de un vocabulario caracterizaremos una imagen como un conjunto de palabras, teniendo tantas como características hayan sido extraídas de la imagen. Finalmente, estas palabras son agregadas con el objetivo de tener una representación compacta de la imagen.

Los pasos en que se divide el método BoF son los siguientes:

1. Extracción de descriptores locales de una imagen dada. Esta extracción puede ser de puntos considerados clave de la imagen (*keypoints*) o en rejilla. Algunos ejemplos de descriptores locales son HOG, SIFT, o incluso regiones de la imagen como en Coates et. al. [22].

2. Aprendizaje del diccionario o vocabulario. A partir de un conjunto de descriptores, se utiliza un algoritmo de agrupamiento (*clustering*) para obtener un diccionario de n palabras. Comúnmente se utiliza el algoritmo *k-means* [53], donde el número de grupos (*clusters*) encontrados es el número de elementos del diccionario; el descriptor asociado a cada uno de los grupos se conoce como *palabra visual*.

3. Codificación. Este paso es considerado como una función de activación del diccionario, de manera que cada palabra visual se activa de acuerdo con su parecido al descriptor original. El vector de características, por lo tanto, tiene tantas componentes como palabras visuales del diccionario que se ha aprendido en el aprendizaje no supervisado.

4. Agregación. En esta etapa todos los vectores codificados de la imagen (o de zonas de la imagen) se combinan para formar un único vector y generar por tanto una representación compacta.

   En el caso de utilizar BoF en clasificación de imágenes, un último paso sería el entrenamiento del clasificador a partir del vector o vectores resultantes de la fase de agregación. Este clasificador puede ser un SVM [23] u otro como los presentados en 2.3.2.

A continuación, exponemos los conceptos básicos sobre algoritmos de clasificación, sobre la extracción de características tradicional y sobre la extracción de características en CNNs.

2.3.2. Clasificación

En el ámbito del aprendizaje supervisado, las tareas de clasificación son aquellas que, a partir de un conjunto de muestra de un problema que ha sido previamente etiquetado (conjunto de entrenamiento). Construyen un modelo predictivo (clasificador) capaz de clasificar ejemplos desconocidos. Cada uno de los ejemplos $x = (x_1, \ldots, x_F)$ contenidos en el conjunto de entrenamiento pertenece a una clase $y \in \mathbb{C} = C_1, C_2, \ldots, C_M$, donde M es el número de clases del problema, y está determinado por un conjunto F variables (también denominadas características o atributos), donde cada variable x_i puede tomar cualquier valor contenido en el conjunto $\mathcal{F}_i$.

Por tanto, la construcción de un clasificador consiste en encontrar una función de decisión $h : \mathcal{F}_1 \times ... \times \mathcal{F}_F \rightarrow \mathbb{C}$ óptima que maximice la bondad del clasificador, la cual es evaluada empleando técnicas de estimación del error. Estas técnicas están basadas en el porcentaje de acierto del clasificador sobre un conjunto de prueba compuesto por ejemplos que no han sido utilizados durante el proceso de aprendizaje. De esta forma, el proceso completo de construcción de un clasificador está compuesto por dos fases:

- Fase de aprendizaje o de entrenamiento: la función de decisión $h : \mathcal{F}_1 \times ... \times \mathcal{F}_F \rightarrow \mathbb{C}$ se ajusta en base al conjunto de entrenamiento, obteniendo el modelo predictivo.
- Fase de prueba o test: el modelo obtenido se evalúa empleando ejemplos que no han sido utilizados durante el aprendizaje. En esta fase se pone a prueba la capacidad de generalización del modelo, es decir, la capacidad de clasificar correctamente ejemplos desconocidos.

En la literatura existen diversos tipos de algoritmos de clasificación, como por ejemplo las Máquinas de Soporte Vectorial (SVM) [124], los Árboles de Decisión[102], el Razonamiento Basado en Casos [3], las Redes Neuronales Artificiales (ANN) [130], o los Sistemas de Clasificación Basados en Reglas Difusas (SCBRDs) [58]. Generalmente estos métodos difieren en varios aspectos como la precisión, la interpretabilidad del modelo, o los requerimientos computacionales y de almacenamiento.

Ensembles multiclase De manera general, es más sencillo construir clasificadores para problemas donde solo existen dos clases (binarios) que para problemas con más de dos clases (problemas multi-clase). Esto es debido a que la complejidad de las fronteras de decisión suele aumentar cuando se considera un número mayor de clases. Motivadas por este hecho surgen las estrategias de descomposición, las cuales afrontan los problemas multi-clase dividiéndolos en problemas binarios más sencillos de resolver. Cada uno de estos problemas binarios es abordado por un clasificador independiente denominado *clasificador base* que se especializa únicamente en un par de clases. De la agregación de las predicciones de todos los clasificadores base se obtiene la clase predicha. En varios estudios se demuestra que este tipo de metodología ha sido capaz de mejorar el rendimiento de clasificadores en los que se aborda el problema multi-clase de manera inherente [40, 37, 56, 38]. Además, su utilización permite aplicar clasificadores binarios populares como las SVM [124].

En la literatura especializada se han propuesto diferentes estrategias de descomposición [84], a continuación se presentan dos aproximaciones muy extendidas como son la aproximación uno contra todos (*One-vs-All*) y la aproximación uno contra uno (*One-vs-One*).

Uno contra todos. La descomposición uno contra todos o en ingles *One-vs-All (OVA)* divide un problema de clasificación multiclase con m clases en m problemas binarios de clasificación. Cada sub-problema es un clasificador binario, también llamado clasificador base, el cual debe clasificar entre la clase C_i y el resto (todas las demás son consideradas una única clase). Para entrenar cada clasificador base se utiliza toda la muestra de entrenamiento, considerando las instancias de la clase

C_i como positiva y el resto de la muestra como clase negativa. En validación, cada instancia es evaluada por todos los clasificadores binarios. La aproximación más común es utilizar la confianza resultante de cada clasificador binario para decidir la respuesta final, habitualmente la clase con mayor confianza es la considerada como clase estimada por el modelo. Partiendo de un vector de confianzas $R = (r_1, r_2, \cdots, r_m)$, donde $r_i \in [0, 1]$ es el valor de confianza para cada clase i, entonces la clase estimada viene dada por:

$$\text{Class} = \arg \max_{i=1,\ldots,m} r_i. \tag{I.6}$$

Uno contra uno. La descomposición uno contra uno, en ingles *One-vs-One (OVO)*, divide un problema con m clases en $m(m-1)/2$ problemas binarios. Cada problema es tratado como un clasificador binario responsable de diferencia entre la clase C_i de la clase C_j. Cada instancia es clasificada considerando todos los clasificadores base cuyas salidas o confianzas son almacenados en una matriz de votación:

$$R = \begin{pmatrix} - & r_{12} & \cdots & r_{1m} \\ r_{21} & - & \cdots & r_{2m} \\ \vdots & & & \vdots \\ r_{m1} & r_{m2} & \cdots & - \end{pmatrix}, \tag{I.7}$$

siendo $r_{ij} \in [0, 1]$ el valor de confianza para C_i dado un clasificador que diferencia entre $\{C_i, C_j\}$; a su vez la confianza de C_j es calculada como $r_{ji} = 1 - r_{ij}$. Siendo la matriz de votación simétrica.

Para clasificar la clase a la que pertenece una instancia pueden ser utilizados diferentes métodos de agregación a partir de la matriz de votación [39, 15]. El método más simple es que cada clasificador proporciones un voto a la clase estimada, siendo la clase (o fila) con más votos la que será definida como clase estimada. Uno de los principales problemas de OVO es cómo combinar los resultados de todos los clasificadores binarios, en este sentido hay diferentes aproximaciones posibles como las basadas en los resultados obtenidos de la clasificación [32] o las basadas en el coste [55]. Además, la estrategia OVO tiene el problema de los clasificadores no competentes. Por ejemplo, si un clasificador OVO intenta clasificar una instancia de la clase z, el clasificador que se ha entrenado para distinguir entre la clase i y la clase j, se le llama como no competente ya que no ha sido entrenado con ejemplos de la clase z. En [41] Galar et. al. propusieron un método para evitar la influencia de los clasificadores no competentes mediante una matriz de votación ponderada basada en la distancias relativas de las instancias a clasificar con respecto a las clases.

2.3.3. Recuperación de imágenes

Los primeros trabajos en recuperación de imágenes utilizaban características locales, del inglés *Hand-crafted features*, para la representación de imágenes junto a algoritmos de búsqueda o verificación [85, 114, 98, 62] para el proceso de recuperación. Posteriormente la mayoría de trabajos utilizaron la representación de características de BoF, con tamaños muy grandes del diccionario visual [98, 6], y la verificación espacial, en el que las características locales deben cumplir con una relación geométrica entre ellas [98, 110]. También se mejoró la precisión del proceso añadiendo el concepto de QE [20, 101, 121]. Estas aproximaciones implicaban tener por cada imagen un conjunto amplio de descriptores locales que debían ser comparados con cada una de las imágenes de la base de datos. Con el objetivo de conseguir una representación compacta para cada una de las imágenes algunos trabajos de recuperación de imágenes se centraron en la agregación en un único vector o representación compacta, haciendo uso de técnicas basadas en las transformaciones de base como el PCA, *Fisher Vectors* [64] o el *VLAD* [63, 95].

2.4. Redes Neuronales Convolucionales

Las redes neuronales convoluciones, o (*Convolutional Neural Networks* CNNs) son un caso particular de las redes neuronales artificiales (*Artificial Neural Networks* ANNs) centradas en la resolución de problemas de imagen, fueron introducidas por Fukushima [36] y posteriormente reafirmadas por LeCun [79] y Hinton [75]. A partir de 2012, tras los resultados obtenidos con la propuesta de Krizhevsky et. al. [75] en la competición de reconocimiento *ImageNet Large Scale Visual Recognition Challenge* [109] las CNNs han supuesto un avance en el estado del arte en prácticamente la mayoría de las tareas o problemas relacionados con la visión por computador y el procesamiento del lenguaje natural.

El concepto de ANN está inspirado en el cerebro humano, en el que conjuntos de neuronas producen respuestas o decisiones ante un problema concreto. El elemento básico es una neurona, que a partir de un impulso o impulsos es capaz de generar una respuesta. Una neurona es definida por sus entradas, los pesos con los que multiplica a las mismas y el bias. Adicionalmente, y de manera opcional, a continuación de cada neurona puede aplicarse algún tipo función no-lineal, (Ver Figura 13 (a)). En las Redes Neuronales Totalmente Conectadas, del inglés *Fully Connected Neural Networks*, las neuronas se organizan en capas donde todas las neuronas de una capa están conectadas con todas las de la siguiente (Ver Figura 13 (b)). Por tanto, toda la red neuronal se puede expresar como una función diferenciable que a su vez puede ser entrenada para un problema determinado en base a un conjunto de datos y a una función de pérdida que permita aprender los pesos y bias de cada neurona individual de la red.

En las CNNs una imagen es una estructura de datos de tres dimensiones (*ancho x alto x color*), por lo que para poder aplicar los conceptos de las ANNs se necesitan capas adicionales. Una CNN

(a) (b)

Figura 13: (a) Neurona individual, elemento básico de las redes neuronales. (b) Red neuronal totalmente concetada, compuesta de neuronas agregadas en capas.

esta compuesta al menos por cuatro tipos de capas diferentes:

- **Capa convolucional**. Realiza la convolución entre un tensor[9] de entrada y una serie de filtros para generar una nueva representación (Ver Figura 14). Suponiendo que en cada filtro hay un tipo de característica, al hacer la convolución entre cada filtro y la imagen conseguimos una nueva representación con valores mayores donde se encuentren las diferentes características. (Similar al concepto de convolución con filtros de *Gabor* presentado en 2.3.1). El resultado de esta operación se denomina *mapa de características o mapa de activaciones*.

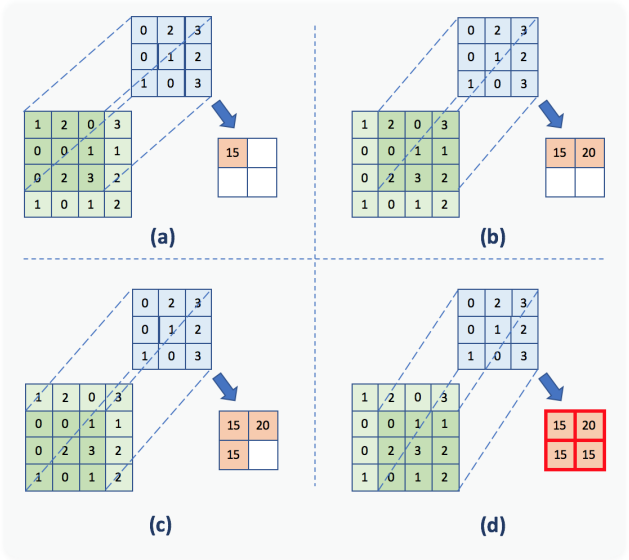


Figura 14: Proceso de convolución entre un filtro y una imagen.

[9]Siendo el tensor correspondiente a una imagen un array de tres dimensiones, *ancho x alto x profundo*. La profundidad en la primera capa dependerá de la representación de la imagen (1 canal blanco y negro o 3 en color), en las siguientes capas dependerán del número de filtros de la convolución anterior.

- **ReLU**: La capa *ReLU* (del inglés *rectified linear unit layer*) es utilizada después de la operación de convolución. Se trata de una función de activación que se aplica a cada valor del mapa de características, siendo su objetivo el dotar a la red de capacidad de introducir no-linealidades. (Ver Figura 15).

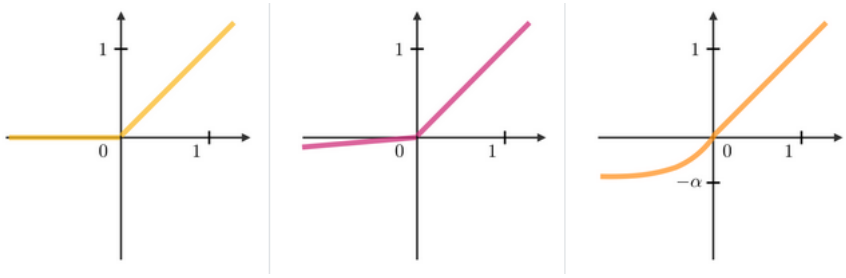


Figura 15: Funciones de activación. *ReLU* $g(z) = max(0, z)$ para introducir no linealidades al sistema, junto a *Leaky ReLU* y *ELU*, alternativas de funciones no lineales utilizadas.

- **Capa de Pooling**: Se aplica justo después de la operación convolución para reducir la dimensionalidad a la vez que proporciona a la red robustez ante las variaciones espaciales. Esta operación queda definida por la región a agregar y la función a aplicar, siendo las funciones más comunes el máximo y la media. (Ver Figura 16).

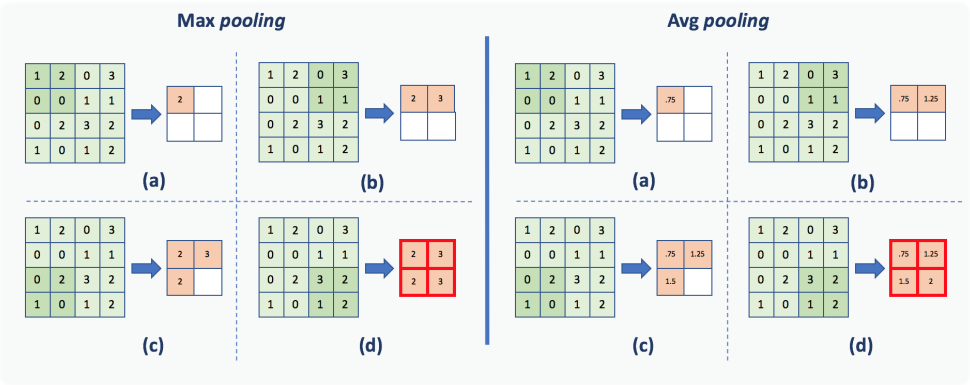


Figura 16: Operación de *pooling*. Operación máximo a la izquierda y media a la derecha..

- **Capas totalmente conectadas**: Las capas totalmente conectadas, del inglés *Fully Connected Layers*, se aplican tras la última capa de extracción de características con el propósito de clasificar; son equivalentes a una red neuronal totalmente conectada. (Ver Figura 13 b).

Las capas presentadas anteriormente son las mínimas necesarias para crear una CNN, aunque habitualmente son concatenados varios conjuntos de capa convolucional + ReLU + *pooling* antes de las capas totalmente conectadas. La arquitectura LeNet-5 propuesta por LeCun et al. [81], para

clasificación de caracteres en los años 90, es un buen ejemplo de red neuronal convolucional. LeNet-5 consta de tres capas convolucionales, dos capas de *pooling* y tres capas totalmente conectadas. Las tres capas convolucionales van seguidas de sus correspondientes funciones de activación, y las dos primeras van seguidas de la operación de *pooling*. La última capa totalmente conectada presenta una función de activación *Softmax*[10]. (Ver Figura 17).

El proceso de aprendizaje de las CNNs es similar al de las redes neuronales tradicionales, es decir, tanto los filtros de las capas convolucionales como los pesos y bias de las capas totalmente conectadas de las CNNs son aprendidos a partir de conjuntos de muestra etiquetada (aprendizaje supervisado). Este aprendizaje completo, de toda la red se basa en el concepto de retropropagación o *back-propagation* en el que para una entrada dada, la salida obtenida tras la inferencia del modelo hacia adelante o *feed-forward* y una salida esperada, se calculan los gradientes de todos los pesos de la red y se actualizan los pesos según una estrategia de aprendizaje. La actualización de dichos pesos (filtros, pesos, bias...) de manera iterativa inducen el aprendizaje de la misma para la resolución del problema concreto en el que ha sido entrenado.

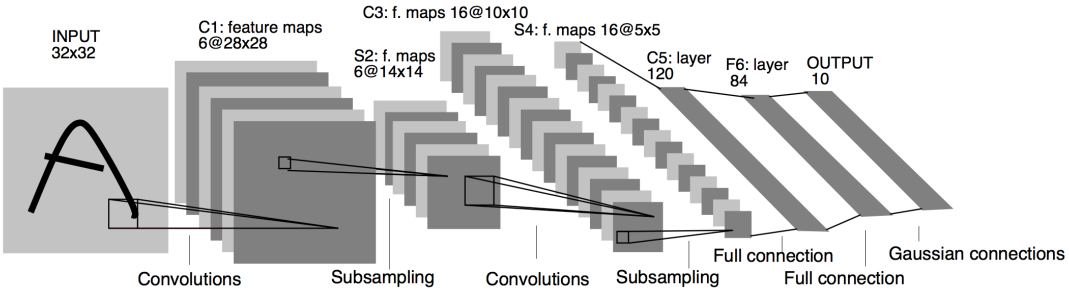


Figura 17: Arquitectura LeNet-5.

Como se ha descrito anteriormente en la sección de clasificación 2.1, las CNNs tienen dos bloques claramente diferenciados: Primero la extracción de características, obtenida a partir de una serie de operaciones de convolución, *pooling* y funciones de activación y segundo el bloque de clasificación, que se realiza en las capas completamente conectadas. Ambos bloques se aprenden de manera simultánea, durante el proceso de aprendizaje en las CNNs a diferencia de como se realizan en la visión por computador tradicional.

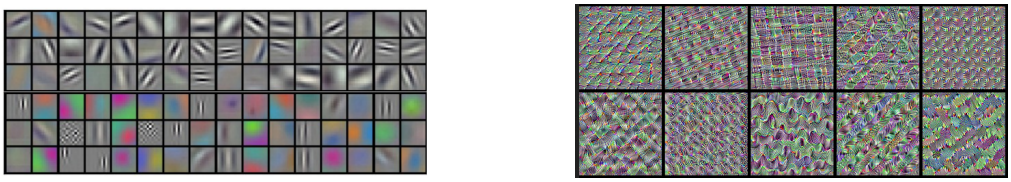
2.4.1. Extracción de características

La fase de extracción de características en las CNNs se realiza en las capas convolucionales, estas capas al convolucionar los filtros generan mapas de activación (también llamados *mapas de características*). El mapa de activación de cada capa convolucional es la entrada de la siguiente

[10]La función *Softmax* añade la restricción de que todas los valores de la última capa sumen 1 y puede ser utilizada para representar una distribución categórica

fase de convolución, siendo la entrada de la segunda fase de convolución de una CNN el mapa de características resultante de la primera convolución. Esto induce a que la red genere una relación jerárquica de características de menor a mayor nivel, de manera que la última capa de convolución la que extrae características más complejas.

En las primeras capas convolucionales la red neuronal extraen características de bajo nivel tales como bordes o texturas, como los obtenidos con filtros de *Gabor* [34] o *Maximum Response filters* [16], [119]. Conforme la red es más profunda, la complejidad de las características aprendidas por las CNNs se incrementa, de esta manera las últimas capas representan características que pueden corresponderse con objetos complejos. Zeiler y Fergus [139] y Olah et al. [93], entre otros, trabajaron en la visualización de las características aprendidas por una red. En la Figura 18 (a) podemos ver los filtros de la primera capa convolucional de una CNN, los cuales se asemejan mucho a bordes o patrones de color; en la misma Figura 18 (b) se representan 10 imágenes que activan únicamente un filtro de la última capa convolucional, mostrando las formas que son capaces de identificar dicho filtros. Esta técnica nos muestra como en la última capa convolucional una CNN es capaz de aprender patrones complejos.



(a) (b)

Figura 18: (a) Filtros de la primera capa convolucional aprendidos por Krizhevsky et al. [75], (b) Imagen que maximiza 10 filtros diferentes de la ultima capa convolucional de una CNN.

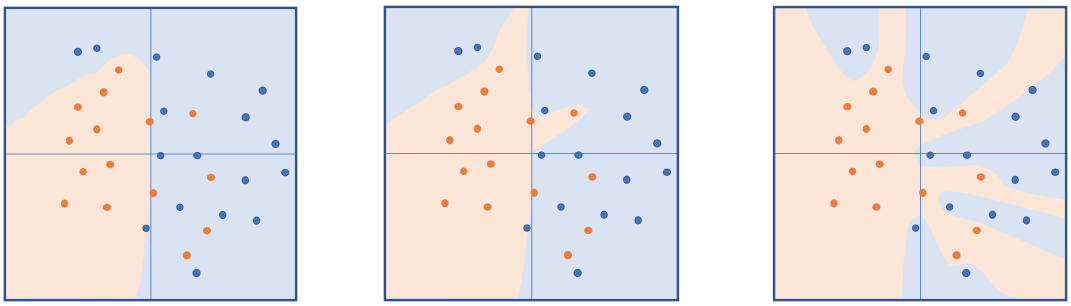
La fase de extracción de características basadas en capas convolucionales es el equivalente a la fase de extracción de características de las técnicas clásicas basadas en descriptores. De hecho, algunos trabajos se centran en la creación de características de imagen locales al igual que en los métodos clásicos pero basadas en CNNs, por ejemplo en [126] se basaron en la detección de puntos clave repetitivos, Verdieet al. [135] propusieron una tecnica basada en CNNs para estimar la orientación canónica de una característica local y en MatchNet [51] y DeepCompare [138] propusieron un modelo *end-to-end* para aprender representaciones de regiones y Yi et al. [134] propusieron un modelo *end-to-end* para detectar puntos clave, estimar la orientación y calcular descriptores.

Además de las capas convolucionales el resto de la arquitectura de la red dependerá del problema a abordar, en concreto, en clasificación de imágenes, tras las capas convolucionales hay una fase de clasificación basada en capas totalmente conectadas; sin embargo, en recuperación de imágenes a partir del mapa de características se obtiene un descriptor global de la imagen. A continuación profundizamos en los conceptos de las capas totalmente conectadas y su uso como clasificador.

2.4.2. Clasificación con CNNs

En las redes neuronales convolucionales, después de la fase de extracción de características hay una fase de clasificación basada en capas totalmente conectadas, del inglés *Fully Connected Layers*, siendo la última de ellas una capa con tantas neuronas como clases tenga el problema de clasificación multi-clase en cuestión, con una función de activación *Softmax*.

Algunas de las arquitecturas más conocidas de CNNs; VGG [113], AlexNet [75], o ResNet [54] tienen 3, 3 y 1 capas totalmente conectadas, respectivamente. De la configuración de estas capas totalmente conectadas, (numero de capas y numero de neuronas por capas), dependerá su capacidad de modelar funciones complejas. (Ver Figura 19.)



(a) (b) (c)

Figura 19: Capacidad de decisión de una red totalmente conectada dependiendo del número de neuronas ocultas. 3, 6 y 20 neuronas ocultas para (a), (b) y (c) respectivamente.

En general en las CNNs se utilizan siempre las capas totalmente conectadas como clasificador, porque resulta especialmente interesante frente a la hora de modelar funciones complejas o con una alta dimensional de características. Aunque existen estudios, como el de Janke et al. [60], que estudian la aplicación de diferentes algoritmos de clasificación a partir de características de alto nivel de las capas convolucionales de redes neuronales, demostrando en ocasiones mejor rendimiento utilizando algoritmos de clasificación como los *SVM* [124] o los árboles de decisión [102] cuando se trata de problemas de clasificación de muchas clases.

2.4.3. Recuperación de Imágenes con CNNs

Los primeros trabajos que constan de recuperación de imágenes en los que se obtienen los descriptores a partir de CNNs son del año 2014 [8], en donde se proponía generar el descriptor a partir de las capas totalmente conectadas. Posteriormente, Razavian et al. [107] estudio el rendimiento del sistema, obteniendo el descriptor en diferentes capas de las CNNs y concluyó que la mejor representación se obtiene a parir del mapa de características resultante tras la última capa convolucional. Para la transformación de dicho mapa de características en un descriptor global aplicaban una operación de agregación, o *pooling*, global a dicho mapa. Teniendo en cuenta este resultado, varios trabajos se

han centrado en obtener un descriptor con el mismo número de características como filtros tenga la última capa convolucional [7, 67, 120], o en obtener un descriptor compacto utilizando métodos como *VLAD* o *Fisher Vectors* [136, 123, 5].

Ante la ausencia en ese momento de conjuntos de datos de entrenamiento suficientemente grandes de recuperación de imágenes y teniendo en cuenta la capacidad tan grande de generalización de los modelos entrenados con *ImageNet* [109] que contiene 1.2 millones de imágenes de 1000 clases diferentes, inicialmente se utilizaron CNNs pre-entrenadas [8, 122] en dicho conjunto para el problema de clasificación de imágenes, esta manera de abordar el problema de recuperación de imágenes se denominó *pretrained single pass*.

Posteriormente, se utilizaron modelos a los cuales se hace un proceso de *Fine-tuning*[11] [5, 104, 47] utilizando un conjunto de datos de entrenamiento específicos para el problema de recuperación de imágenes. Demostrando que el proceso de *Fine-tuning* mejora mucho el rendimiento del sistema, siendo el inconveniente principal la cantidad de datos anotados que son necesarios; por ello diversos trabajos [131, 106] se centraron en la generación de dichos conjuntos.

En la aproximación de *Fine-tuning* las CNNs son entrenadas habitualmente con funciones de pérdida como *triplet loss* [47, 28] y *contrastive loss* [106] o estrategias de aprendizaje basado en métricas adaptadas [17], con el objetivo de maximizar la similitud entre clases positivas y minimizarla entre clases negativas, como Gordo et al. [47, 48] y Radenovic et al. [104].

Actualmente la mayoría de los métodos recientes de recuperación de imágenes generan una representación compacta [104, 47] de la imagen a partir de CNNs entrenadas en conjuntos específicos.

2.5. Agregación de características

En diferentes algoritmos de visión por computador se encuentra una fase de agregación de características dedicada a combinar la información de diferentes vectores de características [79, 25, 64, 140]. El objetivo de esta agregación es obtener una representación más compacta, de menor dimensionalidad y que a su vez dote al sistema de cierto grado de robustez al ruido, a las oclusiones, así como invarianza antes traslaciones, giros o transformaciones.

Un ejemplo de agregación de características en el que se obtiene un vector más compacto y robusto es la conocida como pirámide espacial [78], que se utiliza en clasificación de imágenes. En la pirámide espacial (ver Figura 20), cada pixel de la imagen está representado por un vector de características; primero se agregan los vectores de características de cada región utilizando el máximo de cada característica y después se concatenan los vectores resultantes de las regiones en un único vector. Esta agregación tan sencilla mostró mejores resultados con respecto al estado del arte [78]. Este ejemplo sirve para presentar los aspectos fundamentales que se han estudiado en la agregación

[11] *Fine-tuning* es el proceso de re-entrenar una red neuronal partiendo de un entrenamiento anterior en un dominio similar. En general este proceso es realizado cuando se quiere acelerar el proceso de entrenamiento o cuando no se dispone de muestra suficiente del problema concreto a abordar para entrenar.

de vectores de características: la función encargada de agregar los valores y la forma de representar la relación espacial de las características en la imagen.

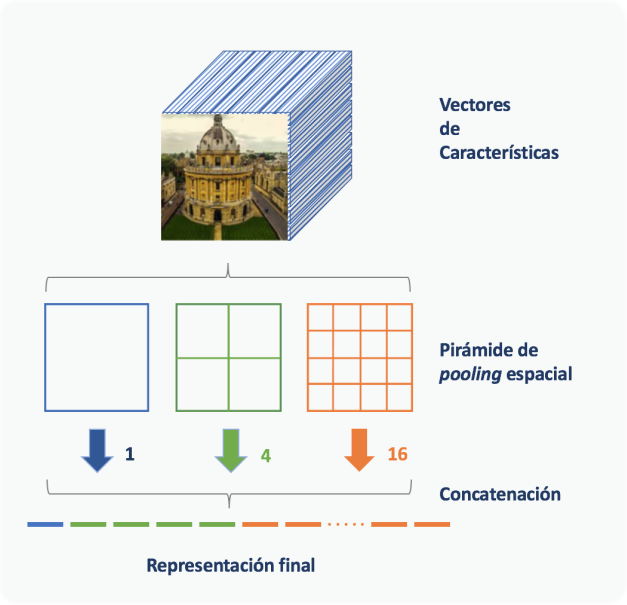


Figura 20: Metodología de agregación basada en pirámide espacial.

2.5.1. Funciones de Agregación

Una función de agregación es una aplicación no decreciente en cada argumento, $M : \mathbb{R}^n \to \mathbb{R}$ con $(n \in \mathbb{N}, n \geq 2)$. Cuando utilizamos una función de agregación para agregar vectores de características, aplicamos la función de agregación al conjunto de valores de cada característica. Por ejemplo, si tenemos un conjunto de vectores $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^k$ y los vectores tienen n características cada uno; si utilizamos la media aritmética, la componente j del vector agregado final $\boldsymbol{z}$ será:

$$z_j = \frac{1}{k} \sum_{i \in k} x_j^i \tag{I.8}$$

Comúnmente, las funciones de agregación que se utilizan son la media aritmética y el máximo (por ejemplo, en la fase de *pooling* de las CNN). Debido a la importancia de la fase de agregación, existen numerosos estudios que analizan y proponen diferentes funciones de agregación. Uno de los más importantes es el presentado por Boureau et. al. [11] y [12] donde desarrollan un estudio teórico y experimental en el que demuestran que utilizar el máximo es mejor que la media cuando las características que definen a una clase tienen poca probabilidad de activación (por ejemplo en clasificación utilizando la pirámide espacial [78]).

Este resultado fue validado experimentalmente por Koniusz et al. [73] y por Wang et al. [129]; en

ambos trabajos el operador máximo obtiene los mejores resultados cuando se utilizan vectores dispersos de características muy grandes. Además, en [12] y [73] identificaron casos en los que funciones de agregación que suavizan máximo, como el máximo esperado, obtienen todavía mejores resultados que el máximo. Pese a que los estudios de [73] y [12] estuvieron centrados en la agregración de características en el método de *Bag-of-Features*, se argumenta que las conclusiones obtenidas se pueden trasladar a la operación de *pooling* de las CNNs, ya que aplicado en capas, el modelo BoF es equivalente a una CNN. El objetivo de utilizar una función que suavice el máximo proviene del hecho de reducir la aparición de características, que no representan a la clase de la imagen, pero que tienen un valor de activación alto.

A partir de estos trabajos otros autores han propuesto otras expresiones como Kolesnikov et al. [72] que introdujeron la idea de ponderación ordenada (*weighted rank-pooling*). Pinheiro et al. [99] utilizaron una aproximación suavizada de la función máximo llamada *Log-Sum-Exp*, donde un parámetro r controla el suavizado de la función produciendo valores grandes de r una función similar al máximo y valores muy pequeños de r una función similar a la media; Graham et al. [49] introdujeron el uso del máximo estocástico y posteriormente Shi et al. [111], propusieron tres esquemas diferentes de agregación basados medias ponderadas y procesos estocásticos.

Tenenbaum et al. [118] introdujeron el concepto de *pooling* bilineal, que consiste en realizar el producto tensorial entre dos vectores, permitiendo capturar las interacciones entre todos sus elementos. Dada a la enorme dimensionalidad del vector resultante, ($250k$ elementos para dos vectores de 512 dimensiones), Gao et al. [42] presentaron el llamado *Compact Bilinear Pooling*, que presentaba resultados similares con una reducción de dimensionalidad de dos órdenes de magnitud.

Otros muchos trabajos han intentado encontrar la mejor expresión para la agregación, creando funciones paramétricas y aprendiendo el valor correcto de dichos parámetros durante el proceso de aprendizaje. Gulcehre et al. [50] presentaron el concepto llamado *Learned-Norm Pooling* utilizando el concepto de norma L_p (donde $p = 1$ corresponde a la media y el máximo con $p = \infty$) donde el parámetro p es aprendido durante el entrenamiento. Radenovic et al. [105] partieron de la idea de media generalizada [29] para introducir el concepto de *generalized-mean (GeM) pooling*). En [46] Goodfellow et al.interpretaron la función de *pooling* como una función de activación no lineal, introduciendo el concepto de *maxout unit*, el cual agrega un conjunto de respuestas no lineales o salidas de neuronas actuando como una agregación de funciones de activación lineales.

2.5.2. Agregación de características teniendo en cuenta la posición espacial

La información espacial es clave en la percepción visual humana y por consiguiente en visión por computador. Por ejemplo, a la hora de determinar si un objeto es de una clase u otra es necesario conocer la posición absoluta en la imagen de determinadas características o la posición relativa entre diferentes características. Por tanto, en muchos algoritmos de visión por computador se realiza la agregación de características teniendo en cuenta la posición espacial de dichas características en la

imagen. El resultado de esta agregación, teniendo en cuenta la posición espacial, es un vector que agrupa la información de la región sobre la que se ha aplicado y además tiene en cuenta las relación espacial entre dichas características.

El caso más sencillo es la media ponderada, en la que cada pixel o posición de la imagen tiene un peso dependiendo de lo importante que sea, resultando en un vector con el mismo número de características que los vectores que se agregan. Existen multitud de trabajos que estudian diferentes formas de calcular el factor de ponderación espacial. Por ejemplo, en Babenko et. al. [7] utilizaron una ponderación en función de la distancia al centro de la imagen para agregar el último mapa de activaciones de una CNN; Kalantidis et. al. [67] propuso una ponderación espacial en función de los valores de las propias activaciones del mapa de características. En el trabajo de Lanskar y Kannala [76] se estudia el uso del *saliency* y en el trabajo de Jimenez et. al. [65] los *Class Activation Maps* para generar los factores espaciales.

Existen otro tipo de agregaciones en las que se obtiene una nueva representación de las características, diferente a los vectores que agregan, como por ejemplo la matriz de *Co-Ocurrencias*. A principios de los 1970s se publicó uno de los principales trabajos en caracterización de texturas, conocido como Matriz de *Co-Ocurrencias* de niveles de gris [52]. La matriz de *Co-Ocurrencias* representa la frecuencia con la que un nivel de gris se produce al lado de otro nivel de gris. Basándose en esta idea de *Co-Ocurrencia* se han desarrollado muchos trabajos que agregan características como el de Yang and Newsam [133], que caracteriza la dependencia espacial de las características visuales o el reciente de Shih et al. [112] que ha propuesto una nueva representación de *Co-Ocurrencias* para capas convolucionales de una CNN.

3. Motivación

Una vez presentados los principales conceptos sobre los que está centrada esta memoria, nos planteamos los siguientes problemas abiertos que motivan el presente proyecto de tesis:

- La generación y la selección de las características idóneas de imágenes para resolver un problema clasificación es una tarea compleja y clave para el rendimiento final del sistema de clasificación, tanto es así, que desde hace unos años hay numerosos trabajos de la literatura especializada que estudian la creación de características de diferente naturaleza con una mayor capacidad discriminativa y de generalización. Es importante señalar que la elección de dichas características en clasificación de imágenes no es sencilla, esto es debido principalmente a que las características que nos permiten discriminar entre dos clases pueden no ser discriminativas entre otro par de clases, provocando que en determinados casos nos interese una característica en concreto, por ejemplo la forma, y en otros casos nos interese una característica diferente, como por ejemplo el color.

  Una aproximación común para resolver este problema es agregar características de diferente

naturaleza. Habitualmente esta agregación se realiza a través de la concatenación de diferentes descriptores. Esto provoca que la complejidad y el número de parámetros del modelo aumente significativamente, y no resolviendo el problema de robustez ante conjuntos de datos no balanceados (una de las clases, mayoritaria, está sensiblemente más representada que el resto de clases). Por ello, resulta interesante estudiar nuevas técnicas de agregación que permitan la combinación de características bajo un esquema de complejidad reducida junto a un profundo estudio empírico para el problema de clasificación de imágenes.

- La operación de agregación de descriptores, o *pooling* espacial, es un paso crucial en diferentes técnicas visión por computador, como en las CNNs o en el método BoF. La operación de *pooling* genera una representación compacta a partir de los descriptores vecinos de una región dada (local o global). Esta representación proporciona robustez ante problemas como el ruido, oclusiones e invarianzas ante pequeñas traslaciones y rotaciones; por tanto, la operación de *pooling* es capaz de agregar la información relevante y de eliminar la información irrelevante.

  La operación de *pooling* idónea para cada problema depende del problema a abordar, de la naturaleza de las imágenes, de las características extraídas, de la aproximación dada (como BoF o CNNs) o incluso de la arquitectura de red utilizada. Esto hace que no sea sencillo seleccionar la operación óptima en cada caso, por lo que muchas veces esta operación se define como un hiper-parámetro adicional en la fase de experimentación, siendo las funciones de *pooling* más utilizadas las operaciones máximo y media. Por tanto, creemos interesante profundizar en si esta función de *pooling* puede ser aprendida con una función de *pooling* paramétrica, donde el máximo y la media sean casos particulares de la misma, de manera que su selección sea la óptima para cada problema.

- En el problema de recuperación de imágenes es crucial obtener descriptores similares para imágenes diferentes de una misma instancia. En el estado del arte, es habitual generar una representación compacta de cada imagen utilizando CNNs (aplicando una operación de *pooling* al mapa de características resultante de la última convolución). Al agregar todo el mapa de características en un único descriptor compacto, con funciones de *pooling* como la media, se agregan todas las características presentes en la imagen, sean estas representativas o no. Por ejemplo, si queremos representar un edificio, no nos interesa incluir la información relativa al cielo o las personas que puedan aparecer en la imagen. Esto dificulta la generación de un descriptor invariante cuando estas condiciones cambian, afectando al rendimiento final del sistema. Por ello, pensamos que es necesario estudiar esquemas de agregación que permitan agregar en el descriptor final la información relativa al objeto principal de la imagen, descartando la información irrelevante.

- El proceso de describir o codificar las características en las CNNs se realiza a partir de las activaciones de las capas convolucionales, generando un mapa de características tras cada

capa convolucional. En esta codificación no se tiene en cuenta la interdependencia espacial entre las características del propio mapa de características. Creemos que esta relación espacial es importante ya que, de manera abstracta, podría reflejar información relevante para generar una representación mejorada. Por ejemplo, la relación espacial entre la características puerta y ventana existentes para una casa puede ayudar a distinguir entre casas diferentes, más allá de únicamente codificar su presencia. Pensamos que esta representación llamada representación de *Co-Ocurrencias* puede ser especialmente interesante en casos como la recuperación de imágenes donde todas las activaciones son agregadas en un único vector y se pierde la representación en tres dimensiones del mapa de características. Adicionalmente la interdependencia entre características puede ser diferente en cada clase de imagen. Por ello consideramos importante generar esta representación de *Co-Ocurrencias* de manera personalizada o aprendida para cada problema. Con el mismo ejemplo de antes, puede ser interesante capturar la relación entre puerta y ventana, pero no entre ventana y cara en una imagen con personas y edificios. Por tanto, nos planteamos formular un método de caracterización de *Co-Ocurrencias* para CNNs que a su vez pueda ser aprenderse, con el objetivo de mejorar la representación final de la imagen en el problema de image retrieval.

4. Objetivos

El objetivo principal de esta memoria es: *estudiar técnicas de agregación de información para mejorar el rendimiento en diferentes problemas de visión por computador, como son los problemas de clasificación multi-clase y el problema de recuperación de imágenes*. La memoria está organizada en torno a cuatro grandes objetivos, uno por cada problema abierto expuesto en la sección anterior:

- *Estudiar y proponer un método de agregación de información para mejorar la precisión de problemas de clasificación de imágenes multi-clase, basado en las estrategias de descomposición multi-clase OVO y OVA*. Inicialmente utilizamos los métodos de ensembles para problemas de clasificación multiclase como mecanismo para agregar información de diferente naturaleza a la decisión final; en concreto, en los métodos de descomposición One-vs-All y One-vs-One. Planteamos ponderar la decisión de dichos ensembles a partir de la similitud de una imagen dada con respecto a las diferentes clases posibles a clasificar. Adicionalmente, con el objetivo de estudiar el impacto de la agregación de información, ideamos formular diferentes esquemas de agregación así como experimentar con características de diferente naturaleza utilizadas para clasificación de imágenes, como son: *Histogram of Oriented Gradients* (HOG), *Fuzzy Color Histograms* (FCH), *Bag of Features* (BoF) o matrices de *Co-Ocurrencias*.
- *Desarrollar un método que permita el aprendizaje de la operación de pooling para el problema de clasificación de imágenes multi-clase*. Proponemos crear una función de agregación paramétrica que se aprenda como operador de *pooling*, en la que los operadores máximo y media

sean casos particulares de nuestra función. Planteamos estudiar este nuevo operador de *pooling* utilizando diferentes técnicas de imagen (como son las CNNs y el método BoF), diferentes esquemas de agregación por región (local y global), así como diferentes esquemas de aprendizaje de la función de *pooling* (aprendizaje por capa, canal, región y la combinación de ellas). Consideramos interesante analizar la robustez de los modelos de clasificación obtenidos con funciones de *pooling* entrenadas. Adicionalmente, planteamos realizar una experimentación intensiva donde podamos evaluar la mejora en precisión y en coste computacional de utilizar esta aproximación.

- *Proponer un método de agregación capaz de seleccionar o dar más importancia a las activaciones más representativas del mapa de características de CNNs para el problema de recuperación de imágenes.* Planteamos realizar un proceso similar al que hacemos los humanos cuando descartamos la información no relevante de una imagen. Nuestra hipótesis es que esto puede realizarse agregando únicamente las características que se corresponden a la región de la imagen en la que se encuentra el objeto. No es trivial identificar el objeto principal de una imagen en determinadas ocasiones, por este motivo planteamos el hacerlo a través de una medida específica denominada prominencia (*saliency* en ingles).

- *Crear un método que permita caracterizar la dependencia espacial entre las características de CNNs para mejorar el rendimiento en el problema de recuperación de imágenes.* Planteamos el desarrollo de un nueva representación de *Co-Ocurrencias* capaz de capturar la correlación espacial entre características, y generar diferentes esquemas de agregación para añadir esta información de manera efectiva a la representación final. Adicionalmente consideramos interesante proponer un método capaz de aprender la representación de *Co-Ocurrencias* idónea para cada problema, de manera que pueda ser aprendido durante el proceso de aprendizaje de la CNN.

5. Discusión de resultados

En esta sección presentamos una breve descripción de cada uno de los artículos que componen la presente memoria. Además, incluimos los principales resultados obtenidos en su realización.

5.1. Combinación de características a través de ensembles ponderados para clasificación de imágenes

En este primer trabajo nos centramos en mejorar la precisión en problemas de clasificación multi-clase de imágenes a partir de la agregación de información o características. Una aproximación habitual en los problemas multi-clase es dividirlo en sub-problemas binarios de clasificación y agregar

la respuesta de sus clasificadores base (One-vs-All (OVA) o One-vs-One (OVO)). Esta agregación de los clasificadores base suele ser clave para la obtención de resultados óptimos.

Teniendo en cuenta estos dos aspectos en este trabajo proponemos combinar la información de vectores de características de naturaleza diferente en la fase agregación del ensemble multi-clase. Para ello realizamos la combinación de los vectores de características a través de la ponderación de los clasificadores base que forman el ensemble. Los clasificadores base del ensemble se entrenan con un tipo de vectores de características y posteriormente, los ponderamos utilizando otro tipo diferente de vectores de características. De esta forma, combinando la información de ambos vectores mejoramos el resultado final en clasificación de imágenes.

Más concretamente proponemos las siguientes soluciones:

- Una metodología para obtener una distancia, entre una imagen y un conjunto de imágenes que definen una clase, a partir de los descriptores de la imagen.
- Presentamos dos expresiones para calcular la importancia de un clasificador base, una para el ensemble tipo OVO y otra expresión para OVA. Esta importancia o peso lo obtenemos a partir de la distancia de una imagen a clasificar respecto a las clases con las que se han entrenado los diferentes clasificadores base.
- Proponemos tres posibles ensembles o escenarios de ponderación. Los dos primeros se basan en ponderar los clasificadores base de las metodologías OVO y OVA según la importancia calculada de cada clasificador base. El tercer ensemble consiste en utilizar una red neuronal para fusionar las salidas de los clasificadores base, es decir la matriz de *scores*, con las distancias de cada imagen a las diferentes clases del problema.

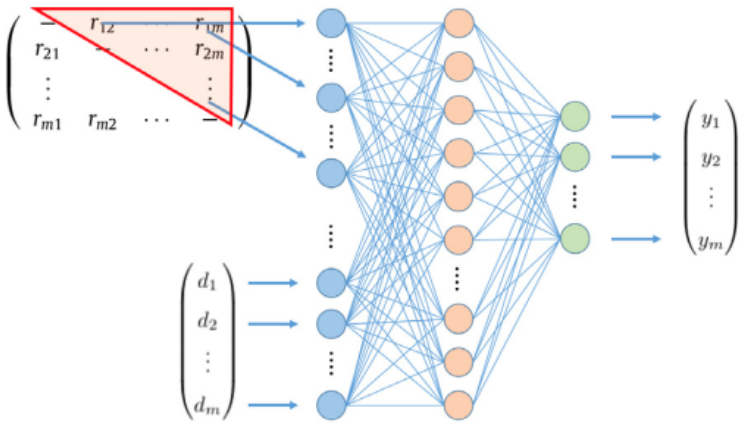


Figura 21: Esquema de combinación de la salida de los clasificadores base y de las distancias de la imagen a cada una de las clases posibles basado en redes neuronales .

Para analizar y evaluar la eficacia de nuestra propuesta, hemos desarrollado un estudio experimental que aborda las siguientes cuestiones:

- Evaluamos en los conjuntos CIFAR-10 y STL-10, los resultados obtenidos con la aproximación OVA y OVO con respecto a las tres diferentes versiones de ponderación. En este primer experimento calculamos las distancias de ponderación utilizando las mismas características (BoF) con las que fue entrenado el clasificador llamado (DRW). A su vez también estudiamos la generación de nuevas representaciones a partir del descriptor BoF como son una representación ponderada espacialmente (SDCRW) y una representación de *Co-Ocurrencias* también a partir del descriptor BoF y denominada COOC-DRW.

- En un segundo experimento evaluamos de nuevo los conjuntos CIFAR-10 y STL-10, en los tres mismos escenarios de ponderación, pero en este caso proponemos obtener los pesos de la ponderación a partir de características diferentes a las utilizadas para los clasificadores base. Las características utilizadas son HOG (Histograms of Oriented Gradients), RGB (Color en formato RGB) y FCH (Fuzzy Color Histogram).

A partir de los resultados obtenidos en el estudio experimental obtenemos las siguientes conclusiones:

- La agregación de características es una opción común en problemas donde hay numerosas clases diferentes con condiciones muy variadas y por ello es positivo combinar diferentes descriptores a través de algún método de agregación.

- Hemos demostrado experimentalmente el beneficio del uso de la estrategia propuesta para agregación de características, sin apenas incrementar el número de parámetros del modelo y siendo computacionalmente más eficiente que la aproximación habitual basada en la concatenación de vectores.

- En la experimentación presentamos mejoras en los conjuntos utilizados, obteniendo incluso resultados de clasificación similares al estado del arte para el conjunto STL-10 pese a utilizar un modelo con muchos menos parámetros.

- Obtenemos los mejores resultados cuando utilizamos el esquema de ponderación basado en redes neuronales.

- Adicionalmente concluimos que resulta beneficioso agregar características de diferente naturaleza frente a hacer la ponderación con las mismas características con las que fueron entrenados los clasificadores base.

El estudio completo ha sido publicado en:

- Forcén JI, Pagola M, Barrenechea E, Bustince H. Combination of features through weighted ensembles for image classification. Applied Soft Computing. 2019 Nov 1;84:105698

5.2. Aprendiendo operaciones de *pooling* basadas en medias ponderadas ordenadas para clasificación de imágenes

La operación de *pooling* espacial es clave en las técnicas más comunes de clasificación de imágenes, como la pirámide espacial o las CNNs. La selección del operador de agregación idóneo no es trivial y depende de la naturaleza de cada problema. Esto hace que en la mayoría de las ocasiones dicho operador sea decidido durante la fase de experimentación, como un parámetro adicional.

Dada esta dificultad y con el objetivo de mejorar el rendimiento en problemas de clasificación multiclase de imágenes, proponemos un nuevo operador de *pooling* donde la operación de *pooling* óptima se aprenda durante la fase de entrenamiento junto al resto de parámetros.

Más concretamente las contribuciones en esta publicación son:

- Introducimos un nuevo operador de *pooling* basado en medias ponderadas ordenadas al que llamamos OWA *pooling*.
- Proponemos un método para aprender los pesos del OWA *pooling*, de manera que la función de agregación pueda aprenderse, siendo la operación máximo y media casos particulares de la agregación presentada.
- Analizamos este método OWA *pooling* para el problema de clasificación de imágenes con dos aproximaciones diferentes: el método *Bag-of-Features* y CNNs.

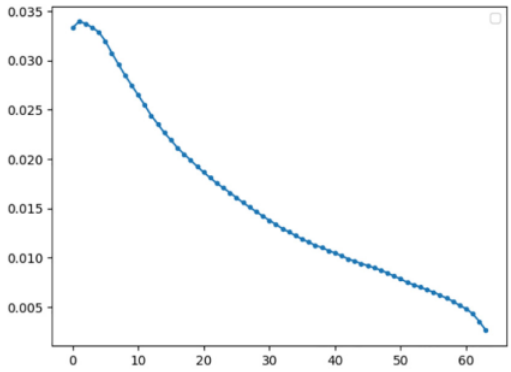


Figura 22: Pesos aprendidos aplicando OWA *pooling* a la operación de *pooling* global en un caso concreto (Red VGG entrenada con el conjunto 15-Scenes). Podemos apreciar que la función resultante es una función intermedia a las funciones máximo y media.

Todo esto nos ha llevado a realizar un extenso estudio experimental para evaluar nuestra propuesta:

- En un primer experimento comprobamos los resultados aplicando la operación de OWA *pooling*

propuesta en el método *Bag-Of-Features* para clasificación de imágenes (en el conjunto 15-Scenes). En este caso la operación de *pooling* es utilizada para agregar todos los descriptores BoF a partir de descriptores SIFT.

- A continuación valoramos la aplicación del operador OWA *pooling* en CNNs. Evaluamos en los conjuntos CIFAR-10 y CIFAR-100 diferentes arquitecturas conocidas, utilizando diferentes operaciones de *pooling*: las originales de la red, el máximo, la media y diferentes variaciones del OWA *pooling* propuesto, en las que plantemos el aprendizaje por capa, por capa y canal junto a un término de regularización en el aprendizaje.

  1. Estudiamos la combinación del OWA *pooling* con diferentes técnicas de regularización como son *random erasing* y *dropout estructurado* (también llamado *DropBlock*).
  2. Valoramos la robustez del modelo a variaciones de giro y traslaciones cuando es utilizado el operador de *pooling* original y el OWA, con el objetivo de valorar si el añadir parámetros a la función de *pooling* puede producir al sistema a un mayor sobre-ajuste.

- En un tercer experimento estudiamos aplicar el operador OWA *pooling* como operación de *pooling* global en arquitecturas de CNNs como VGG13 y MobileNet. Este experimento es motivado porque habitualmente en las operaciones de *pooling* global los valores a agregar son muchos más que en las operaciones locales donde solo son 2x2 o 3x3 valores.

  En este caso usamos los dataset 15-Scenes y Caltech-UCSD Birds (CUB200). Valoramos las operaciones de *pooling* Máximo, Media, OWA y de nuevo diferentes variaciones del mismo como son OWAco, OWAnr y OWAC. En las que plantemos apredizaje del operador o del operador por cada canal.

- Adicionalmente analizamos los valores de los pesos aprendidos en la operación de *pooling* para los experimentos en CNNs. (Experimentos 2 y 3).

A partir de los resultados obtenidos en el estudio experimental obtenemos las siguientes conclusiones:

- El operador de *pooling* propuesto basado en medias ponderadas ordenadas proporciona resultados iguales o mejores tanto en el método BoF como en CNNs para la extensa experimentación realizada.

- Podemos determinar por tanto que la operación OWA *pooling* puede considerarse como una operación de *pooling* auto-configurable que puede utilizarse para evitar el tener que determinar dicha operación de manera experimental.

- Demostramos que el OWA *pooling* es más robusto ante pequeñas variaciones de rotación y traslación de la imagen, siendo además el OWA *pooling* complementario a las técnicas habituales de regularización.

- Concluimos que el uso del operador OWA *pooling* en arquitecturas grandes como función de *pooling* tras cada capa convolucional no es aconsejable por el tiempo de procesamiento que añade. Sin embargo el uso del OWA *pooling*, como operador *pooling* global, demuestra ser mejor que el resto de aproximaciones.

El desarrollo completo ha sido publicado en el siguiente artículo:

- Forcén JI, Pagola M, Barrenechea E, Bustince H. Learning ordered pooling weights in image classification. Neurocomputing. 2020 Oct 21;411:45-53.

5.3. Agregación de características profundas en recuperación de imágenes basadas en la detección de objetos

En recuperación de imágenes cada imagen es representada de manera compacta como un único descriptor, el cual es generado a partir de la agregación del último mapa de características de una CNN. En este trabajo nos centramos en mejorar dicha representación final, mejorando el proceso de agregación del mapa de características. Para ello proponemos agregar de manera ponderada los objetos más prominentes de la imagen que han sido detectados a través de la medida de *Saliency*. De esta manera, el objeto principal de la imagen tendrá mayor influencia en la representación final. Más concretamente las contribuciones en esta publicación son:

- Proponemos un esquema de ponderación para la generación de representaciones compactas basado en la detección del objeto principal de la imagen.
- Extendemos la metodología de ponderación, a la representación o agregación basada en múltiples regiones.
- Proponemos el uso de plantillas espaciales en combinación con otras medidas de ponderación.
- En la experimentación demostramos la efectividad de este método en la categoría *pre-trained single-pass*.

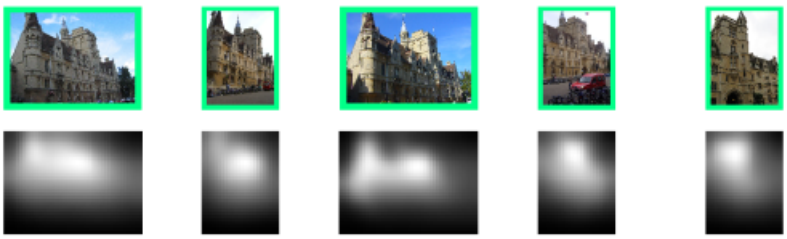


Figura 23: Ejemplos de la ponderación espacial a partir de la medida de *saliency* para diferentes imágenes de una misma instancia del conjunto Oxford-5k.

Todo esto nos ha lleva a profundizar en este trabajo en los siguientes puntos:

- Experimentamos en tres conjuntos conocidos de recuperación de imágenes Oxford-5k, Paris-6k, y Holidays; comparamos la precisión de nuestro método basada en la ponderación espacial basada en la medida *saliency* frente a la agregación uniforme o la agregación basada en el valor de las propias activaciones. Incluimos en la experimentación el estudio de la influencia de las máscaras espaciales.

- En un segundo experimento evaluamos en los mismo conjuntos el desempeño cuando utilizamos la aproximación basada en múltiples regiones (3 escalas diferentes - 20 regiones) como método de extensión de la agregación ponderada presentada en el primer experimento.

A partir de los resultados obtenidos en el estudio experimental obtenemos las siguientes conclusiones:

- En este trabajo presentamos un método de ponderación de las regiones más importantes de una imagen, descartando o reduciendo la influencia del fondo de la imagen. Esto se traduce en una representación compacta mejorada para el problema de recuperación de imágenes.

- En la comparación con métodos del estado del arte, obtenemos resultados similares en la categoría *pre-trained single pass* con respecto a métodos comunes *CROW* [67], *BLCF-SalGAN* [90], *R-MAC* [7], *CAM* [65], presentando mejoras sustanciales cuando es combinado con expansión de la consulta (QE).

El trabajo asociado completo ha sido publicado en:

- Forcén JI, Pagola M, Barrenechea E, Bustince H. Aggregation of deep features for image retrieval based on object detection. In Iberian Conference on Pattern Recognition and Image Analysis 2019 Jul 1 (pp. 553-564). Springer, Cham.

5.4. Co-ocurrencia de características de redes neurales convolucionales aplicadas a recuperación de imágenes

En este trabajo profundizamos de nuevo en la agregación de características, proponiendo en este caso una nueva representación capaz de caracterizar la correlación espacial entre características. Esta representación la aplicamos al problema de recuperación de imágenes donde es especialmente importante capturar la dependencia espacial entre activaciones o características, teniendo en cuenta que dicha relación espacial habitualmente se pierde en las operaciones de *pooling*. El objetivo final en este trabajo es conseguir una representación mejorada para el problema de recuperación de imágenes que proporcione una mayor precisión. Las contribuciones en esta publicación son:

- Proponemos una nueva representación de *Co-Ocurrencias* basada en las características de CNNs. Esta representación de *Co-Ocurrencias* es capaz de representar la correlación de cada

característica del mapa de activaciones de una CNN con respecto al resto de características para una posición y una región dada.

- Presentamos una implementación basada en una operación convolucional con un filtro capaz de capturar las relaciones espaciales entre las activaciones de diferentes canales. Esta implementación es computacionalmente mejor que las propuestas actualmente en el estado del arte.
- El filtro para capturar las *Co-Ocurrencias* es denominado filtro de *Co-Ocurrencias* y permite aprenderse a través del entrenamiento de la propia CNN, de manera que la agregación de las interacciones entre canales pueda aprenderse según cada problema concreto.
- Utilizamos dos métodos de agregación lineal y bilineal para fusionar la información del mapa de activaciones de la CNN, con el mapa *Co-Ocurrencias* en un único descriptor global.
- Demostramos con una experimentación exhaustiva que con nuestra propuesta obtenemos una representación mejorada para el problema de recuperación de imágenes.

(a) (b)

Figura 24: (a) *Co-Ocurrencia* de dos características diferentes en el mapa de activaciones de una CNN. (b) Esquema de representación basado en la agregación de la información de *Co-Ocurrencias* para el problema de recuperación de imágenes.

Para evaluar la eficiencia de nuestra propuesta hemos realizado el siguiente estudio experimental:

- Estudiamos la correlación entre las representaciones de *Co-Ocurrencias* para instancias iguales de los conjuntos Oxford-5k y Paris-6k, con el objetivo de validar que dos clases similares tienen representaciones de *Co-Ocurrencias* similares.
- Comparamos el método propuesto y denominado método directo de *Co-Ocurrencias*, donde las *Co-Ocurrencias* son calculadas con un filtro fijo en 5 conjuntos comunes de recuperación de imágenes; Oxford, Paris, ROxford, RParis y Holidays. Estudiamos este método directo con dos esquemas de agregación (lineal y bi-lineal), para generar un descriptor único para la búsqueda de recuperación de imágenes. Adicionalmente utilizamos técnicas comunes como ponderaciones espaciales o expansión de la consulta.

  - En la agregación lineal comparamos el resultado agregando el tensor de *Co-Ocurrencias* con el método linear weighting *pooling* con respecto a otros métodos y agregaciones lineales. $ChCO - SC_T(SpTD)(SpCt)$.
  - En la agregación bilineal estudiamos la precisión del modelo agregando las *Co-Ocurrencias* a través de *pooling* bilineal, frente a la agregación bilineal de las propias activaciones. También probamos la combinación de la máscara espacial C_T.

- En un segundo experimento evaluamos los resultados aprendiendo el filtro de *Co-Ocurrencias*, en un modelo con el resto de capas congeladas dentro de un esquema de aprendizaje de arquitectura siamesa y entrenado con tuplas de imágenes. Comparamos estos resultados con el estado del arte en métodos entrenados.
- Finalmente comparamos nuestro método de *Co-Ocurrencias* con el de Shih et al. [112], obteniendo resultados mejores con respecto al coste computacional y con respecto a la precisión final del modelo.

A partir de los resultados del estudio experimental obtenemos las siguientes conclusiones:

- Hemos propuesto una expresión que captura la dependencia espacial de las activaciones de una CNN a través de la agregación de sus activaciones en forma de representación de *Co-Ocurrencias*.
- La combinación del tensor de *Co-Ocurrencias* con el mapa de activaciones a través de dos técnicas distintas, una basada en la agregación lineal de ambas y otra en la bilineal mejora la representación lineal frente a utilizar únicamente los mapas de activaciones.
- Experimentalmente hemos demostrado que incluyendo la representación de *Co-Ocurrencias* se mejoran los resultados con respecto a un método estandar, por lo tanto las *Co-Ocurrencias* permiten generar una representación final mejorada.
- La implementación que proponemos en forma de filtro de *Co-Ocurrencias* permite aprender la representación de las *Co-Ocurrencias* adaptada a cada problema, mejorando aún más los resultados finales del problema de recuperación de imágenes.

El desarrollo completo de este trabajo ha sido publicado en:

- Forcen JI, Pagola M, Barrenechea E, Bustince H. Co-occurrence of deep convolutional features for image search. Image and Vision Computing. 2020 Mar 25:103909.

6. Conclusiones

En esta sección, presentamos las conclusiones obtenidas a partir de la investigación realizada a lo largo de los diferentes trabajos descritos en esta memoria. Las propuestas realizadas han tenido por objeto mejorar la precisión en el problema de clasificación de imagen multi-clase y en el problema de recuperación de imágenes. Hemos planteado estas propuestas desde la mejora de las fases agregación de características, presentes en los problemas citados, esto que nos ha llevado a profundizar en las propias operaciones de agregación, en los métodos de agregación de información, en la generación de nuevas representaciones a partir de una representación dada y en la combinación de ellas.

Hemos agregado información en los modelos OVO y OVA a través de la ponderación de los clasificadores base y mediante una MLP para el problema de clasificación de imagen multi-clase. También hemos propuesto la agregación de información en recuperación de imágenes a través de esquemas de agregación lineal y bi-lineal para generar un descriptor compacto. En cuanto a las operaciones de agregación, hemos propuesto el uso de medias ponderadas ordenadas como *pooling* auto-configurable, basado en medias ponderadas ordenas, donde las operaciones más comunes son casos particulares de la misma. Esto permite infinitas funciones de *pooling*, pudiendo auto-seleccionarse la función de *pooling* óptima para cada problema concreto durante el proceso de aprendizaje.

También hemos propuesto un método de agregación de información para el problema de recuperación de imágenes basado en la detección o localización de la regiones relevante de la imagen. Hemos trabajado en una nueva representación de información, en la que somos capaces de capturar la correlación espacial entre características. Está representación de *Co-Ocurrencias* permite ser aplicada a CNNs así como el aprender la propia representación.

Atendiendo a los resultados experimentales, hemos demostrado como la agregación de información de diferente naturaleza como la matriz de *Co-Ocurrencias* o características de diferente naturaleza (HOG, FCH, RGB) a través de los esquemas de agregación previamente planteados producen mejoras significativas en los resultados de diferentes problemas de visión por computador como son la clasificación de imágenes y el problema de recuperación de imágenes.

Experimentalmente hemos demostrado el valor de todos estos aportes en aproximaciones de visión por computador tradicional y moderna de manera exhaustiva. A modo de resumen, las contribuciones más relevantes de la presente tesis son las siguientes:

- La agregación de información proporciona un resultado mejorado en problemas de imagen, para ello es positivo combinar o agregar información de diferente naturaleza. Nosotros hemos propuesto un método de agregación que tiene en cuenta la similitud de una imagen a un clase, con un coste computacional muy bajo, sin apenas incrementar el número de parámetros del modelo y hemos demostrado la mejora proporcionada por el mismo tanto con características de la misma naturaleza como diferente.

- El operador de *pooling* óptimo para cada problema depende de múltiples factores. Por ello,

proponemos una operación de *pooling* auto-configurable, demostrando resultados similares o mejores a las funciones más comunes, demostrando a su vez resultados más robustos frente a variaciones de rotación y traslación de las imágenes.

- En recuperación de imágenes la fase de agregación de información en un representación compacta es clave, el ser capaces de detectar las características más relevantes a agregar nos permite un mejor desempeño del sistema.
- La información o relación espacial entre características contiene información muy valiosa para poder mejorar en problemas de visión por computador. Esta relación desaparece cuando el mapa de características es agregado en un único vector, de ahí la importancia de poder capturar dicha dependencia, codificarla y agregarla a la representación final mediante nuestra propuesta de tensor de *Co-Ocurrencias*.

7. Líneas futuras

Finalizamos esta primera parte de la memoria presentando las líneas futuras de investigación que surgen a partir de los trabajos realizados en la presente tesis.

Desarrollar el concepto de ponderación basada en la distancia de una imagen a una clase para conjuntos no balanceados. Cuando un clasificador o clasificadores son entrenados en conjuntos desbalanceados dicho clasificador tiende a estar sesgado hacia la clase mayoritaria. La metodología propuesta en el primer trabajo, en la que los clasificadores son ponderados en función de las distancias de la imagen a procesar con respecto a cada clase, es especialmente interesante para poder reducir o compensar el efecto producido por el sesgo. Por ello en este sentido proponemos estudiar de manera intensiva esta línea en conjuntos claramente desbalanceados, tanto en métodos clásicos, como especialmente en métodos basados en CNNs, en este caso se propone inferir la distancia a la clase en función de la representación compacta que tendríamos tras la última capa convolucional. Adicionalmente sugerimos explotar este concepto junto al concepto de *metric learning* en CNNs donde guiamos a la red para que las representaciones compactas de las imágenes de una misma clase se representen en una región determinada del espacio.

Estudio de las técnicas de agregación en la fase de expansión de la consulta, utilizada en problemas de recuperación de imágenes. La fase de expansión de la consulta es realizada en la mayoría de las aproximaciones al problema de recuperación de imágenes, justo después de la primera consulta. Esta técnica es utilizada para crear una segunda consulta mejorada, que contenga información adicional de las imágenes más similares. Este proceso normalmente es realizado agregando los descriptores de las imágenes más parecidas, siendo el caso más sencillo la media. Consideramos

interesante profundizar en el estudio de diferentes técnicas para este problema de agregación. Creemos que pueden extenderse técnicas utilizadas en esta memoria, como la agregación ponderada, el concepto de distancia de una imagen a una clase, y la técnica de agregación aprendida en base a una función paramétrica.

Profundizar en la aplicación de la representación de *Co-Ocurrencias* a los problemas de *fine-grained-classification* El problema de Fine-grained classification, es un subcampo del campo de clasificación de imágenes y consiste en clasificar entre categorías detalladas de una misma clase, un ejemplo es el conjunto Caltech-UCSD Birds 200 [128] que contiene 200 tipos diferentes de especies de pájaros. En esta posible línea de trabajo, proponemos profundizar en el uso de la representación de *Co-Ocurrencias* en problemas de esta naturaleza, donde la diferencia entre una clase y otra es mínima y la correlación o dependencia espacial entre características puede ser determinante. Además consideramos interesante valorar la aplicación de nuestra representación de *Co-Ocurrencias* no solo tras la última capa convolucional, sino aplicado a varios o a todos los mapas de características de las CNNs. Este último punto implicaría a su vez profundizar en un nuevo esquema de agregación de ambas representaciones.

Profundizar en la agregación de información espacial en la representación compacta del problema de recuperación de imágenes. En algunas aproximaciones al problema de recuperación de imágenes, es habitual realizar una fase de refinamiento de la búsqueda llamada verificación espacial, con el objetivo de mejorar la precisión. La verificación espacial permite descartar aquellos casos en los que dos imágenes tienen representaciones compactas similares, pero la geometría o la relación de sus descriptores locales es diferente. El motivo de que dos imágenes con geometría diferente tengan descriptores globales similares es debido a que durante la fase de agregación, de los descriptores locales en uno global, se pierde la información espacial de la imagen. Por ello proponemos profundizar en el estudio de esta fase proponiendo uno o varios métodos capaces de caracterizar la dependencia espacial entre características, de manera similar a la representación de *Co-Ocurrencias*, pero con carácter global, y estudiar el posible beneficio en comparación con la fase de verificación espacial convencional basada en el método iterativo *RANSAC (RAndom SAmple Consensus)* [33] a partir de los descriptores locales de la imagen.

Estudio de la aplicación del método de *Co-Ocurrencias* para problemas de síntesis de texturas o transferencia de estilo. En visión por computador el campo de síntesis de texturas (*Texture Synthesis*) [31] se dedica a generar contenido basado en la imitación de una imagen dada. Asimismo la linea de transferencia de estilo (*Style Transfer*) [43] se dedica a transferir o aplicar el estilo de una imagen en otra, normalmente utilizando redes generativas antagónicas (*Generative Adversarial Networks*) [45]. En ambos casos es necesario caracterizar texturas o relaciones espaciales

entre características. Este proceso normalmente es realizado a través de funciones de perdida específicas o añadiendo al modelo la capacidad de caracterizar las dependencias entre texturas. Viendo la capacidad de caracterizar la dependencia espacial de las características del método presentado de *Co-Ocurrencias*, creemos que puede ser interesante estudiar la aplicabilidad de dicho método a ambos problemas de síntesis de texturas y transferencia de estilo.

Bibliografía

[1] Cifar-10 and cifar-100 datasets. https://www.cs.toronto.edu/ kriz/cifar.html.

[2] Stl-10 dataset. https://cs.stanford.edu/ acoates/stl10/.

[3] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

[4] Petros Fotios Alvanitopoulos, Andrian Moroi, George Bagropoulos, and Kieran Dundon. Content based image retrieval and its application to product recognition. In *IFIP international conference on artificial intelligence applications and innovations*, pages 3–18. Springer, 2015.

[5] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016.

[6] Yannis Avrithis and Yannis Kalantidis. Approximate gaussian mixtures for large scale vocabularies. In *European Conference on Computer Vision*, pages 15–28. Springer, 2012.

[7] Artem Babenko and Victor Lempitsky. Aggregating local deep features for image retrieval. In *Proceedings of the IEEE international conference on computer vision*, pages 1269–1277, 2015.

[8] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. Neural codes for image retrieval. In *European conference on computer vision*, pages 584–599. Springer, 2014.

[9] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[10] Brent Berlin and Paul Kay. *Basic color terms: Their universality and evolution*. Univ of California Press, 1991.

[11] Y-Lan Boureau, Francis Bach, Yann LeCun, and Jean Ponce. Learning mid-level features for recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2559–2566. IEEE, 2010.

[12] Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 111–118, 2010.

[13] Alan C. Bovik, Marianna Clark, and Wilson S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE transactions on pattern analysis and machine intelligence*, 12(1):55–73, 1990.

[14] Titus J Brinker, Achim Hekler, Alexander H Enk, Joachim Klode, Axel Hauschild, Carola Berking, Bastian Schilling, Sebastian Haferkamp, Dirk Schadendorf, Tim Holland-Letz, et al. Deep learning outperformed 136 of 157 dermatologists in a head-to-head dermoscopic melanoma image classification task. *European Journal of Cancer*, 113:47–54, 2019.

[15] H. Bustince, M. Pagola, R. Mesiar, E. Hullermeier, and F. Herrera. Grouping, overlap, and generalized bientropic functions for fuzzy modeling of pairwise comparisons. *IEEE Transactions on Fuzzy Systems*, 20(3):405–415, June 2012.

[16] Geert Caenen and Luc Van Gool. Maximum response filters for texture analysis. In *2004 Conference on Computer Vision and Pattern Recognition Workshop*, pages 58–58. IEEE, 2004.

[17] Bingyi Cao, André Araujo, and Jack Sim. Unifying deep local and global features for image search. In *European Conference on Computer Vision*, pages 726–743. Springer, 2020.

[18] Jesus Chamorro-Martínez, Jose Manuel Soto-Hidalgo, Pedro Manuel Martínez-Jiménez, and Daniel Sánchez. Fuzzy color spaces: A conceptual approach to color vision. *IEEE Transactions on Fuzzy Systems*, 25(5):1264–1280, 2016.

[19] Olivier Chapelle, Patrick Haffner, and Vladimir N Vapnik. Support vector machines for histogram-based image classification. *IEEE transactions on Neural Networks*, 10(5):1055–1064, 1999.

[20] Ondřej Chum, Andrej Mikulik, Michal Perdoch, and Jiří Matas. Total recall ii: Query expansion revisited. In *CVPR 2011*, pages 889–896. IEEE, 2011.

[21] Ondrej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[22] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

[23] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[24] George R Cross and Anil K Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):25–39, 1983.

[25] G Csurka. Dance, cr fan, l. willamowski, j. and bray c.,"visual categorization with bags of keypoints,". In *Proc. European Conference on Computer Vision (ECCV)*, 2004.

[26] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[27] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[28] Eric Dodds, Huy Nguyen, Simao Herdade, Jack Culpepper, Andrew Kae, and Pierre Garrigues. Learning embeddings for product visual search with triplet loss and online sampling. *arXiv preprint arXiv:1810.04652*, 2018.

[29] Piotr Dollár, Zhuowen Tu, Pietro Perona, and Serge Belongie. Integral channel features. 2009.

[30] Geoff Dougherty. *Pattern recognition and classification: an introduction*. Springer Science & Business Media, 2012.

[31] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1033–1038. IEEE, 1999.

[32] Xiaodong Feng, Zhi Xiao, Bo Zhong, Jing Qiu, and Yuanxiang Dong. Dynamic ensemble classification for credit scoring using soft probability. *Applied Soft Computing*, 65:139–151, 2018.

[33] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[34] Itzhak Fogel and Dov Sagi. Gabor filters as texture discriminator. *Biological cybernetics*, 61(2):103–113, 1989.

[35] William T Freeman, Edward H Adelson, et al. The design and use of steerable filters. *IEEE Transactions on Pattern analysis and machine intelligence*, 13(9):891–906, 1991.

[36] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[37] Johannes Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2(Mar):721–747, 2002.

[38] Johannes Fürnkranz. Round robin ensembles. *Intelligent Data Analysis*, 7(5):385–403, 2003.

[39] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44(8):1761–1776, 2011.

[40] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44(8):1761–1776, 2011.

[41] Mikel Galar, Alberto Fernández, Edurne Barrenechea, and Francisco Herrera. Drcw-ovo: distance-based relative competence weighting combination for one-vs-one strategy in multi-class problems. *Pattern recognition*, 48(1):28–42, 2015.

[42] Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 317–326, 2016.

[43] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[44] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[45] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[46] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *International conference on machine learning*, pages 1319–1327. PMLR, 2013.

[47] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *European conference on computer vision*, pages 241–257. Springer, 2016.

[48] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2):237–254, 2017.

[49] Benjamin Graham. Fractional max-pooling. *arXiv preprint arXiv:1412.6071*, 2014.

[50] Caglar Gulcehre, Kyunghyun Cho, Razvan Pascanu, and Yoshua Bengio. Learned-norm pooling for deep feedforward and recurrent neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 530–546. Springer, 2014.

[51] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3279–3286, 2015.

[52] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621, 1973.

[53] John A Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[54] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[55] Jian Hou and Marcello Pelillo. A simple feature combination method based on dominant sets. *Pattern Recognition*, 46(11):3129–3139, 2013.

[56] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, 13(2):415–425, 2002.

[57] Zilong Hu, Jinshan Tang, Ziming Wang, Kai Zhang, Ling Zhang, and Qingling Sun. Deep learning for image-based cancer detection and diagnosis- a survey. *Pattern Recognition*, 83:134–149, 2018.

[58] Hisao Ishibuchi, Tomoharu Nakashima, and Manabu Nii. *Classification and modeling with linguistic information granules: Advanced approaches to linguistic Data Mining*. Springer Science & Business Media, 2004.

[59] Anil K Jain and Farshid Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern recognition*, 24(12):1167–1186, 1991.

[60] Jonathan Janke, Mauro Castelli, and Aleš Popovič. Analysis of the proficiency of fully connected neural networks in the process of classifying digital images. benchmark of different classification algorithms on high-level image features from convolutional layers. *Expert Systems with Applications*, 135:12–38, 2019.

[61] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 304–317, Berlin, Heidelberg, 2008. Springer-Verlag.

[62] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *European conference on computer vision*, pages 304–317. Springer, 2008.

[63] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3304–3311. IEEE, 2010.

[64] Herve Jegou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Perez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *IEEE transactions on pattern analysis and machine intelligence*, 34(9):1704–1716, 2011.

[65] Albert Jimenez, Jose M Alvarez, and Xavier Giro-i Nieto. Class-weighted convolutional features for visual instance search. *arXiv preprint arXiv:1707.02581*, 2017.

[66] Yushi Jing, David Liu, Dmitry Kislyuk, Andrew Zhai, Jiajing Xu, Jeff Donahue, and Sarah Tavel. Visual search at pinterest. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1889–1898, 2015.

[67] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *European conference on computer vision*, pages 685–701. Springer, 2016.

[68] Peyman Hosseinzadeh Kassani and Andrew Beng Jin Teoh. A new sparse model for traffic sign classification using soft histogram of oriented gradients. *Applied Soft Computing*, 52:231 – 246, 2017.

[69] Kenneth Low Kelly. *The ISCC-NBS method of designating colors and a dictionary of color names*, volume 553. US Government Printing Office, 1955.

[70] Salman Khan, Hossein Rahmani, Syed Afaq Ali Shah, and Mohammed Bennamoun. A guide to convolutional neural networks for computer vision. *Synthesis Lectures on Computer Vision*, 8(1):1–207, 2018.

[71] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. *Logistic regression*. Springer, 2002.

[72] Alexander Kolesnikov and Christoph H Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *European conference on computer vision*, pages 695–711. Springer, 2016.

[73] Piotr Koniusz, Fei Yan, and Krystian Mikolajczyk. Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection. *Computer Vision and Image Understanding*, 117(5):479–492, may 2013.

[74] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.

[75] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[76] Zakaria Laskar and Juho Kannala. Context aware query image representation for particular object retrieval. In *Scandinavian Conference on Image Analysis*, pages 88–99. Springer, 2017.

[77] Kenneth I Laws. Rapid texture identification. In *Image processing for missile guidance*, volume 238, pages 376–381. International Society for Optics and Photonics, 1980.

[78] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178. IEEE, 2006.

[79] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[80] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[81] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[82] Stan Z. Li and Anil Jain. Encyclopedia of biometrics. *Springer US*, pages 883–883, 2009.

[83] Qiong Liu and Ying Wu. *Supervised Learning*, pages 3243–3245. Springer US, Boston, MA, 2012.

[84] Ana Carolina Lorena, André CPLF De Carvalho, and João MP Gama. A review on the combination of binary classifiers in multiclass problems. *Artificial Intelligence Review*, 30(1-4):19, 2008.

[85] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[86] Jitendra Malik and Pietro Perona. Preattentive texture discrimination with early vision mechanisms. *JOSA A*, 7(5):923–932, 1990.

[87] Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 1989.

[88] Bangalore S Manjunath and Wei-Ying Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on pattern analysis and machine intelligence*, 18(8):837–842, 1996.

[89] Jianchang Mao and Anil K Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern recognition*, 25(2):173–188, 1992.

[90] Eva Mohedano, Kevin McGuinness, Xavier Giro-i Nieto, and Noel E. O'Connor. Saliency Weighted Convolutional Features for Instance Search. nov 2017.

[91] Loris Nanni, Stefano Ghidoni, and Sheryl Brahnam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158–172, 2017.

[92] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[93] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017.

[94] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *Science and Information Conference*, pages 128–144. Springer, 2019.

[95] Florent Perronnin, Yan Liu, Jorge Sánchez, and Hervé Poirier. Large-scale image retrieval with compressed fisher vectors. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3384–3391. IEEE, 2010.

[96] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[97] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[98] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[99] Pedro O Pinheiro and Ronan Collobert. From image-level to pixel-level labeling with convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1713–1721, 2015.

[100] Gowdham Prabhakar, Binsu Kailath, Sudha Natarajan, and Rajesh Kumar. Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving. In *2017 IEEE Region 10 Symposium (TENSYMP)*, pages 1–6. IEEE, 2017.

[101] Danfeng Qin, Stephan Gammeter, Lukas Bossard, Till Quack, and Luc Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR 2011*, pages 777–784. IEEE, 2011.

[102] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[103] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5706–5715, 2018.

[104] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In *European conference on computer vision*, pages 3–20. Springer, 2016.

[105] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 3–20, Cham, 2016. Springer International Publishing.

[106] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668, 2018.

[107] Ali S Razavian, Josephine Sullivan, Stefan Carlsson, and Atsuto Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4(3):251–258, 2016.

[108] Yong Rui, Thomas S Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of visual communication and image representation*, 10(1):39–62, 1999.

[109] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[110] Xiaohui Shen, Zhe Lin, Jonathan Brandt, and Ying Wu. Spatially-constrained similarity measurefor large-scale object retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 36(6):1229–1241, 2013.

[111] Zenglin Shi, Yangdong Ye, and Yunpeng Wu. Rank-based pooling for deep convolutional neural networks. *Neural Networks*, 83:21–31, 2016.

[112] Ya Fang Shih, Yang Ming Yeh, Yen Yu Lin, Ming Fang Weng, Yi Chang Lu, and Yung Yu Chuang. Deep co-occurrence feature learning for visual object recognition. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-January:7302–7311, 2017.

[113] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[114] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *null*, page 1470. IEEE, 2003.

[115] Michael J Swain and Dana H Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.

[116] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[117] Kezhu Tan, Won Suk Lee, Hao Gan, and Shuwen Wang. Recognising blueberry fruit of different maturity using histogram oriented gradients and colour features in outdoor scenes. *Biosystems Engineering*, 176:59 – 72, 2018.

[118] Joshua B Tenenbaum and William T Freeman. Separating style and content with bilinear models. *Neural computation*, 12(6):1247–1283, 2000.

[119] Fok Hing Chi Tivive and Abdesselam Bouzerdoum. Texture classification using convolutional neural networks. In *TENCON 2006-2006 IEEE Region 10 Conference*, pages 1–4. IEEE, 2006.

[120] Giorgos Tolias, Teddy Furon, and Hervé Jégou. Orientation covariant aggregation of local descriptors with embeddings. In *European Conference on Computer Vision*, pages 382–397. Springer, 2014.

[121] Giorgos Tolias and Hervé Jégou. Visual query expansion with or without geometry: refining local descriptors by feature aggregation. *Pattern recognition*, 47(10):3466–3476, 2014.

[122] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. Particular object retrieval with integral max-pooling of cnn activations. *arXiv preprint arXiv:1511.05879*, 2015.

[123] Tiberio Uricchio, Marco Bertini, Lorenzo Seidenari, and Alberto Bimbo. Fisher encoded convolutional bag-of-windows for efficient image retrieval and social image tagging. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 9–15, 2015.

[124] Vladimir Vapnik and Vlamimir Vapnik. Statistical learning theory wiley. *New York*, 1:624, 1998.

[125] Nuno Vasconcelos and Andrew Lippman. A probabilistic architecture for content-based image retrieval. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 1, pages 216–221. IEEE, 2000.

[126] Yannick Verdie, Kwang Yi, Pascal Fua, and Vincent Lepetit. Tilde: A temporally invariant learned detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5279–5288, 2015.

[127] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

[128] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[129] Chong Wang and Kaiqi Huang. How to use Bag-of-Words model better for image classification. *Image and Vision Computing*, 38:65–74, 2015.

[130] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[131] Tobias Weyand, Andre Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2575–2584, 2020.

[132] Hongtao Xie, Yongdong Zhang, Jianlong Tan, Li Guo, and Jintao Li. Contextual query expansion for image retrieval. *IEEE Transactions on Multimedia*, 16(4):1104–1114, 2014.

[133] Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10*, page 270, 2010.

[134] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *European Conference on Computer Vision*, pages 467–483. Springer, 2016.

[135] Kwang Moo Yi, Yannick Verdie, Pascal Fua, and Vincent Lepetit. Learning to assign orientations to feature points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 107–116, 2016.

[136] Joe Yue-Hei Ng, Fan Yang, and Larry S Davis. Exploiting local features from deep networks for image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 53–61, 2015.

[137] Lotfi A Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

[138] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4353–4361, 2015.

[139] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[140] Xi Zhou, Kai Yu, Tong Zhang, and Thomas S Huang. Image classification using super-vector coding of local image descriptors. In *European conference on computer vision*, pages 141–154. Springer, 2010.

Capítulo II

Publicaciones, trabajos publicados

1. Combinación de características a través de ensembles ponderados para clasificación de imágenes

J.I. Forcén, M. Pagola, E. Barrenechea, H. Bustince

Combination of features through weighted ensembles for image classification

Estado: ***Publicado***

Revista: ***Applied Soft Computing Journal***

DOI: `https://doi.org/10.1016/j.asoc.2019.105698`

Índice de impacto: ***5.472 - Q1 JCR.***

Áreas de conocimiento: ***Computer Science, Artificial Intelligence***

Applied Soft Computing Journal 84 (2019) 105698

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Combination of features through weighted ensembles for image classification

J.I. Forcén [a,b,*], M. Pagola [b], E. Barrenechea [b], H. Bustince [b,c]

[a] das-Nano - Veridas, Poligono Industrial Talluntxe II, Tajonar, Spain
[b] Dpt. Estadística, Informática y Matemáticas and with the Institute of Smart Cities Universidad Pública de Navarra, 31006, Pamplona, Spain
[c] King Abdulaziz University, Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 27 November 2018
Received in revised form 29 March 2019
Accepted 7 August 2019
Available online 19 August 2019

Keywords:
Image classification
Feature combination
Multi-class ensemble
Score vector

ABSTRACT

Image classification is a multi-class problem that is usually tackled with ensembles of binary classifiers. Furthermore, one of the most important challenges in this field is to find a set of highly discriminative image features for reaching a good performance in image classification. In this work we propose to use weighted ensembles as a method for feature combination. First, a set of binary classifiers are trained with a set of features and then, the scores are weighted with distances obtained from another set of feature vectors. We present two different approaches to weight the score vector: (1) directly multiplying each score by the weights and (2) fusing the scores values and the distances through a Neural Network. The experiments have shown that the proposed methodology improves classification accuracy of simple ensembles and even more it obtains similar classification accuracy than state-of-the-art methods, but using much less parameters.

1. Introduction

Image recognition is the ability of a program to identify objects, places, people, or actions in images. It is usually presented as a multi-class classification problem in which the identification of a relevant set of specific features which performs best is a difficult task [1]. It exists many different features that can be extracted from an image and they have been used in image classification systems with good performance. Typical local low-level features are the Histogram of Oriented Gradients (HOG) [2], color histograms [3], the Scale-Invariant Feature Transform (SIFT) [4] and patch-based processing of images [5]. Even more, mid-level features can be defined using the well known Bag-of-Features (BoF) method [6]; those mid-level features derived are very discriminative and report great performance in image classification [7]. Although classification problems are usually multi-class ones, binary classifiers are much more studied in the literature. Therefore, usually, multi-class problems are divided into easier-to-solve binary classification problems. As a result, a set of classifiers is learned, each one being responsible for a binary problem. Common decomposition strategies are the well-known, One-vs-All (OVA) and One-vs-One (OVO) [8–10]. After each binary classifier is trained, the main problem is how to combine their different results. Different strategies can be employed to combine the results, for example based on their classification results [11] or by means of a cost-sensitive combination [12].

Basically, the performance of an image classification system depends on the choice of the feature vectors, the classifier learning paradigm and the decomposition strategy. Leaving apart the classifier, in this work we are going to deal with the problems of selecting a set of feature vectors and the performance of the decomposition strategy.

First of all, the problem of choosing a correct set of features can be tackled using different types of feature vectors and combining them. Feature combination can be categorized into two types [13], the first one is to join different feature vectors into one vector and then fed the classifier with these larger vectors (for example [14]); obviously with this methodology the learning step is harder. The second type is to affect directly to the score of the classifier allowing the combination of different type of classifiers. This approach is really interesting as different types of classifiers and feature vectors can be combined together. However, in these types of ensembles a cross validation step is needed to find the best combination of features [15] and classifiers [13]. Therefore, all of the feature combination methodologies add an extra computational complexity.

Secondly, in OVA setting of a multi-class problem imbalanced training data-sets are usually produced when instances from the single class are compared with all other instances in the data-set. It is well-known in Machine Learning that imbalanced data-sets can cause some undesirable effects in the derived classifiers. All classifiers are biased through the majority class, so the classifier

* Corresponding author at: das-Nano - Veridas, Poligono Industrial Talluntxe II, Tajonar, Spain.
E-mail address: forcen.91678@e.unavarra.es (J.I. Forcén).

https://doi.org/10.1016/j.asoc.2019.105698
1568-4946/

output of the real class for a new instance could be underestimated. Also, OVO ensembles suffer a relevant problem of non-competent classifiers. For example, if an OVO ensemble tries to classify an instance of class z, the base classifier that distinguishes between class i and class j is said to be non-competent since it has not been trained with examples of class z.

The objective of this work is to propose a new methodology in which the problems of decomposition strategies are soften and features combination is possible with little extra computational complexity. In our method, first a set of features, a classifier and a decomposition strategy (OVO or OVA) are selected. Then, we weight the score vector or the score matrix of the ensemble with the information provided by other feature vectors.

In [16] it was proposed a method to avoid the non-competent classifiers in OVO weighting the score matrix with weights obtained from a relative distance. The distances where calculated with the same feature vectors used to train the classifiers. In this work, we extend this methodology as a method for feature combination. In addition, we propose to use this methodology in the OVA ensembles and we present a trainable method to obtain the best way for combining the distances and the scores of the ensembles.

The contributions of this paper can be summarized as follows:

- We propose a methodology for feature combination through weighting ensembles. We propose some expressions to calculate the weights for image classification in both OVA and OVO settings.
- Based on the weighed ensembles we propose a trainable Neural Network to find the best combination between the distances and the scores.
- In the experimental results we demonstrate the effectiveness of our method, that improves other feature combination methods or distance based classifiers. Basically we show that it is a good methodology to combine mid level image features like Bag-of-Features vectors, and low level features vectors, as Histogram of Oriented Gradients.

This paper is divided into the following sections: first, in Section 2, the basic concepts of ensemble classification and feature vectors for image classification are recalled. In Section 3 an exhaustive description of our proposal is given and a detailed explanation of how to calculate distances between images is detailed Section 4. Experimental results are shown in Section 5. Finally, the most important conclusions and future works are drawn.

2. Notation and related work

In this section we are going to introduce concepts that are used through the paper. Firstly, we are going to recall the OVA and OVO ensembles methods in multi-class problems and the description of some features for image classification which are used in the experimentation.

2.1. Multi-class ensembles

Decomposition strategies allow to deal with multi-class problems considering binary classifiers (in [10] a deep and complete revision is presented). Next, is described two extended approaches, One-vs-All and One-vs-One decompositions, both used in this article.

2.1.1. One-versus-all ensemble for multi-class problems

One-vs-All (OVA) decomposition divides multi-class classification problem with m classes, into m binary classification problems. Each problem is faced by a binary classifier, usually referred to as base classifier, which is the responsible for distinguishing one class C_i from the rest of the classes (all of them considered as a single class). The learning step of the classifiers is done using the whole training data, considering the instances from C_i as positives and all other examples as negatives. In the validation phase, an instance is presented to each one of the binary classifiers. The most common approach uses the confidence of the classifiers to decide the final output, predicting the class from the classifier with the largest confidence; that is, from a score vector $R = (r_1, r_2, \ldots, r_m)$, where $r_i \in [0, 1]$ is the confidence value for class i, then the output class is given by:

$$\text{Class} = \arg \max_{i=1,\ldots,m} r_i. \tag{1}$$

2.1.2. One-versus-one ensemble for multi-class problems

One-vs-One (OVO) ensemble divides an m class problem into $m(m-1)/2$ binary problems. Each problem is faced by a binary classifier which is responsible for distinguishing from one class C_i from another class C_j. A new instance is classified considering all the base classifiers where the outputs can be stored in a voting matrix:

$$R = \begin{pmatrix} - & r_{12} & \cdots & r_{1m} \\ r_{21} & - & \cdots & r_{2m} \\ \vdots & & & \vdots \\ r_{m1} & r_{m2} & \cdots & - \end{pmatrix}, \tag{2}$$

being $r_{ij} \in [0, 1]$ the confidence value in favor of C_i given by the classifier that distinguishes between $\{C_i, C_j\}$; on the contrary, the confidence for C_j is calculated by $r_{ji} = 1 - r_{ij}$ (just in case this value is not provided by the classifier). Therefore, the voting matrix is symmetric.

To establish the class to which a new instance belongs, different aggregation methods can be used to the voting matrix [10, 17]. The simplest one is that each classifier gives a vote to the predicted class; the class (i.e. the file of the score matrix) with more votes is the predicted one.

2.2. Features for image classification

Below we describe briefly the image features that are used in the experimentation. Specific details such as the values of the parameters are given in the experimental results section.

2.2.1. Histogram of oriented gradients (HOG)

The main idea of *HOG* [2] is that an object and its shape can be characterized through the distribution of their local gradients. To obtain the *HOG* of an image, the process to be performed is as follows. Firstly, the image is divided into blocks and, in order to avoid problems caused by different light conditions, a contrast normalization process is carried out in each block. Later on, considering that each block contains cells, a histogram of 9 gradient directions is created by adding the module of the gradient of the pixels of each cell in each direction. Finally, all the histograms are concatenated to obtain the HOG features vector of the image. These vectors are usually used to train a classifier. HOG feature vectors have been widely used in a large range of applications [18,19].

2.2.2. Fuzzy Color Histograms (FCH)

A histogram is a function that provides information about the pixels frequency with a range of color. Therefore, it depends on the color representation of the image. In this work, two different color representations are used, RGB and Fuzzy Color. Usually images are stored in RGB codification, so work in this representation is really straightforward, however colors are imprecise, so in order to tackle that, fuzzy color representation is also tested. Fuzzy set theory was introduced by Zadeh [20] and allows a gradual degree of membership of the elements to a set. This way each set is described with a membership function valued in the real unit interval [0, 1]. A fuzzy color is the computational representation of a color term defined and named by humans. Hence, a fuzzy color is a linguistic label whose semantics is represented as a normal fuzzy subset of colors [21]. That means that each pixel has a membership degree to every fuzzy color, which set represented by means of a membership function. Like RGB color histograms, fuzzy color histograms give us the number of pixels of a fuzzy color in an image [22].

2.2.3. Bag of Features (BoF)

Bag-of-Features [6] is a popular approach to find descriptive image features, which transforms previously computed image descriptors into an image representation that is used in matching and classification. Basically, the BoF model for image classification can be summarized in the following steps:

1. Extraction of descriptors. Local image descriptors, such as HOG or processed patches, are extracted from images at interest points in a dense grid.
2. Dictionary creation or discovery. From the previous descriptors, an unsupervised learning algorithm is used to discover a set of prototype descriptors; this set is called a *dictionary*. This operation is usually done with k-means. Each of the descriptors found in the dictionary is called a visual feature.
3. Feature coding. This step can be understood as an activation function for the dictionary, in such a way that each of the visual features is activated according to a local descriptor. The local descriptor is transformed in a new feature vector, that has the size of the dictionary, in which every value corresponds to the similarity of the local descriptor with the visual features of the dictionary.
4. Pooling. The vectors associated with visual image features are combined over some image neighborhood.
5. Training and classification. It can be performed on the final feature vectors (usually the concatenation of the signature feature vectors of different cells) by a classifier, e.g. support vector machine (SVM) [3,23].

3. Weighted ensembles

Next, to stress the motivation and the goal of this paper we present an example. Let us consider an image data-set which contains 10 classes of images: *Plane, Bird, Car, Cat, Deer, Dog, Horse, Monkey, Ship* and *Truck*. First, we train an OVA classifier with SVM as base classifier and given a new instance (Fig. 1) to be classified, the score vector R of confidence values obtained is shown in Table 1. In this case the base classifiers assign negative confidences to all the classes and, as we can observe, the method fails due to the maximum value corresponds to the *Cat* class.

To deepen into these results, we have calculated the Euclidean distances between a color feature vector of Fig. 1 to the nearest image of each class, both are shown in Fig. 2. In this example, the color feature vector of an image is the concatenation of R, G and B histograms with 9 bins in each channel, i.e. 27 features.

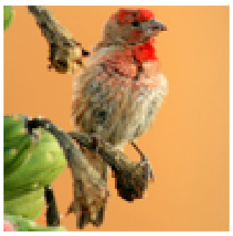


Fig. 1. Bird, example test image from STL10 dataset.

Table 1
Score vector R.

Plane	Bird	Car	Cat	Deer
−0.5007	−0.306	−2.803	−0.122	−0.645
Dog	Horse	Monkey	Ship	Truck
−1.314	−0.9458	−1.0924	−1.492	−2.013

As we can observe, the value of the Euclidean distance to *Bird* class is the second lowest value. Therefore, if we weight the score vector R with the inverse of these values, we can increase the score value of the *Bird* class and then make a correct prediction.

Therefore, we propose the following methodology to combine different feature vectors and increase the accuracy in an image classification problem:

- First, train an ensemble (OVA or OVO) with the images represented by a feature vector.
- Next, in the evaluation phase of a test image, calculate a weight for every class by means of the distance of such image to each class with a different type of feature vector to represent the images.
- Combine the weights and the values of the score values to obtain a better classification of the test image.

Next, we describe how to calculate a distance of an image to a class (Section 3.1), we present the necessary expressions to calculate the weights in an OVA and an OVA ensemble (Section 3.2 and 3.4), and finally, we describe a Neural Network based methodology (Section 3.6) to train the aggregation of scores and distances. We conclude with a discussion about score normalization (Section 3.7).

3.1. Distance to a class

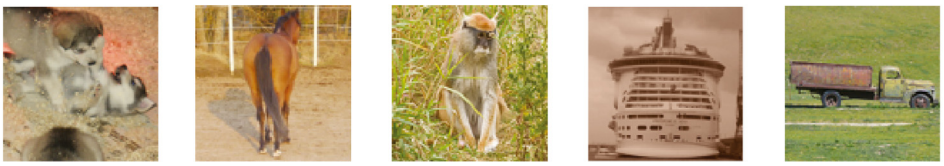
In [24] is defined the distance from a test instance to a C_i class, notated as d_i, as the average of the k distances of the $k-$nearest neighbors in this C_i class. Note that in this work the distances will be distance between the test image and the images in the training set of class C_i.

3.2. Weights in OVA

In order to compute the weights in OVA ensemble Eq. (3) is used. The weight, w_i, of the base classifier of C_i is calculated as the relation between the average of the distances between the test image and the other classes and the distance to the current class, i.e., d_i. Therefore, the nearer a C_i class is to the image, then the larger the weight will be. So classifiers which contain the most similar images to the test image (compared to the rest of the classes) are boosted.

$$w_i = \left(\frac{1}{m} \sum_{\substack{j=1\ldots m \\ j \neq i}} d_j \right) / d_i \tag{3}$$

(a) 0.9282 (b) 0.8546 (c) 1.1754 (d) 0.9241 (e) 0.8422



(f) 0.9189 (g) 0.8830 (h) 0.8777 (i) 1.0345 (j) 1.0447

Fig. 2. Nearest images per class and their Euclidean distances.

being d_i and d_j the distance to class i and j respectively and m the number of classes.

Remark. Notice that d_i cannot be equal to zero due to the images in the test set are different from the images in the training set.

3.3. Weighted OVA

We combine the results obtained in the OVA classifier, $R = (r_1, r_2, \ldots, r_m)$, and the weights, $W = (w_1, \ldots, w_m)$, calculated with Eq. (3), to obtain the following vector:

$$R^w = (r_1 w_1, r_2 w_2, \ldots, r_m w_m). \tag{4}$$

From Eq. (4) we can estimate the class of the test samples by means of Eq. (5):

$$\text{Class} = \arg \max_{i=1,\ldots,m} r_i w_i. \tag{5}$$

3.4. Weights in OVO

On the other hand, considering an OVO ensemble, the weight w_{ij} of the base classifier that distinguishes between C_i (positive class) and C_j (negative class) is obtained by Eq. (6). So, if the test image is near to the $i-$th class and far from the $j-$th we suppose that the image could be of C_i class, and therefore the base classifier should have a large weight. But if the distance of the image to C_i and C_j is similar, we suppose that the image is neither C_i nor C_j, and therefore the weight is low (in [16] there are other expressions to calculate the weights).

$$w_{ij} = \frac{d_j}{d_i} \tag{6}$$

being d_i and d_j the distance to class i and j respectively.

3.5. Weighted OVO

Similar to the strategy carried out in OVA, we combine the results obtained in the OVO classifier in R, Eq. (2), and OVO weighted results W, Eq. (6), to obtain the following matrix:

$$R^w = \begin{pmatrix} - & r_{12}w_{12} & \cdots & r_{1m}w_{1m} \\ r_{21}w_{21} & - & \cdots & r_{2m}w_{2m} \\ \vdots & & & \vdots \\ r_{m1}w_{m1} & r_{m2}w_{m2} & \cdots & - \end{pmatrix} \tag{7}$$

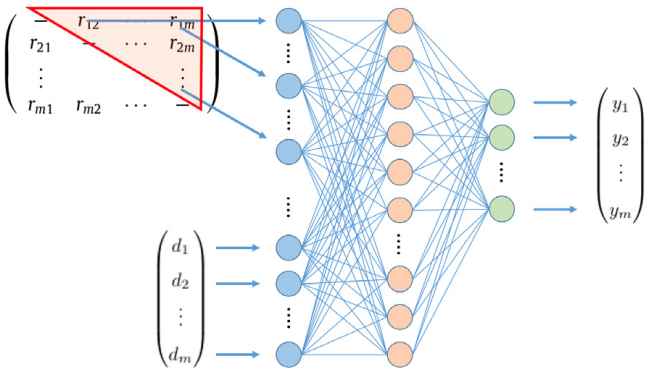


Fig. 3. Neural Network based weighted ensemble.

From Eq. (7) we can estimate the class of the test samples by means of Eq. (8):

$$\text{Class} = \arg \max_{i=1,\ldots,m} \sum_{1 \le j \ne i \le m} r_{ij} w_{ij} \tag{8}$$

3.6. Neural network based weighted ensemble

In this approach, an artificial neural network is trained to obtain the best combination of scores and distances in order to obtain a better classification accuracy. Therefore, using the same training examples used to train the base classifiers, a simple multilayer perceptron is trained. For each training example the input vector is the score vector (or the score matrix in OVO) obtained from the base classifiers concatenated with all of the distances from such example to the classes. In Fig. 3 is depicted the scheme of this method for an OVO ensemble (the values under the diagonal are not used since they are just the negative of the scores obtained by the base classifiers). The output vector is a binary vector encoding the class of the example. In this second training phase, the neural network combines the distances and the original scores, therefore we named it as neural network based weighted ensemble. The objective of this methodology is that the training of the network should find the fittest combination of distances and scores. Although we add a new training module, it adds few computational complexity because the size of the network is small. The size of the input depends on the number of classes and in the experimental results we will show that adding one hidden layer is enough.

3.7. Positive scores

In our proposal the weights are multiplied by the scores of the ensemble. For example, SVM is a binary classifier that can be used as base classifier where the output of each one is a real number (like in the *Bird* example). The score values are negative (if the example belongs to the negative class) and positive (if the example belongs to the positive class). Only classes with positive values will have the possibility to have the maximum value after multiplying the weights. Therefore, we apply a linear normalization to the values of the score vector of the classifier into the range [0, 1], to ensure that all the scores are positive and can be influenced by the weights; in this way, a classifier which output is negative, after the weighting could be the winning class. In the case of using OVO ensembles the resultant matrix in each prediction is normalized by rows.

4. Distances between images

The calculation of distances between images is very important in our proposal since they are the base to calculate, what we call distance of an image to a class. There are a lot of methods to compare images [25,26] and evaluate their similarity. In this work we consider the distance between two images as the Euclidean distance between the feature vectors that describes the images.[1]

However images have an spatial information which could be useful in the image classification process. We have used two different methodologies to take into account the spatial information when we calculate the distance between images, the spatial weighted distance and the co-occurrence matrix.

4.1. Spatial weighted distance

To have spatial information of the image, the image can be divided in blocks, and each block can be represented by a feature vector. For example, an image can be divided into $n \times m$ blocks. A spatial weighted distance between two images is calculated as the weighted sum of the distances between the blocks of both images. Usually a higher importance (weight) is given to the block at the center of the images. In this work we will use a Gaussian distribution function to obtain the weights of the blocks, with the mean located at the center block and the covariance matrix is a diagonal matrix with $n/2$ and $m/2$ in the diagonal.

In Table 2 (a) is depicted an original image divided in 5×5 blocks and in (b) their corresponding weights following a Gaussian distribution with mean = (3,3) and covariance = (2.5, 0, 0, 2.5).

4.2. Matrix of co-occurrence

Co-occurrence representation [27] characterizes the spatial dependency of the visual features in images by means of the visual feature co-occurrence matrix (VFCM). The VFCM is a count of the number of times two visual features satisfy a spatial condition. Given an image I containing a set of visual features $c_i \in C$ at pixel locations (x_i, y_i), and given a distance r, the spatial co-occurrence is defined to be true if two features appear within a distance of r pixels to each other:

$$\rho(c_i, c_j) = \begin{cases} 1, & \text{if} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq r \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

The VFCM computed means the number of times the pairs of features are near to each other. So, that count is represented as a matrix with dimension $M \times M$ for a set of M different features. A feature c_i appears, i.e. is taken into account in the calculation of $\rho(c_i, c_j)$, if its value is larger than a threshold t at a position (x_i, y_i). Being c_p and c_q two different features which appears N_p and N_q times in the image respectively:

$$VFCM(c_p, c_q) = \sum_{p=1}^{N_p} \sum_{q=1}^{N_q} \rho(c_p, c_q) \quad (10)$$

Once these co-occurrence matrices are obtained, the distance between two images is the Euclidean distance between the VCFM matrices of the images.

5. Experimental results

To evaluate our proposal we carried out two different experiments in two images classification problems. We use the cifar-10 data-set [28] (10 categories with 5000 images for training and 1000 for test per class of 32×32) and the STL-10 data-set [29] (10 categories with 500 images for training and 800 for test per class of 96×96).

In both experiments the base classifier used is a linear kernel Support Vector Machine. Specifically L2-SVM, that uses the square sum of the slack variables (the regularization term) in the objective function instead of the linear sum, which makes the SVM less susceptible to outliers and improve its overall generalization. The value of its regularization parameter C is selected by cross-validation within the training set.

5.1. First experiment

In this experiment we evaluate the performance of our proposal, comparing results obtained by OVA and OVO classifiers against their corresponding weighted versions where weights are obtained from the same BoF feature vector.

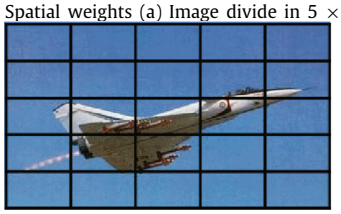
5.1.1. Steps to obtain the BoF vector from an image

1. Select randomly 400.000 random patches (or windows) of 6×6 from the training dataset.
2. Normalize and whiten[2] [29] the set of patches.
3. Use K-Means to obtain a set of M representative patches.
4. Extract all the patches centered at every pixel of the image.
5. Use the triangle assignment method [29] to obtain an activation feature vector of size M for every patch.
6. Divide the image into 4 non overlapped blocks. Combine all the feature vectors of a block by the sum of all of the feature vectors.
7. Concatenate the 4 feature vectors into one feature vector to represent the image.

[1] Feature scaling is used to calculate the distance between two vectors by the Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this particular feature. Therefore, the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance. In this work we do not use feature normalization due to all of the methods used to calculate feature vectors, provide vectors in which their variables are in the same range (f.e. BoF variables are in [0, 1]).

[2] Normalization is done by subtracting the mean and dividing by the standard deviation of its elements. For visual data, this corresponds to local brightness and contrast normalization. In the whitening process, also known as Zero Components Analysis, the first step is to calculate the covariance matrix of a set of images, next compute the eigenvectors of the covariance matrix and project the images onto the eigenvectors and divide each component by the square root of its eigenvalue. This projection makes that there is no correlation between the components of each image, then, after a whitening process, we obtain a less redundant representation of the input images.

Table 2
Spatial weights (a) Image divide in 5 × 5 blocks and (b) weights for each block.



(a)

0.0030	0.0133	0.0219	0.0133	0.0030
0.0133	0.0596	0.0983	0.0596	0.0133
0.0219	0.0983	0.1621	0.0983	0.0219
0.0133	0.0596	0.0983	0.0596	0.0133
0.0030	0.0133	0.0219	0.0133	0.0030

(b)

Table 3
First experiment results.

	Cifar-10				STL-10			
Number of Features	128	256	512	1024	128	256	512	1024
OVA	66.32	70.31	73.20	75.53	62.03	64.85	65.88	67.31
DRW-OVA	68.25	72.40	74.92	76.89	63.43	66.15	66.73	**67.81**
SDCRW-OVA	68.24	72.42	74.92	76.83	63.53	66.18	66.73	67.15
COOC-DRW-OVA	68.33	72.44	74.91	75.17	**63.85**	**66.48**	66.45	66.87
NNW-OVA	**70.85**	**73.92**	**76.12**	**77.1**	63.26	65.75	**66.8**	67.25
OVO	65.47	68.61	70.83	72.13	58.03	60.28	61.85	62.49
DRW-OVO	68.87	71.66	73.77	74.87	59.71	60.81	62.50	62.90
SDCRW-OVO	68.97	71.64	73.54	74.03	59.67	60.80	62.67	62.56
COOC-DRW-OVO	68.03	71.19	73.14	73.54	59.31	60.66	61.8	62.32
NNW-OVO	**73.00**	**74.92**	**75.69**	**74.95**	**62.4**	**64.45**	**64.67**	**64.8**
3-NN	53.80	56.27	58.54	57.18	43.21	43.9	44.45	43.48

After executing these steps, an image is represented by a feature vector of size $4 \times M$. The number of representative patches tested in the experiments are $M = 128, 256, 512$ and 1024.

We have considered the following cases to calculate the weighted ensembles:

1. Distance Relative Weighting OVA/OVO (DRW-OVA/DRW-OVO). The distance between two images is obtained from the Euclidean distance of their corresponding BoF vectors. The weights are calculated by Eq. (3).
2. Spatial Distance Relative Weighting OVA/OVO (SDCRW-OVA/SDCRW-OVO). The distance between images is spatially weighted as explained in Section 3.1. The image is divided 8×8 blocks and the weight of each block is given by a Gaussian distribution with mean = (4,4) and covariance = (4, 0, 0, 4).
3. Co-Occurrence Distance Relative Weighting OVA/OVO (COOC-DRW-OVA/COOC-DRW-OVO). The distance of two images is the Euclidean distance between their respective matrices of co-occurrence obtained from BoF features (Spatial condition is set with radius $r = 3$ and threshold = 1). The weights are calculated by Eq. (3).
4. Neural Network based weighted OVA/OVO (NNW-OVA/NNW-OVO). A simple neural network is trained with one hidden layer of 50 neurons (ReLu activation function) and a 10 softmax output layer. The inputs are the scores of the ensemble (OVA or OVO) concatenated with the classes distances obtained from the Euclidean distance between the BoF vectors of the images.

In all cases the distance of an image to a class is calculated with $k-$nearest neighbor being $k = 3$, and we recall that we apply a normalization to the SVM scores. In Table 3 we show the accuracy in the test sets of cifar-10 and STL-10.

Analyzing the results obtained, weighting the scores with the relative distance improves the accuracy of simple ensembles OVO and OVA and therefore the drawbacks of the ensembles are reduced. Specifically with OVA, all base classifiers are biased through the majority class and with OVO we have the problem of non-competent classifiers (it was also shown in [16] for OVO ensembles). Even more, the accuracy of classical OVO and OVA ensembles are increased 2%–3% in the Cifar-10 dataset by our weighted methodology. The neural network based weighting is better than the weights calculated by Eqs. (3) and (6), as it was expected. We can also conclude that spatial information added with the co-occurrence matrix or the spatial weighting only improves DRW-OVA result in few cases.

In the last row of Table 3, are showed the k-nearest neighbor results (KNN) with $k = 3$ for the BoF vectors used in the experiments. Although, these results are much lower than results obtained with SVM and BoF, our distance based methodology improves both accuracies.

5.2. Second experiment

In the second experiment we use the weighted ensembles as a feature combination method. More specifically, we use BoF features from the first experiment to train the OVA and OVO classifiers and then we calculate the weights from the distances between images represented by different low-level features such as, HOG or FCH features. Next, these two methods of calculating low-level features are detailed.

5.2.1. Steps to obtain the HOG feature vector from an image

1. Divide the image into 64 non overlapped blocks.
2. Create a histogram of 9 gradient directions for each block [2].
3. Concatenate all the histograms to obtain the HOG vector containing $64 \times 9 = 576$ features to represent the image.

5.2.2. Steps to obtain the Fuzzy Color Histogram (FCH) feature vector from an image

In this experiment we have used the Basic color ($\widetilde{\Gamma}_{ISCC-basic}$) representation [21], which consists of 13 fuzzy colors corresponding to the 10 basic color terms (pink, red, orange, yellow, brown, olive, green, yellow–green, blue, purple), and 3 achromatic ones (white, gray, and black).

1. Divide the image into 16 non overlapped blocks.
2. Compute the fuzzy color histogram for each block by adding the membership degrees of each pixel to every fuzzy color.
3. Concatenate all the histograms to obtain the FCH feature vector containing $16 \times 13 = 208$ features to represent an image.

In this second experiment we have considered the following cases to calculate the weighted ensembles:

1. OVA/OVO with (HOG+BoF). We concatenate BoF vectors to HOG to feed the OVA/OVO classifier, just to compare this simple feature combination technique with our method.
2. Spatial Distance Relative Weighting OVA/OVO (SDCRW-OVA/SDCRW-OVO). The distance between images is calculated with HOG feature vectors and it is spatially weighted

Table 4
Second experiment results.

	Cifar-10				STL-10			
Number of Features	128	256	512	1024	128	256	512	1024
OVA (BoF)	66.32	70.31	73.20	75.53	62.03	64.85	65.88	67.31
OVA (HOG+BoF)	70.48	72.63	75.24	77.14	67.23	68.59	69.3	**69.86**
SDCRW-OVA	69.71	73.48	76.02	78.03	65.4	68.15	68.39	69.51
W-OVA-RGB-16	67.95	72.13	74.39	76.74	63.05	65.31	66.14	67.68
W-OVA-FCH	68.05	72.16	74.75	76.69	63.25	66.06	66.45	67.48
NNW-SDCRW-OVA	**75.32**	**77.08**	**79.1**	**79.96**	**67.31**	**68.61**	**69.55**	69.26
NNW-OVA-RGB-16	68.99	72.37	74.65	76.58	63.975	65.8	66.92	67.5
OVO (BoF)	65.47	68.61	70.83	72.13	58.03	60.28	61.85	62.49
OVO (HOG+BoF)	67.74	69.76	71.72	73.31	64.11	64.68	64.80	65.04
SDCRW-OVO	70.16	72.96	77.30	76.03	63.13	64.79	65.93	66.36
W-OVO-RGB-16	68.49	70.78	72.57	73.73	59.50	61.18	62.10	63.25
W-OVO-FCH	68.12	71.06	72.95	73.92	59.44	60.79	62.21	63.23
NNW-SDCRW-OVO	**76.77**	**78.03**	**78.45**	**77.5**	**67.2**	**68.27**	**68.01**	**67.82**
NNW-OVO-RGB-16	72.21	74.2	75.29	74.55	62.67	64.45	65.03	65.33

as explained in Section 3.1. The image is divided 8×8 blocks. The weight of each block is given by a Gaussian distribution with mean = (4,4) and covariance = (4, 0, 0, 4).

3. W-OVA-RGB-16/W-OVO-RGB-16. The weights are obtained from the Euclidean distance between the RGB vector of the images divided into 16 cells.
4. W-OVA-FCH/W-OVO-FCH. Weights are obtained from the Euclidean distance between the FCH feature vectors of the images.
5. NNW-SDCRW-OVA/NNW-SDCRW-OVO. A Neural Network with one hidden layer (50 –10) based weighted OVA/OVO. The inputs are the scores of the ensemble (OVA or OVO) concatenated with the distances between HOG feature vectors (similar to SDCRW).
6. NNW-OVA-RGB-16/NNW-OVO-RGB-16. A Neural Network with one hidden layer (50 –10) based weighted OVA/OVO. The inputs are the scores of the ensemble (OVA or OVO) concatenated with the Euclidean distance between the RGB vector of the images divided into 16 cells.

In all cases the distance of an image to a class is calculated with $k-$nearest neighbor being $k = 3$, and we recall that we apply a normalization to the SVM scores.

Table 4 shows the results of the second experiment. As we can appreciate in most cases our feature combination is better than concatenating both feature vectors (HOG + BoF). Similar to the first experiment NNW-SDCRW-OVA/NNW-SDCRW-OVO methods obtain the best result. Moreover, we showed that our methodology is useful for feature combination, since the results in this second experiment are around 2%–3% greater than the first experiment. So, adding another feature information and using a simple neural network to combine the values of the score and the relative distances ends up in an increase of the accuracy.

In addition to the numerical results, in Fig. 4 are depicted some test images of STL-10 and its classification result. In each row, the first column is the test image and the second column is the nearest image using the Spatial Distance Relative Weighting with HOG feature. The third and fourth columns are the output of an OVA ensemble and the output of the weighted ensemble NNW-SDCRW-OVA, respectively. In the first two test images, the simple OWA classifies correctly the image (remember that the class selected is the class with maximum confidence value) and the output of our proposal is the same, but the confidence of the classes are different and now the confidence of the correct class is much higher. For the third and fourth test images, the OVA fails to classify correctly, but after weighting the ensemble our method chooses the correct class of the images. Note that in the third image although the nearest image is a dog, as we use the distance to a class as the average of the 3 nearest images, the weighted ensemble increases the score of the class bird and it is finally the class selected. Finally, in the latest two images both OVA and our proposal produce a misclassification. Analyzing these images (and other images wrongly classified) we note that both, the prediction of the OVA and the classes with lower distance to the test image do not include the correct class. So, as a conclusion, the feature vectors used in our proposal must contain complementary information, in such way, that at least one of the representations is able to provide a nearly correct classification.

Table 5
State of the art results Cifar10 and STL-10.

	CIFAR10	STL10
OVA	78.47	67.43
SDCRW-OVA	80.23	70.69
NNW-SDCRW-OVA	80.9	70.41
OVO	74.19	63.37
SDCRW-OVO	77.02	67.43
NNW-SDCRW-OVO	77.81	67.43
Convolutional Kernel Networks [31]	82.18	62.32
Deep non-linear classifier (SVM layers) [32]	79.7	–
Coates et al.[29] (4000 features)	79.6	–
CNN Unsupervised Feature Learning [33]	–	72.8
CNN Unsupervised Pre-training [34]	–	70.2

5.3. State of the art comparison

In this section, considering Cifar-10 and STL-10 data-sets, we compare the results of our method against the most important state of the art methods (see Table 5). Coates et al. [29] use 4000 features with BoF in Cifar-10 data-set reaching the state of the art at this time (2012) using linear models 79.6%. As it was expected from our previous experiments, with the same settings our method obtain an improvement in the results of 80.9%. However, recent improvements in CNNs have shown accuracies greater than 90%, for example in [30] is reached 95.59%. However in STL-10, in which there are less labeled images for the training phase, our approach improves some results obtained with CNNs, even using a linear model with significantly less parameters than the CNNs. For example, comparing with Convolutional Kernel Networks [31] our model is a bit worse in cifar-10, but on STL-10 our model achieves much better performance.

Regarding the time complexity of our proposed method, just notice that the calculation of the distances is linear in the size of the training set $O(m)$. This extra time is suitable due to SVM is generally around $O(m^2)$. Therefore computing the distances is assumable and the total time is less than many feature selection algorithms (searches over a 2^n number of feature subspaces) or CNNs.

6. Conclusions

In this work we propose to weight ensembles, i.e. to modify the score values of the score vector to boost base classifiers which class is similar to the sample image. The scores of the base classifier are modified through weights, which are calculated depending on how similar is the current image to the different classes. We evidence, with the results obtained, that our method outperforms base classifier results and is a useful methodology for feature combination.

Experimental results showed that we can increase around a 5% accuracy for an OVO and OVA ensemble with little extra complexity. We have also shown that the combination between Bof and HOG gives the best results. Furthermore, evaluating our method

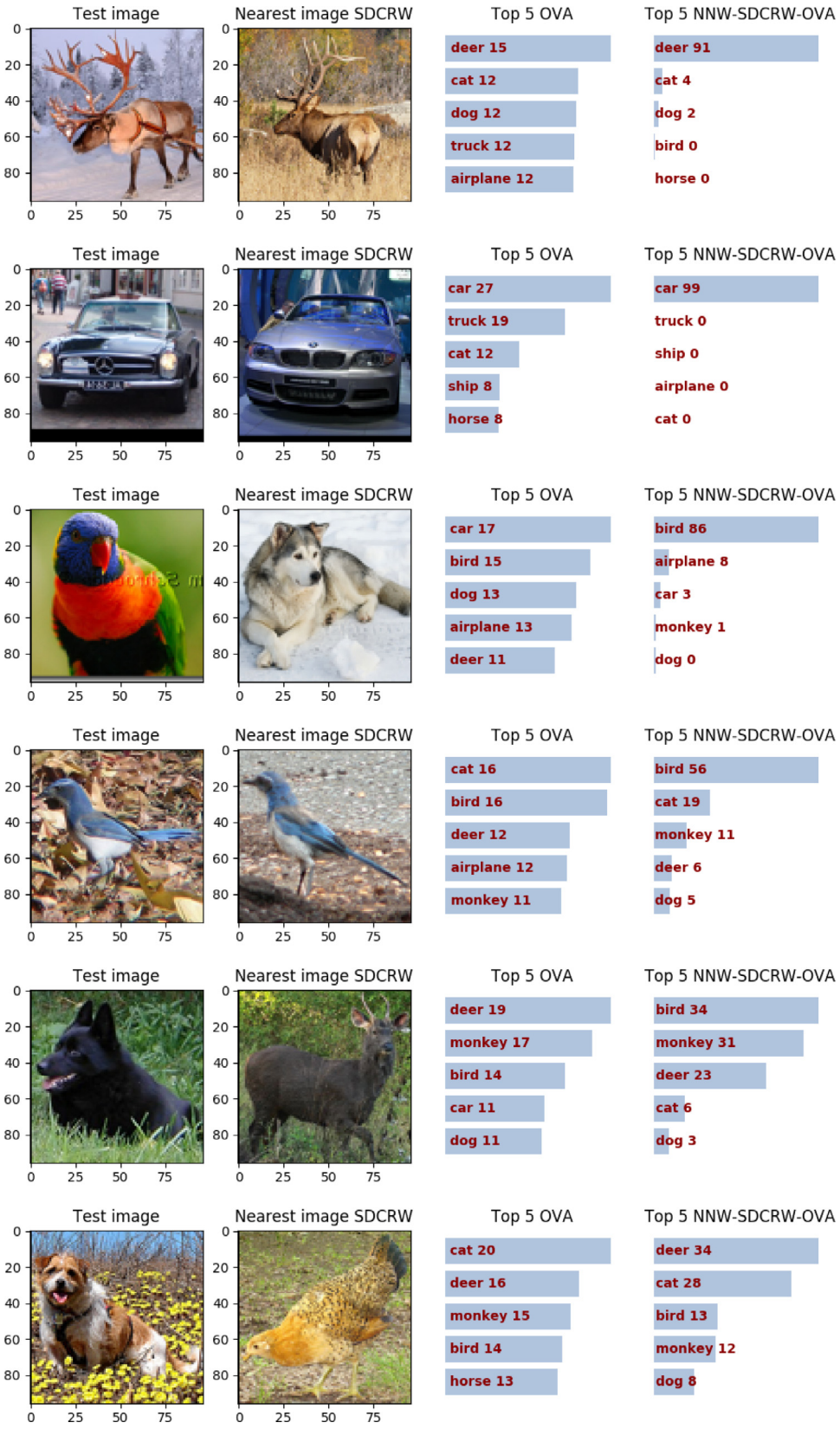


Fig. 4. Example of test images, their nearest images and the scores of a simple OVA and the NNW-SDCRW-OVA.

in the STL-10 image dataset we obtain similar results to the ones obtain by state-of-the-art results. The last main conclusion is that the Neural Network based methodology obtain in general better results than the expressions of Eq. (3) and Eq. (6).

Taking into account these results our proposal is a usable methodology for specific image classification problems when we have different feature vectors (so we can combine them positively) and we have few training examples.

There exist some aspects of our methodology that needs further research for example, how to calculate the distance of an image to a class, where other approaches like, unsupervised distance learning, should be investigated. Another important question is how to select feature vectors with complementary information in order to obtain the highest increase in accuracy. Regarding the OVO ensemble, we plan to research on the aggregation phase, due to there is still a big gap between the accuracy of the base classifiers and the final accuracy of the ensemble.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is partially supported by the research services of Universidad Publica de Navarra and by the project TIN2016-77356-P (AEI/FEDER, UE).

References

[1] A. Jain, D. Zongker, Feature selection: evaluation, application, and small sample performance, IEEE Trans. Pattern Anal. Mach. Intell. 19 (2) (1997) 153–158.
[2] B.T. N. Dalal, Histograms of oriented gradients for human detection., IEEE Conf. Comput. Vision Pattern Recognit. (2005).
[3] O. Chapelle, P. Haffner, V. Vapnik, SVMS for histogram-based image classification, L (3) (1999) 1–11.
[4] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.
[5] D. Karimi, R.K. Ward, Patch-based models and algorithms for image processing: a review of the basic principles and methods, and their application in computed tomography, Int. J. Comput. Assist. Radiol. Surgery 11 (10) (2016) 1765–1777.
[6] J. Sivic, A. Zisserman, Video google: a text retrieval approach to object matching in videos, IEEE Int. Conf. Comput. Vision (Iccv) (2003) 1470–1477.
[7] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. 2 (2006) 2169–2178.
[8] C.-W. Hsu, C.-J. Lin, A comparison of methods for multiclass support vector machines, IEEE Trans. Neural Netw. 13 (2) (2002) 415–425.
[9] A.C. Lorena, A.C.P.L.F. de Carvalho, J.M.P. Gama, A review on the combination of binary classifiers in multiclass problems, Artif. Intell. Rev. 30 (1) (2009) 19.
[10] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes, Pattern Recognit. 44 (8) (2011) 1761–1776.
[11] X. Feng, Z. Xiao, B. Zhong, J. Qiu, Y. Dong, Dynamic ensemble classification for credit scoring using soft probability, Appl. Soft Comput. 65 (2018) 139–151.
[12] Y. Zhang, D. Miao, J. Wang, Z. Zhang, A cost-sensitive three-way combination technique for ensemble learning in sentiment classification, Internat. J. Approx. Reason. 105 (2019) 85–97.
[13] J. Hou, M. Pelillo, A simple feature combination method based on dominant sets, Pattern Recognit. 46 (11) (2013) 3129–3139.
[14] S.F. Chevtchenko, R.F. Vale, V. Macario, F.R. Cordeiro, A convolutional neural network with feature fusion for real-time hand posture recognition, Appl. Soft Comput. 73 (2018) 748–766.
[15] D.J. Hemanth, J. Anitha, Modified genetic algorithm approaches for classification of abnormal magnetic resonance brain tumour images, Appl. Soft Comput. 75 (2019) 21–28.
[16] M. Galar, A. Fernández, E. Barrenechea, F. Herrera, DRCW-OVO: Distance-based relative competence weighting combination for One-vs-One strategy in multi-class problems, Pattern Recognit. 48 (1) (2015) 28–42.
[17] H. Bustince, M. Pagola, R. Mesiar, E. Hullermeier, F. Herrera, Grouping, overlap, and generalized bientropic functions for fuzzy modeling of pairwise comparisons, IEEE Trans. Fuzzy Syst. 20 (3) (2012) 405–415.
[18] K. Tan, W.S. Lee, H. Gan, S. Wang, Recognising blueberry fruit of different maturity using histogram oriented gradients and colour features in outdoor scenes, Biosyst. Eng. 176 (2018) 59–72.
[19] P.H. Kassani, A.B.J. Teoh, A new sparse model for traffic sign classification using soft histogram of oriented gradients, Appl. Soft Comput. 52 (2017) 231–246.
[20] L. Zadeh, Fuzzy Sets, Inform. Control 8 (1965) 338–353.
[21] J. Chamorro-Martínez, J. Soto-Hidalgo, P. Martínez-Jiménez, D. Sánchez, Fuzzy Color spaces: a conceptual approach to color vision, IEEE Trans. Fuzzy Syst. (2016).
[22] A discussion on fuzzy cardinality and quantification. some applications in image processing, Fuzzy Sets and Systems 257 (2014) 85–101.
[23] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.
[24] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, Dynamic classifier selection for One-vs-One strategy: Avoiding non-competent classifiers, Pattern Recognit. 46 (12) (2013) 3412–3424.
[25] H. Bustince, M. Pagola, E. Barrenechea, Construction of fuzzy indices from fuzzy DI-subsethood measures: Application to the global comparison of images, Inform. Sci. 177 (3) (2007) 906–929.
[26] M. Sesma-Sara, L.D. Miguel, M. Pagola, A. Burusco, R. Mesiar, H. Bustince, New measures for comparing matrices and their application to image processing, Appl. Math. Model. 61 (2018) 498–520.
[27] Y. Yang, S. Newsam, Bag-of-visual-words and spatial extensions for land-use classification, in: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10, 2010, pp. 270.
[28] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, Science Department, University of Toronto, Tech., 2009, pp. 1–60.
[29] A. Coates, A. Arbor, A.Y. Ng, An analysis of single-layer networks in unsupervised feature learning, Aistats 2011 (2011) 215–223.
[30] J.T. Springenberg, A. Dosovitskiy, T. Brox, M.A. Riedmiller, Striving for simplicity: The all convolutional net, CoRR abs/1412.6806 (2014).
[31] J. Mairal, P. Koniusz, Z. Harchaoui, C. Schmid, Convolutional kernel networks, 2014, pp. 1–9.
[32] O. Vinyals, Y. Jia, L. Deng, T. Darrell, Learning with recursive perceptual representations, Nips (2012) 1–9.
[33] A. Dosovitskiy, J.T. Springenberg, M. Riedmiller, T. Brox, Discriminative unsupervised feature learning with convolutional neural networks, in: Proceedings of NIPS, Adv. Nueral Inform. Process. Syst. 27 (2014) 1–13.
[34] T.L. Paine, P. Khorrami, W. Han, T.S. Huang, An analysis of unsupervised pre-training in light of recent advances, Iclr (2015) 1–10.

2. Aprendiendo operaciones de *pooling* basadas en medias ponderadas ordenadas para clasificación de imágenes

J.I. Forcén, M. Pagola, E. Barrenechea, H. Bustince

Learning ordered pooling weights in image classification

Estado: ***Publicado***

Revista: ***Neurocomputing***

DOI: `https://doi.org/10.1016/j.neucom.2020.06.0288`

Índice de impacto: ***4.438 - Q1 JCR.***

Áreas de conocimiento: ***Computer Science, Artificial Intelligence***

Neurocomputing 411 (2020) 45–53

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Learning ordered pooling weights in image classification

J.I. Forcén [a,*], Miguel Pagola [b], Edurne Barrenechea [b], Humberto Bustince [c]

[a] Das-nano-Veridas, 31192 Tajonar, Spain
[b] Universidad Pública de Navarra Campus Arrosadía, 31006 Pamplona, Spain
[c] King Abdullazih Universitiy, Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 19 June 2019
Revised 6 April 2020
Accepted 9 June 2020
Available online 16 June 2020
Communicated by Xinmei Tian

Keywords:
Pooling
Ordered weighted aggregation
Image classification
Bag-of-words
Mid-level features
Convolutional neural networks
Global pooling

ABSTRACT

Spatial pooling is an important step in computer vision systems like Convolutional Neural Networks or the Bag-of-Words method. The spatial pooling purpose is to combine neighbouring descriptors to obtain a single descriptor for a given region (local or global). The resultant combined vector must be as discriminant as possible, in other words, must contain relevant information, while removing irrelevant and confusing details. Maximum and average are the most common aggregation functions used in the pooling step. To improve the aggregation of relevant information without degrading their discriminative power for image classification, we introduce a simple but effective scheme based on Ordered Weighted Average (OWA) aggregation operators. We present a method to learn the weights of the OWA aggregation operator in a Bag-of-Words framework and in Convolutional Neural Networks, and provide an extensive evaluation showing that OWA based pooling outperforms classical aggregation operators.

1. Introduction

Image classification is one of the main problems in computer vision and pattern recognition, which plays an important role in scene understanding, object categorization, and many other vision tasks.

Pooling is an essential step in the state-of-the-art image classification methodologies as Convolutional Neural Networks (CNN) [1] or Spatial pyramid framework [2]. Pooling is an operation which aggregates local features into a new and more usable vector. The aggregation of local feature vectors produce more compact representations and improve robustness to noise, clutter and invariance to image translation and transformations.

The result of a pooling operation depends on the aggregation operator g used, because it defines how the local features are aggregated. The most common operators g are the maximum, used in the well-known network architectures AlexNet [3] or VGG [4] and the arithmetic mean used in Network [5] or GoogleNet [6]. Despite having a large influence on the performance of the network, most models use the maximum or the arithmetic mean [7]. The maximum and the mean have the advantage that, no parameters are learned in the pooling layers, however, the pooling operator g used in each layer becomes an hyper-parameter of the network. Therefore, it is another hyper-parameter of the network as the number of layers, number of filters per layer, the size of the filters, learning ratio, etc. It is well known that the selection, for a given problem, of the most appropriate network architecture and hyper-parameters has a high computational cost [8].

Let's suppose a trained CNN with several layers, if the image I which is being processed by the network contains, for example, a highly defined corner, it will produce a very high activation value in the feature channel where the corners are represented. If the maximum operator is used in the following pooling layer, said value (which is very high) will spread through the network architecture and become part of the final feature vector representation of the image. Therefore, if such corner is representative of the class of the image, then it will be a discriminative feature and will serve to correctly classify the image.

Maximum pooling tends to give more importance to high activations, regardless of their frequency in the image, on the other hand the arithmetic mean filters the features that appear in isolation and end up having a smaller value in the final representation.

The selection of the pooling operator g is not a trivial decision because rely on each problem, and is influenced by factors like the images themselves, the features extracted from the images or the architecture proposed.

* Corresponding author.
E-mail address: jiforcen@das-nano.com (J.I. Forcén).

https://doi.org/10.1016/j.neucom.2020.06.028
0925-2312/

In this paper we propose a new method called OWA-pooling, in which the aggregation is made by a weighted average of the ordered elements. In Ordered Weighted Aggregation functions, the weights are associated not with particular inputs, but with their magnitudes (in the case of a convolution the weights are associated to their spatial position in the filter). The weights will be learned in the training phase such a way the pooling function will propagate the highest values activations, but also takes into account activation values that appear frequently.

In summary, the main contributions of this paper are the following ones:

1. We propose to use Ordered Weighted Operators as the pooling operation in image classification methods.
2. We provide the methodology to learn the weights of the OWA-pooling operator such a way we obtain a parametrized transition between maximum and average pooling.
3. We analyse the performance of the OWA-pooling in two different image classification frameworks: Bag-of-Words features with a global spatial pooling [2] and Convolutional Neural Networks.
4. We show experimentally on public benchmarks that the proposed pooling method can provide performance gains with respect average pooling and maximum pooling.

The remaining sections of this paper are organized as follows. Section 2 contains a literature survey of related works. Section 3 presents a preliminary study of the proposed methodology in the Bag-of-Features methodology. In Section 4 the pooling methodology is described in CNNs. In both Sections 3 and 4 some experimental results are provided. We conclude in Section 5.

2. Related work

Previous to the fast development of high-performance hardware and the great success of CNNs in image recognition, Bag-of-Words (BoW) was the state of the art methodology in image classification [2]. BoW usually includes the following steps: feature detection, feature description known as coding, and pooling to obtain a final feature vector of the image. The final feature vectors are used to classify the image by means of a classifier, usually a Support Vector Machine (SVM). The process to obtain the final feature vector could be repeated on and on for regions of the image, like the layers of a network [9], therefore in [10,11] it is argued that the conclusions obtained for a pooling operator in a BoW model can be assumed in CNN models.

In [7,11] Boureau et al. provide a theoretical study and an empirical comparison between maximum and average pooling in the BoW model applied to image classification. They found that maximum pooling is much better than average pooling, particularly, it is well suited for very sparse feature vectors. Furthermore, they prove that for binary feature vectors, using the Bernoulli distribution under the i.i.d. assumption, the maximum expectation outperforms both average and maximum. The idea of using an expectation of the maximum comes from the fact that a negative factor of the pooling operation is to favour high values of local descriptors with low representativeness of the object of the image. P. Koniusz et al. [10] try to solve this problem by simply pooling over the n largest values. Experiments carried out in [10] use a fixed cardinality n of three, seven or fifteen. In [12] we studied the cardinality of the pooling operator that performs the average on the $\mathcal{N}$ major values (using BoW and Spatial Pyramid). Also in [13] we verified that the ordered weighted averages with fixed weights also give good results in this image classification methodology.

Regarding CNNs, Scherer et al. [14] showed that a max pooling operation is superior compared to sub-sampling operations, however they found that max pooling easily overfits the training set. In stochastic pooling [15], the pooled map response is generated by sampling from a multinomial distribution formed from the activations of each pooling region improving the performance in some datasets. In [16] a global spatial structure is proposed with multiple Gaussian distributions which pools the features according to the relations between features and Gaussian distributions.

Due to hyper-parameter selection is a major drawback in CNNs, there are some works that propose to learn the pooling operator. For example, in [17] is proposed the operator AlphaMEX which is a non-linear smooth log-mean-exp function to perform a global pooling before the softmax layer of a CNN. In [18] was proposed a general pooling framework that captures higher order interactions of features in the form of a Gaussian Radial Basis Function kernel. Another two methods are proposed in which the pooling function is learned in [19]. In the first strategy, a mixing parameter of the maximum value and the arithmetic mean value is learned. In the second method, a function of pooling is learned in the form of a tree that mixes the results of different pooling filters. Sun et.al. [20] propose a learned pooling operation as a linear combination of the neurons in the region for each feature channel.

Basically, all of these methods try to find a method with the of max pooling and average pooling, but avoiding their drawbacks.

3. Learning ordered weighted pooling weights in the Bag-of-Words setting

In this section we present a first analysis of learning weights in ordered weighted pooling in the BoW model for image classification. First, we recall the BoW method, that can be summarised in the following steps:

1. First, local image descriptors, such as SIFT [21], HOG [22] or Gabor are extracted from images at interest points or in a dense grid.
2. An unsupervised learning algorithm is used to discover a set of prototype descriptors that is called a dictionary.
3. In the feature coding step, image descriptors are locally transformed in a new vector decomposing the initial descriptor on the dictionary. It can be understood as an activation function for the dictionary, activating each of the visual words according to their similarity with the local descriptor.
4. In the pooling step the codes associated with local image features are combined over some image neighbourhood. The codes within each cell are aggregated to create a single feature vector.
5. Training and classification can be performed on the final feature vectors (usually the concatenation of the signature feature vectors of different cells) by a classifier, e.g. SVM [23].

Also, we introduce some notation used throughout the work. Let an image I be represented by a set of low-level descriptors or local features (e.g. SIFT) $\boldsymbol{x_i}$ at $\mathcal{N}$ locations identified with their indices $i = 1, \ldots, N$. The signature vector z representing the whole image is obtained by sequentially coding (Eq. 1) and and pooling (Eq. 2) over all descriptors:

$$\boldsymbol{\alpha}_i = f(\boldsymbol{x}_i), i = 1, \ldots, N \tag{1}$$

$$\boldsymbol{z} = g(\{\boldsymbol{\alpha}_i\}_{i \in \mathcal{N}}) \tag{2}$$

We will denote as f and g, the coding and pooling operators respectively and $\boldsymbol{\alpha}_i$ the coded vector. The classification performance using $\boldsymbol{z}$ as the input of a classifier (e.g. SVM [23]) depends on the properties and the combination of f and g.

In average pooling, the obtained vector $\boldsymbol{z}$ is the average over the activations $\boldsymbol{\alpha}_i$ of the elements of the image for each component of the coded vector.

$$\boldsymbol{z} = \frac{1}{|\mathcal{N}|}\sum_{i\in\mathcal{N}}\boldsymbol{\alpha}_i \tag{3}$$

Whereas in maximum pooling selects the largest value between the activations of the elements of the image for each component of the coded vector.

$$z_j = \max_{i\in\mathcal{N}}\{\alpha_{i,j}\}, \text{for} j = 1,\ldots,K \tag{4}$$

being K the number of visual words, i.e. the length of the coded vector.

3.1. Ordered weighted pooling

OWA functions belong to the class of averaging aggregation functions. They differ to the weighted arithmetic means in the weights, that are associated not with particular inputs, but with the input magnitude. Formally, an OWA operator of dimension n is a mapping $\phi_w : \mathcal{R}^n \to \mathcal{R}$that has an associated collection of weights $\boldsymbol{w} = (w_1,\ldots,w_n)$ lying in the unit interval and summing to one. They were introduced by Yager [24]. We recall the notation of an ordered vector as $(\boldsymbol{z}\searrow) = z_1 \geqslant z_2 \geqslant \ldots \geqslant z_n$.

$$\phi_w(\boldsymbol{z}\searrow) = \sum_{i=1}^{n} w_i z_i \quad \text{s.t.} \quad \sum_{i=1}^{n} w_i = 1 \quad \text{and} \quad w_i \geqslant 0 \quad \text{for every} \quad i = 1,\ldots,n. \tag{5}$$

Obviously the calculation of the value of an OWA function involves sorting the array of values to be aggregated. We can obtain typical aggregation functions [25], with specific weights, for example:

- If $\boldsymbol{w} = (0,0,\cdots,0,1)$,then ϕ_w = minimum.
- If $\boldsymbol{w} = (1,0,\cdots,0,0)$, then ϕ_w = maximum.
- If $\boldsymbol{w} = (\frac{1}{n},\frac{1}{n},\cdots,\frac{1}{n},\frac{1}{n})$, then ϕ_w = arithmetic mean.

Let $\boldsymbol{z}^{(I)}$ the final feature vector of an image I. This final vector is the aggregation of all of the coded vectors $\boldsymbol{\alpha}$ of the dense grid of size $\mathcal{N}$. So, following Eq. (2), an element j of the vector $\boldsymbol{z}^{(I)}$ is calculated with the expression:

$$z_j^{(I)} = \boldsymbol{w}\cdot(\boldsymbol{\alpha}_j\searrow) \tag{6}$$

3.2. Learning the weights

Let's suppose that the classifier used is a linear kernel support vector machine L2-SVM. The L2-SVM uses the square sum of the slack variables (the regularization term) which makes the SVM less susceptible to outliers and improve its overall generalization. Next equation shows the cost function $J(\Theta)$ of an L2-SVM, being Θ the parameters of the model, $\boldsymbol{z}^{(i)}$ the final feature vector of an example, $y^{(i)}$ the class of that example, m the number of training examples, K number of features, i.e. the length of the feature vector and C_1 the regularization parameter:

$$J(\Theta) = \frac{C_1}{m}\sum_{i=1}^{m} max\left(0, 1 - \Theta^T\boldsymbol{z}^{(i)}y^{(i)}\right)^2 + \frac{1}{2}\sum_{j=1}^{K}\Theta_j^2 \tag{7}$$

Due to we want to learn also the weights of the pooling operator $\boldsymbol{w}$, if we substitute $\boldsymbol{z}$ using Eq. (6), the cost function remains as follows:

$$J(\Theta, w) = \frac{C_1}{m}\sum_{i=1}^{m} max\left(0, 1 - \Theta^T(\boldsymbol{w}\cdot\boldsymbol{\alpha}\searrow)^{(i)}y^{(i)}\right)^2 + \frac{1}{2}\sum_{j=1}^{K}\Theta_j^2 \tag{8}$$

Moreover we add a regularization term to smooth distribution of weights for similar activations, such a way we should interpret easily the weights obtained.

$$J(\Theta, w) = \frac{C_1}{m}\sum_{i=1}^{m} max\left(0, 1 - \Theta^T(\boldsymbol{w}\cdot\boldsymbol{\alpha}\searrow)^{(i)}y^{(i)}\right)^2 + \frac{1}{2}\sum_{j=1}^{K}\Theta_j^2 + +C_2\sum_{i=1}^{\mathcal{N}}\sum_{j=1}^{\mathcal{N}}A_{ij}(w_i - w_j)^2 \tag{9}$$

where C_2 is th regularization parameter and A is a matrix with elements $A_{ij} = 1$ if $j = i+1$ and $A_{ij} = 0$ otherwise. However, the pooling weights must satisfy the restrictions $\sum_{i=1}^{\mathcal{N}} w_i = 1$, and $\forall w_i \geqslant 0$ therefore the training step is a constrained optimization problem. Using the method of Lagrange multipliers to convert it into an unconstrained problem:

$$\mathcal{L}(\Theta, w, \lambda, \mu) = J(\Theta, w) + \lambda\left(\left(\sum_{i=1}^{\mathcal{N}} w_i\right) - 1\right) + \sum_{i=1}^{\mathcal{N}} - w_i * \mu_i \tag{10}$$

The following equations are the partial derivatives of the parameters $\Theta, \boldsymbol{w}$ and the Lagrange multipliers λ and μ_i, which can be used in any gradient descent based optimization algorithm:

$$\frac{\partial\mathcal{L}(\Theta, w, \lambda, \mu)}{\partial\Theta_j} = \frac{-2C_1}{m}\sum_{i=1}^{m} max(0, 1 - \Theta^T(\boldsymbol{w}\cdot\boldsymbol{\alpha}\searrow)^{(i)}y^{(i)})(w\cdot\boldsymbol{\alpha}\searrow)^{(i)}y^{(i)} + \Theta_j \tag{11}$$

$$\frac{\partial\mathcal{L}(\Theta, w, \lambda, \mu)}{\partial w_j} = \frac{-2C_1}{m}\sum_{i=1}^{m} max(0, 1 - \Theta^T(\boldsymbol{w}\cdot\boldsymbol{\alpha}\searrow)^{(i)}y^{(i)})\Theta^T(w_j\cdot\boldsymbol{\alpha}\searrow)^{(i)}y^{(i)} + 2C_2\sum_{i=1}^{\mathcal{N}}\sum_{j=1}^{\mathcal{N}}A_{ij}(w_i - w_j) + \lambda - \mu_j \tag{12}$$

$$\frac{\partial\mathcal{L}(\Theta, w, \lambda, \mu)}{\partial\lambda} = \left(\sum_{i=1}^{\mathcal{N}} w_i\right) - 1 \tag{13}$$

$$\frac{\partial\mathcal{L}(\Theta, w, \lambda, \mu)}{\partial\mu_j} = -w_j \tag{14}$$

Instead of learning all of the parameters (Θ and w) in the training process, we can simplify and learn the parameters in an iterative way, first learning the parameters Θ of the classifier, then find the optimal values of the OWA $\boldsymbol{w}$. Then using the OWA obtained, retrain the parameters Θ of the classifier (being the initial Θ, the one obtained in the first optimization), next, retrain the parameters of the OWA and so on, until convergence, i.e. the parameters do not change. We have used this methodology due to it avoids falling in saddle points and achieves better optimization than training all of the parameters (Θ and w) together from scratch.

3.3. First experiment

In this experiment we evaluate the performance of OWA-pooling by means of the classification accuracy in the 15-Scenes dataset (see Fig. 1)on a BoW setting [2]. Low-level descriptors x_i are 128-dimensional SIFT descriptors [21] of 32 × 32 patches. The descriptors are extracted on a dense grid every 32 pixels, such that the image is divided in 64 cells. The dictionary $\boldsymbol{D}$ is calculated from 200.000 random descriptors. The vocabulary sizes tested are $[2^4, 2^5, .., 2^9]$. The coding step is carried out by two different methods, triangle assignment [9] and sparse coding [26]. The same dic-

Fig. 1. Sample images from 15-Scenes dataset.

tionary was used in both coding methods. Following the usual procedure, we use 150 training images and the rest for testing on the 15-Scene dataset. Experiments are conducted over 10 random splits of the data, and we report the test mean accuracy. The value of the regularization parameters C_1 and C_2 are selected by cross-validation within the training set.

In this first experiment we aggregate the feature vectors of the whole image in the pooling step, without creating regions (like the spatial pyramid method), so that the overlap between regions does not interfere in the results and the performance of the pooling operators is clearly reflected in the results.

Our hypothesis is that the pooling weights learned on the iterative process will be adapted to the data and the features with high activation but low representativeness of the object of the image should be filtered. The results depicted in Table 1, shows that the proposed method improves the results over maximum and average pooling. Especially for features coded with spare coding, we obtain more than a 10% increase in the performance. To analyse the relation between the feature values and their representativeness, in Fig. 2 we represent the contribution of each one of the 64 cells on the sum of $\boldsymbol{w}\cdot(\alpha\searrow)$ (i.e. the value in the final vector) in an image that is misclassified by the maximum pooling. We can see that using the maximum, few cells contribute to the final representation, but the OWA pooling distribute better the contribution of different parts of the image (in the case of average pooling all of the cells have the same importance).

4. Learning ordered weighted pooling in convolutional neural networks

In this section we describe how we can use an OWA-pooling in a CNN. Let's suppose that we have an OWA-pooling after one convolutional layer and we want to learn its parameters $\boldsymbol{w}$ at the same time that we train the network. The pooling parameters could be learned similar to the convolutional filters, because the calculation of the gradient $\frac{\partial J}{\partial w_i}$ is similar to that of a convolution parameter, taking into account that it is necessary to sort the values of the activation layer. However, the pooling parameters must satisfy the restrictions $\sum_{i=1}^{\mathcal{N}} w_i = 1$, and $\forall w_i \geqslant 0$, so we add three terms to the cost function:

$$J = J_{CE} + C_1 \sum_{i=1}^{N} max\{0, -w_i\} + C_2 \left(\left(\sum_{i=1}^{\mathcal{N}} w_i \right) - 1 \right)^2 + C_3 \left(\sum_{i=1}^{\mathcal{N}-1} (w_i - w_{i+1})^2 \right) \tag{15}$$

where J_{CE} is the cost function of the CNN (for example the cross correlation) and C_1, C_2 and C_3 are regularization parameters. The first term trigger the values to be greater than zero, the second term penalizes if the sum of weights is not one and the third term penalizes the differences between the consecutive weights. Therefore, the calculation of the gradient to be used in the optimization algorithm is:

$$\frac{\partial J}{\partial w_i} = \frac{\partial J_{CE}}{\partial w_i} - C_1 + 2C_2 \left(\left(\sum_{i=1}^{\mathcal{N}} w_i \right) - 1 \right) w_i + 2C_3 (w_i - w_{i+1}) \tag{16}$$

If $w_i \geqslant 0$ the term C_1 disappears.

We are going to compare the performance of the OWA-pooling in two different scenarios, in the first experiment, we are going to learn the weights of an OWA-pooling in a small CNN. In the second experiment, we will test the performance of the OWA pooling used

Table 1
Accuracies in 15-Scenes dataset with different codings, dictionary sizes and pooling operations.

Coding	Triangle assignment (% accuracy)					Sparse (% accuracy)				
Dic.Size	32	64	128	256	512	32	64	128	256	512
MAX	43.89	48.94	55.38	60.23	63.09	48.7	55.91	61.78	64.82	68.76
MEAN	39.41	48.02	48.17	48.56	49.43	54.15	61.06	63.06	64.21	64.39
OWA	**45.99**	**49.75**	**55.57**	**60.40**	**64.74**	**58.18**	**65.05**	**71.05**	**74.89**	**80.26**

Fig. 2. Contribution of each cell in the final vector representation. The importance of each cell is superimposed in the image, transparent cells are more important and darker cells are less important. (a) Original image, (b) using MAX pooling and (c) using OWA-pooling.

as a global pooling operation in deeper networks such as VGG and mobilnet.

4.1. First experiment

In this experiment we want to check the accuracy of already known and validated CNN when we replace the original pooling operator with our proposed OWA-pooling. We also want to check if the weights converge towards pooling operators similar to the original operators ($(1,0,0,\cdots,0)$for the maximum or $(\frac{1}{n},\frac{1}{n},\cdots,\frac{1}{n})$for the mean). We are going to use the CNN proposed in [5] which is known as Network in Network (NiN). Its architecture is shown in Table 2, where all convolutions are followed by activations type Relu.

We have trained the network with CIFAR-10 and CIFAR-100 [27] datasets, which contains 50,000 color images for training and 10,000 for test. The images have a resolution of 32 × 32 pixels and there 10 and 100 categories respectively. In Table 3 we compare different pooling configurations:

- Orig: is the original pooling operations as shown in Table 2.
- MAX: Maximum pooling in the three layers.
- AVE: Average pooling in the three layers.
- OWAL: learn OWA weights for each layer.
- OWALnr: learn OWA weights for each layer without restrictions.
- OWALC: learn OWA weights for each channel of each layer.
- OWALCnr: learn OWA weights for each channel of each layer without restrictions.

Table 2
NiN architecture.

Input		Filters channels
32 × 32		5 × 5, 192
32 × 32		1 × 1, 160
32 × 32		1 × 1, 96
32 × 32	pool1	3 × 3 Max *pooling*, stride 2
16 × 16		dropout 0.5
16 × 16		5 × 5, 192
16 × 16		1 × 1, 192
16 × 16		1 × 1, 192
32 × 32	pool2	3 × 3 Ave *pooling*
8 × 8		dropout 0.5
8 × 8		5 × 5, 192
8 × 8		1 × 1, 192
8 × 8		1 × 1, 10 o 100
8 × 8	pool3	8 × 8 Ave *pooling*
10 o 100		Softmax

Table 3
Accuracies in CIFAR-10 and CIFAR-100 with different pooling operations.

Pooling	CIFAR-10 (% error)	CIFAR-100 (% error)
Orig	9.76	41.30
MAX	11.24	45.11
AVE	9.50	41.25
OWAL	9.66	40.67
OWALnr	**9.42**	**40.44**
OWALC	9.68	41.91
OWALCnr	9.64	41.33

The results obtained verify that when OWA-pooling is used, a similar or better accuracy is reached compared with the original. Moreover, when the weights are trained without restrictions, the accuracy is slightly higher. It is also verified that fitting a pooling operator for each channel of each layer is not necessary in this case, in fact, the accuracy of OWALC is lower than OWAL.

4.1.1. OWA pooling and robustness

Adding more trainable parameters to the network usually means networks proned to overfitting and therefore with less generalization capacity. To check this fact we have carried out a robustness test [19]. In this test, the test images are rotated between −8 and 8 degrees and we compute the accuracy with the trained networks (Fig. 3). We verify that in the case of NiN, the model with OWAL is the most robust to changes. Therefore, to weight the activations taking into account only their value, causes the representative feature of the image to spread through the network to the final representation and improve the classification.

4.1.2. OWA pooling and regularization

In this section, we have trained the NiN model with OWA pooling layers, using two recent regularization techniques. Our objective is to verify if OWA pooling layers are complementary with regularization, leading to an improved result. We have tested two different recent regularization approaches in CNNs: (1) data augmentation by means of random erasing [28] and (2) a structured dropout called DropBlock [29]. In training, random erasing randomly selects a rectangle region in an image and erases its pixels with random values. We have trained the NiN model (both, the original architecture and using OWA pooling in every pooling layer) in the CIFAR-10 dataset with random erasing; this regularization does not have a positive effect on the results, actually the accuracy in both models, with and without random erasing was almost the same as in Table 3. This fact may be because the NiN network is not a deep network compared to ResNet or VGG, where

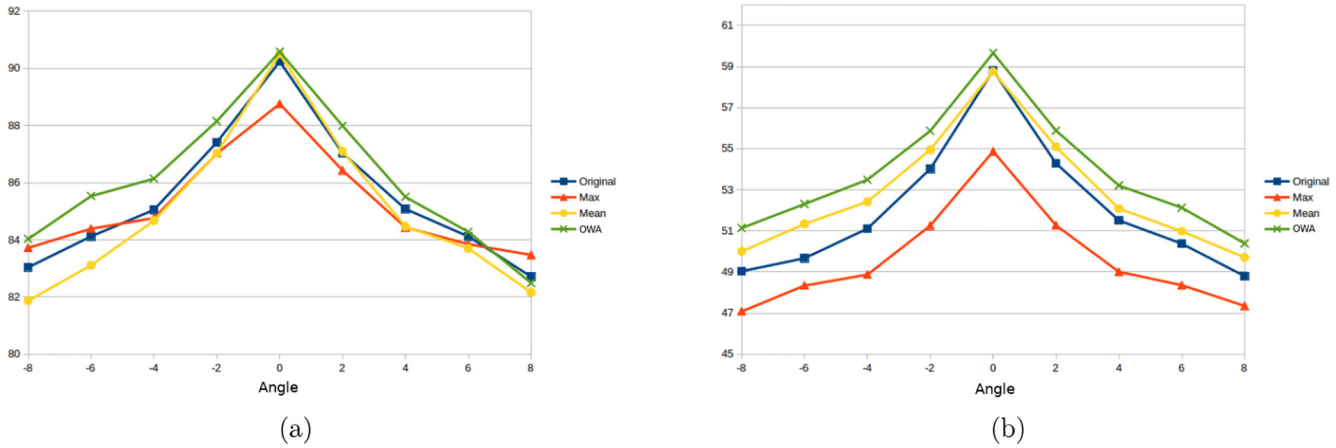


Fig. 3. Robustness to rotation. Vertical axis of the graphs represents the accuracy and the horizontal axis the angle that the images are rotated. (a) Results for NiN trained in CIFAR-10 and (b) Results for NiN trained in CIFAR-100.

other authors have proved that random erasing increases accuracy in CIFAR-10.

However, the structured dropout DropBlock [29], where units in a contiguous region of a feature map are dropped together, has improved the accuracy in our experiments. We have tested different block sizes and probabilities and the results show that the model trained with DropBlock increases accuracy. The error of the best OWA model decreases from 9.42% to 8.26% which is a very important improvement (Table 4). Therefore, OWA pooling is complementary with DropBlock and can be used to increase the accuracy. Even, the largest improvement is obtained when the NiN network pooling operators are OWAs.

4.1.3. Weight analysis

In the Fig. 4 are depicted the weights of the three pooling layers learned for NiN network in the OWAL case for CIFAR-10 (weights for CIFAR-100 are similar). In both figures the weights are depicted in order, i.e. the weight in position 1 will multiply the largest value of the vector, the second weight with the second largest and so on. Fig. 4(a) represent the nine weights for first and second pooling layer, for both layers weights are around the average but both giving more importance to the smaller activations. Fig. 4(b) shows the sixty-four weights of the third layer which also boost lower activations.

4.1.4. Performance analysis

The biggest problem with OWA pooling is the need of sorting the activation values. The introduction of the magnitude order into the training algorithm, makes computing times grow. In the Table 5 we show the increase of time of the different models with respect to the original. The time cost of the OWAL model is not too large and it is probably faster than test the combinations of maximum and average pooling operators for each layer of the network (for small networks). Therefore the main conclusion of this experiment is that, OWA pooling can be used in real cases in which we do not know the correct architecture and we decided to introduce in the training step the pooling operator learning.

4.2. Second experiment. Global pooling layer

This second experiment is motivated by two factors. Firstly, pooling operations in intermediate layers of neural networks usually aggregate only a few values, (usually 2x2 or 3x3 pool size), compared with our first BoW experiment where 64 values were aggregated. So we wonder if this operation is more important when bigger regions are aggregated like the global pooling. Second, as we discuss in the previous section, OWA-pooling increases the execution time of the pooling layers, this problem is bigger as deeper is our neural network and as larger are the input dimensions of our image. For example, for an input image of 224x224x3 pixels that after initial convolutions have a size of 222 width x 222 height x 32 filters a 2 x 2 pooling size will produce 394272 pooling operations. Taking into account this two aspects, we decided to study the performance of applying our OWA-pooling only as global pooling layer. This global pooling layer is common among different well known architectures.

We study the performance in two well-known neural network architectures, VGG13 and MobileNet. VGG architecture do not contain a global pooling layer like MobileNet, so in our experiments we use the original architecture till the pool5 layer, where we concatenate a global pooling layer followed by a softmax.

Two datasets were used, 15-Scenes and Caltech-UCSD Birds (CUB200). CUB200 is a dataset with 6033 bird images classified in 200 different species. For 15-Scenes experiments we set as image size 256x256 pixels, using 150 images per class for training purposes. In Cub200 experiments we rescale images to 224x224 pixels and we split the dataset using 80% of the images for training and the other 20% for validation and test purposes. We made ran-

Table 4
Comparison of error percentage in CIFAR-10 of the NiN network trained without regularization (Original) and with DropBlock in different configurations (being, MAX, AVE and OWA the pooling operators are maximum, the average and OWA, respectively). s_1 and s_2 are the size of the blocks in layers 1 and 2 respectively and p the keep probability.

	NiN	MAX	AVE	OWA
Original	9.76	11.24	9.5	**9.42**
$s_1 = 4, p_1 = 0.8, s_2 = 2, p_2 = 0.8$	9.09	11.26	9.3	9
$s_1 = 5, p_1 = 0.8, s_2 = 3, p_2 = 0.8$	8.92	10.32	8.5	8.31
$s_1 = 5, p_1 = 0.5, s_2 = 3, p_2 = 0.5$	9.85	12.71	9.42	8.89
$s_1 = 7, p_1 = 0.8, s_2 = 5, p_2 = 0.8$	8.71	10.44	8.63	**8.26**
$s_1 = 7, p_1 = 0.5, s_2 = 5, p_2 = 0.5$	9.93	14.3	9.35	9.14

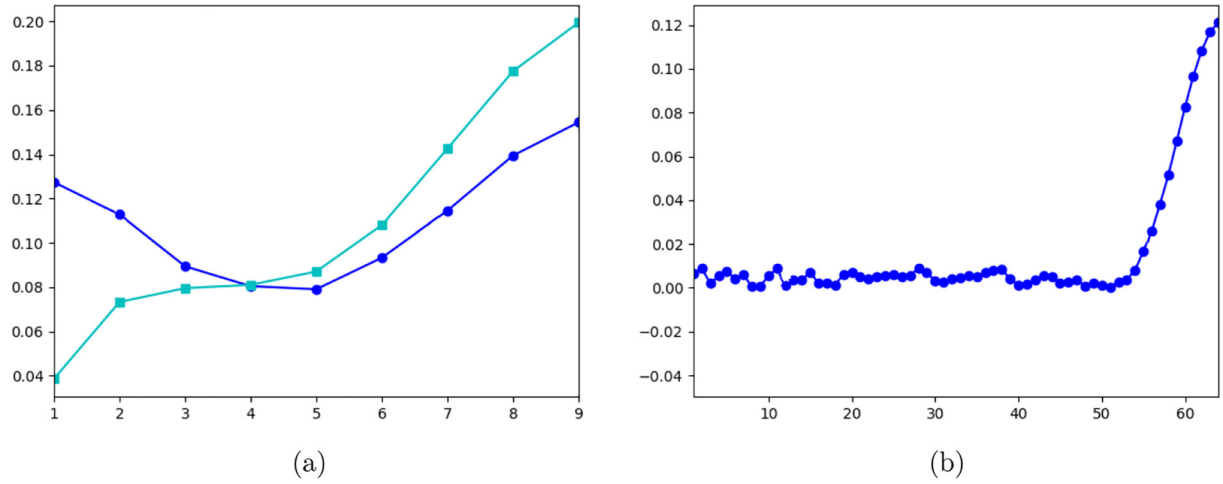


Fig. 4. (a) 9 learned weights of the OWA-pooling for layers 1 and 2 of NiN trained with CIFAR-10. (b) 64 learned weights of tthe OWA-pooling for layer 3 of NiN trained with CIFAR-10.

Table 5
Comparison of training times with respect to the original NiN architecture.

NiN	Orig	MAX	AVE	OWA
Time	1	1.02×	0.95×	4.06×

dom data augmentation in both datasets by horizontal flips, rotations, translations and zooms. The parameters used in 15 scenes were: 10 degrees of rotations, 15% translations, and 10% zoom. In CUB200 experiments were: 15 degrees of rotations, 15% translations, 15% zoom and 15% shear. All of the experiments in this two datasets began with pre-initialized Imagenet weights for both architectures. Then, the model is trained a few epochs with different aggregations in the global pooling layer. The number of epochs that is trained each model depends on its own convergence. CUB200 VGG was trained 300 epochs and CUB200 MobileNet 150 epochs. For 15-Scenes, the net VGG was trained for 25 epochs and MobileNet 50 epochs.

Table 6
Classification error in CUB and 15-Scenes with different global pooling operations

Model	Global Pooling	CUB (300 epochs) (% error)	15-Scenes (25 epochs) (% error)
VGG	Orig (+ 4million param.)	27.92	8.92
VGG	AVE	28.29	14.94
VGG	MAX	25.06	11.61
VGG	OWA	**23.2**	**10.76**
VGG	OWAco	23.78	10.85
VGG	OWAnr	24.48	10.88
VGG	OWAC	23.94	12.21
VGG	OWACco	23.49	11.32
VGG	OWACnr	24.28	10.92
		CUB (150 epochs) (% error)	15Scenes (50 epochs) (% error)
MobileNet	AVE	22.29	6
MobileNet	MAX	25.77	7.82
MobileNet	OWA	23.94	6.42
MobileNet	OWAco	23.69	6.22
MobileNet	OWAnr	26.1	6.53
MobileNet	OWAC	**22.43**	5.88
MobileNet	OWACco	23.62	**5.75**
MobileNet	OWACnr	23.94	6.42

In Table 6 we show the results obtained for different global pooling operators: AVE (average mean), MAX (maximum), OWA, OWAco (OWA-pooling constrains are implemented by means of the constrains module from keras instead of the regularization formulation of Eq. (15)), OWAnr (uncostraied weights) and OWAC (a OWA-pooling operator is learned per channel).

Results using OWA-pooling for VGG architecture are better compared with average or maximum. It is important to take in account that the original version of VGG for the 15-Scenes experiments is better than the rest of values, this accuracy should be seen as a reference, because using the global pooling instead of the fully connected layers means that our proposed test architecture has 4 million less parameters. Also it is remarkable to see that for the CUB200 experiment the better experiment using OWA-pooling outperforms the original model even with significantly fewer parameters.

For MobileNet, in the CUB200 dataset, the best global pooling is the average, which is the pooling operator of the original architecture, but the best OWA reaches almost similar error. However in 15-Scenes, the best result is obtained learning a OWA-pooling for every channel. Summing up, we can conclude again that our proposed aggregation technique produce better results than the classical average or maximum aggregation, and in the worst case results are similar. This take sense if we consider that for some problems maximum pooling and average pooling can be the best option, and this aggregation functions are particular cases of OWA pooling that can be learned during the training process. .

4.2.1. Weights analysis

In Fig. 5 are shown the weights of two different cases of this second experiment. In the left image are shown the 64 weights using the VGG model with the 15-Scenes dataset (OWA). In the right image are represented the 49 weights for each pooling function learned in OWAC using MobileNet model for the CUB200 dataset. The mobilenet architecture used is 1024 dimensions depth in the global pooling operation, so in this image are represented these 1024 learned weight vectors. In both cases, the resultant aggregation functions penalize the highest value but give more weights to

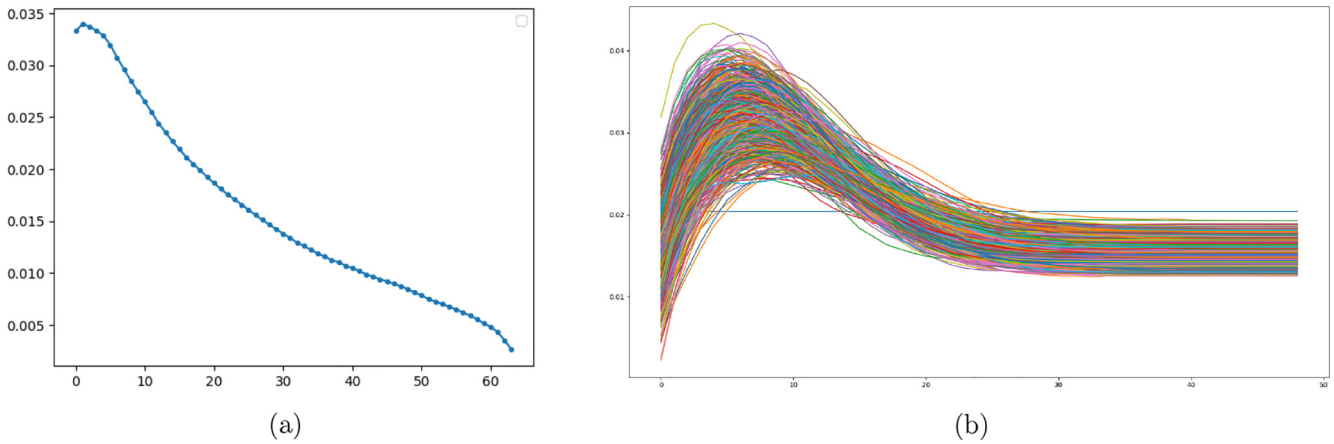


Fig. 5. (a) 64 learned weights of the OWA-pooling for the global pooling layer using VGG model and 15-Scenes dataset. (b) 64 learned weights of the OWA-pooling for the global pooling layerfor every chanel (OWAC) using MobileNet model with CUB200 dataset.

he highest values. This result verifies the conclusions obtained by Boureau et al. [7].

5. Conclusions

In this paper we propose a pooling method based on ordered weighted averaging operators named OWA-pooling. We have studied empirically the performance of OWA-pooling compared to common aggregation methods applied to image classification problems for BoW method and for CNNs, in four different datasets. In all of the experiments, the OWA-pooling learned obtains an accuracy equal or better than the maximum or average pooling. We can determine that OWA-pooling is a self-configurable aggregation, and can be used to avoid the choice of a pooling function in the designing process of a recognition architecture.

CRediT authorship contribution statement

J.I. Forcen: Investigation, Software, Methodology, Validation, Writing - original draft. **Miguel Pagola:** Investigation, Conceptualization, Formal analysis, Writing - original draft. **Edurne Barrenechea:** Supervision, Writing - review & editing. **Humberto Bustince:** Supervision, Writing - review & editing, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

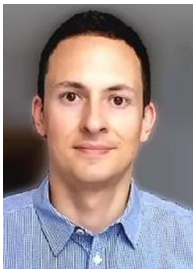
Acknowledgment

This work is partially supported by the research services of Universidad Pública de Navarra and by the project TIN2016-77356-P (AEI/FEDER, UE). We also would like to thank Nvidia for providing equipment.

References

[1] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324, https://doi.org/10.1109/5.726791.
[2] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, Proc. IEEE Computer Soc. Conf. Computer Vision Pattern Recognition 2 (2006) 2169–2178. arXiv:9411012, doi:10.1109/CVPR.2006.68. .
[3] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, Curran Associates Inc, USA, 2012, pp. 1097–1105.
[4] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR abs/1409.1556. arXiv:1409.1556. http://arxiv.org/abs/1409.1556. .
[5] M. Lin, Q. Chen, S. Yan, Network in network, CoRR abs/1312.4400. arXiv:1312.4400. http://arxiv.org/abs/1312.4400. .
[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1–9. doi:10.1109/CVPR.2015.7298594. .
[7] Y.L. Boureau, F. Bach, Y. LeCun, J. Ponce, Learning mid-level features for recognition, Proc. IEEE Computer Soc. Conf. Computer Vision Pattern Recognition (2010) 2559–2566, https://doi.org/10.1109/CVPR.2010.5539963.
[8] Y. Bengio, Practical Recommendations for Gradient-Based Training of Deep Architectures, Springer, Berlin, Heidelberg, Berlin, Heidelberg, 2012, pp. 437–478, https://doi.org/10.1007/978-3-642-35289-8_26.
[9] A. Coates, A. Arbor, A. Y. Ng, An Analysis of Single-Layer Networks in Unsupervised Feature Learning, Aistats 2011 (2011) 215–223 arXiv:fa, doi:10.1109/ICDAR.2011.95. .
[10] P. Koniusz, F. Yan, K. Mikolajczyk, Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection, Comput. Vis. Image Underst. 117 (5) (2013) 479–492, https://doi.org/10.1016/j.cviu.2012.10.010, URL: http://www.sciencedirect.com/science/article/pii/S1077314212001725.
[11] Y.-L. Boureau, J. Ponce, Y. LeCun, A theoretical analysis of feature pooling in visual recognition, Icml (2010) 111–118, citeulike-article-id:8496352. URL: http://www.ece.duke.edu/lcarin/icml2010b.pdf.
[12] M. Pagola, J.I. Forcen, E. Barrenechea, J. Fernández, H. Bustince, A study on the cardinality of ordered average pooling in visual recognition, in: L.A. Alexandre, J. Salvador Sánchez, J.M.F. Rodrigues (Eds.), Pattern Recognition and Image Analysis, Springer International Publishing, Cham, 2017, pp. 437–444.
[13] M. Pagola, J.I. Forcen, E. Barrenechea, C. Lopez-Molina, H. Bustince, Use of owa operators for feature aggregation in image classification, in: 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2017, pp. 1–6, https://doi.org/10.1109/FUZZ-IEEE.2017.8015606.
[14] D. Scherer, A. Müller, S. Behnke, Evaluation of pooling operations in convolutional architectures for object recognition, in: K. Diamantaras, W. Duch, L.S. Iliadis (Eds.), Artificial Neural Networks – ICANN 2010, Springer, Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 92–101.
[15] M. D. Zeiler, R. Fergus, Stochastic Pooling for Regularization of Deep Convolutional Neural Networks (2013) 1–9 arXiv:1301.3557. URL http://arxiv.org/abs/1301.3557 .
[16] Y. Huang, Z. Wu, L. Wang, C. Song, Multiple spatial pooling for visual object recognition, Neurocomputing 129 (2014) 225–231, https://doi.org/10.1016/j.neucom.2013.09.037, URL: http://www.sciencedirect.com/science/article/pii/S0925231213009636.
[17] B. Zhang, Q. Zhao, W. Feng, S. Lyu, Alphamex: A smarter global pooling method for convolutional neural networks, Neurocomputing doi: 10.1016/j.neucom.2018.07.079. URL: http://www.sciencedirect.com/science/article/pii/S0925231218310610. .
[18] Y. Cui, F. Zhou, J. Wang, X. Liu, Y. Lin, S. Belongie, Kernel pooling for convolutional neural networks, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 3049–3058. doi:10.1109/CVPR.2017.325. .
[19] C.-Y. Lee, P. W. Gallagher, Z. Tu, Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree, in: A. Gretton, C. C. Robert (Eds.), Proceedings of the 19th International Conference on Artificial

Intelligence and Statistics, Vol. 51 of Proceedings of Machine Learning Research, PMLR, Cadiz, Spain, 2016, pp. 464–472. URL http://proceedings.mlr.press/v51/lee16a.html .
[20] M. Sun, Z. Song, X. Jiang, J. Pan, Y. Pang, Learning pooling for convolutional neural network, Neurocomputing 224 (2017) 96–104, https://doi.org/10.1016/j.neucom.2016.10.049, URL: http://www.sciencedirect.com/science/article/pii/S0925231216312905.
[21] D.G. Lowe, Distinctive image features from scale invariant keypoints, Int. J. Comput. Vision 60 (2004) 91–110, https://doi.org/10.1023/B:VISI.0000029664.99615.94, http://portal.acm.org/citation.cfm?id=996342.
[22] B. T. N. Dalal, Histograms of Oriented Gradients for Human Detection., IEEE Conference on Computer Vision and Pattern Recognition (CVPR). .
[23] C. Cortes, V. Vapnik, Support-Vector Networks, Mach. Learn. 20 (3) (1995) 273–297, https://doi.org/10.1023/A:1022627411411, arXiv:1011.1669v3.
[24] R. Yager, On ordered weighted averaging aggregation operators in multi criteria decision making, IEEE Trans. Syst. Man Cybern. 18 (1) (1988) 183–190, https://doi.org/10.1109/21.87068, URL: http://dl.acm.org/citation.cfm?id=46931.46950.
[25] G. Beliakov, H. Bustince, T.C. Sánchez, A Practical Guide to Averaging Functions, Springer, 2016.
[26] Jianchao Yang, Yu. Kai, Gong Yihong, T. Huang, Linear spatial pyramid matching using sparse coding for image classification, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1794–1801, https://doi.org/10.1109/CVPR.2009.5206757.
[27] A. Krizhevsky, Learning multiple layers of features from tiny images, Tech. rep. (2009). .
[28] Z. Zhong, L. Zheng, G. Kang, S. Li, Y. Yang, Random erasing data augmentation, in: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2020.
[29] G. Ghiasi, T.-Y. Lin, Q. V. Le, Dropblock: A regularization method for convolutional networks, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems 31, Curran Associates Inc, 2018, pp. 10727–10737. http://papers.nips.cc/paper/8271-dropblock-a-regularization-method-for-convolutional-networks.pdf. .



Juan I. Forcén received the M.Sc. degree in Automation and Electronic engineering in 2009 from the Universidad Politécnica de Madrid. He worked for 8 years in the automatization industry and from 2017 he is involved with the company Das-nano—Veridas in the development of software for document with image processing techniques. He is currently developing his Ph.D. studying different aggregation strategies in deep convolutional networks. He is author 7 international papers (conference procedings and journals) and his research interests include, image retrieval, image classification and Convolutional Neural Networks.

Miguel Pagola received the M.Sc. degree in industrial engineering in 2000 and the Ph.D. degree in 2008 from the Public University of Navarra (UPNa), Pamplona, Spain. He is currently an Associate Lecturer with the Department of Automatics and Computation, UPNa. He is the author or of more than 40 published original articles and is involved in teaching artificial intelligence for students of computer sciences. His research interests include, image processing, Convolutional Neural Networks, feature aggregation, clustering, and applications of Artificial Intelligence in the industry. Dr. Pagola is a Member of the Spanish Society for Artificial Intelligence (AEPIA) and belongs to the Institute of Smart Cities of his University.

Edurne Barrenechea received the MSc degree in Computer Science from University of the Basque Country, San Sebastian, Spain, in 1990 and the PhD degree in Computer Science from the Public University of Navarre (UPNA), Spain, in 2005. The topic of her PhD was interval-valued fuzzy sets applied to image processing. She was with a private company until 2001 when she joined the UPNA. Nowadays, she is Associate Professor at the Department of Statistics, Computer Science and Mathematics. Her publications comprise more than 55 papers in JCR and about 20 book chapters. Her research interests include fuzzy techniques for image processing, fuzzy sets theory, interval type-2 fuzzy sets theory and applications, decision making, and medical and industrial applications of soft computing techniques. She is a Board Member of the Spanish Society for Artificial Intelligence (AEPIA) and belongs to the Institute of Smart Cities of her University.

Humberto Bustince received the BSc degree in physics from the University of Salamanca in 1983 and PhD in mathematics from the Public University of Navarra, Pamplona, Spain, in 1994. He is a Full Professor of Computer Science and Artificial Intelligence in the Public University of Navarra, Pamplona, Spain, where he is the main researcher of the Artificial Intelligence and Approximate Reasoning group, whose main research lines are both theoretical (aggregation functions, information and comparison measures, fuzzy sets, and extensions) and applied (image processing, classification, machine learning, data mining, and big data). He has led more than 10 I + D public-funded research projects, at a national and at a regional level. He has authored more than 210 works, according to Web of Science. He is an Associated Editor of the IEEE Transactions on Fuzzy Systems Journal and a member of the editorial board of the Journals Fuzzy Sets and Systems, Information Fusion, International Journal of Computational Intelligence Systems and Journal of Intelligent & Fuzzy Systems. He is the coauthor of a monography about averaging functions and coeditor of several books.

3. Aggregación de características profundas en recuperación de imágenes basadas en la detección de objetos

J.I. Forcén, M. Pagola, E. Barrenechea, H. Bustince

Aggregation of deep features for image retrieval based on object detection

Estado: **Publicado**

Congreso: ***Iberian Conference on Pattern Recognition and Image Analysis***

Áreas de conocimiento: ***Pattern Recognition*, *Image Analysis***

DOI: https://doi.org/10.1007/978-3-030-31332-6_48

Aggregation of deep features for image retrieval based on object detection

Juan I. Forcén[1], Miguel Pagola[2], Edurne Barrenechea[2], and Humberto Bustince[2,3]

[1] Das-Nano — Veridas, Noain, Spain
[2] Dpto Estadística, Matemáticas e Informática and Institute of Smart Cities Universidad Pública de Navarra, Pamplona, Spain
[3] King Abdullazid University, Jeddah, Saudy Arabia

Abstract. Image retrieval can be tackled using deep features from pre trained Convolutional Neural Networks (CNN). The feature map from the last convolutional layer of a CNN encodes descriptive information from which a discriminative global descriptor can be obtained. However, this global descriptors combine all of the information of the image, giving equal importance to the background and the object of the query. We propose to use an object detection based on saliency models to identify relevant regions in the image and therefore obtain better image descriptors. We extend our proposal to multi-regional image representation and we combine our proposal with other spatial weighting measures. The descriptors derived from the salient regions improve the performance in three well known image retrieval datasets as we show in the experiments.

Keywords: Image retrieval · Feature aggregation · Saliency.

1 Introduction

Visual image search has been evolved from variants of the bag-of-words model [10] based on local features, typically SIFT [11], to approaches based on deep neural network features [21]. The first contributions in image retrieval using deep features from Razavian et al. [16] and Babenko et al. [20] proposed different aggregation strategies for deep features and demonstrated state-of-the-art performance in popular benchmarks. Based on these results, recent contributions have been made in order to improve the quality of the final image representation. Representations for image retrieval need to be compact i.e., representations around a few hundred dimensions, while retaining most of the details of the images.

Some approaches like [2] have tried to fine-tune the CNN with training datasets related to test datasets. Others methods as, [19] or [6], have focused in the methodology of feature extraction from the layers of the network. Fine-tuning approaches improve results in particular datasets, but have drawbacks of requiring training datasets with expensive annotations depending on a category of the test dataset.

We place this work as pre-trained single-pass category of CNN-based approaches, defined by Zheng et al. [21]. We encode images into compact feature vectors using an off-the-shelf CNN, commonly known as a general feature extractor. Kalantidis et. al. [6] established a straightforward way of creating powerful image representations by means of multidimensional aggregation and weighting. In Figure 1 is depicted the basic process of this methodology. An image is processed by a CNN, from which we get the tensor X, i.e. the activation maps of the last convolution layer. These convolutional features can be weighted channel-wise by a vector β and spatially by a matrix α to obtain a weighted feature matrix X'. These weighted features X' are pooled to derive a final feature vector, i.e. the image representation.

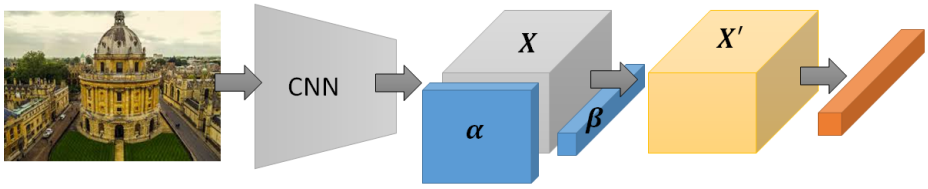


Fig. 1. Methodology to obtain image representations by multidimensional weighting and aggregation.

In this work we follow such methodology to obtain compact image representations. Our objective is to propose a spatial weight matrix α such a way we focus on the relevant object of the image and therefore remove non discriminative information providing better results in image retrieval applications. Basically, we propose to use object detection based saliency measure [22] to identify relevant regions in the image. The contributions of this paper can be summarized as follows:

- We propose a weighting approach based on object detection, over off-the-shelf CNN convolutional features for image retrieval.
- We extend the weighting methodology to multi-regional image representation.
- In the experimental results we demonstrate the effectiveness of our method to well-known image retrieval datasets, and compared it to the state-of-the-art techniques that are in the category of pre-trained single-pass.

This work is divided in the following sections: first, in Section 2, related works with our proposal are recalled. In Section 3 we give a detailed explanation of our proposal followed by the experimental results Section. Finally the conclusions and future work are presented.

2 Related work

CNN-based retrieval methods have been developed in recent years and are replacing the classical local detectors and descriptors. Several CNN models serve as good choices for extracting features, including AlexNet [8], VGGNet [18] or ResNet [8]. Based on the transfer learning principle, the first idea was to extract an image descriptor from a fully-connected layer of the network, however, it has been observed by [19] that the pooling layer after the last convolutional layer (e.g., pool5 in VGGNet), usually yields superior accuracy than the fully-connected descriptors and other convolutional layers. Basically, in [16] and [19], was proposed a feature aggregation pipeline using max-pooling that, in combination with normalization and whitening obtained state-of-the-art results for low dimensional image codes. Following these results, research efforts have been focused on the aggregation of the features from the pre-trained CNNs. Taking into the account Figure 1, the research within the pre-trained single-pass methods, is devoted to study the aggregation procedure to get the final vector. This means, to identify a proper channel weighting vector β, a proper spatial weighting matrix α and select the correct aggregation function, for example, maximum or sum, to obtain a low dimensional image representation. In Babenko et. al. [20] is used a global sum pooling with a centering priority. Kalantidis et. al. [6] proposed a non-parametric spatial weighting method focusing on activation regions and a channel weighting related to activation sparsity with global sum pooling. Similarly, Cao et al. [1] proposed a method to derive a set of base regions directly from the activations of the convolutional layer. An extension to global pooling is the Regional-Maximum Activation of Convolutions (R-MAC) [20] where the image is divided into overlapped square regions of specific scales. R-MAC performs a max pooling for all regional feature maps and a standard post-processing such as l2-normalization and PCA-whitening. The global feature vector is obtained by sum pooling of the region vectors, followed by l2-normalization. Jimenez et al. [5] studied the Class Activation Maps (CAMs) to improve over the fix region strategy of R-MAC.

The notion of saliency in CNN-based image retrieval works has referred to the most active regions of the image. Two different types of saliency measures have been used in the literature. The first saliency measures are calculated from the same CNN from which image representations are extracted. For example, Lanskar and Kannala [9] used a saliency measure directly derived from the convolutional features to weight the contribution of the regions of R-MAC prior to aggregation. Their proposed saliency measure consists in sum aggregating the feature maps over the channel dimension. Similarly, the work of Simeoni et al. [17] propose using a saliency measure based on the k nearest neighbours regions of an image dataset. Other works proposed saliency models trained by an auxiliary network [7].

Other type of saliency measures are human-based saliency models, such as those derived from the popular Itti and Koch model [3]. In [12], different human-based saliency measures are compared when the final feature vector is created by means of bags of local convolutional features (BLCF).

However, all these methods do not take into account that within the query image the most important region of the image (what we are looking for) usually is the main object of the image. Saliency measures based of activations should prioritize important regions for classification, while human based measures focus on important details for eye-recognition. But, in order to improve the final accuracy in the image search problem, all of the area of the query object should be taken into account. In fact, when an object detection search is done, the accuracy increases a lot [19]. Therefore, the spatial weighting scheme should consider all the object. This is why we propose to use saliency measures that splits the image into object and background, such a way that our aggregation scheme could aggregate all of the important features of the query.

3 Image representation

In this section, we describe our proposal to obtain image representations. Our approach follows the multidimensional aggregation and weighting scheme proposed in [6]. Furthermore, we extend this methodology and adapt it when we use a multi scale partition of the image like R-MAC.

This aggregation scheme, as explained in the Section 1 and depicted in Fig. 1, can be applied to any deep CNN. After extracting deep convolutional features from the last convolutional layer of a CNN, activation values are weighted both spatially and per channel before aggregate them to create a final vector. It is important to take into account that the convolutional layer can be of arbitrary size, and therefore avoid resizing and cropping the input image, allowing images of different aspect ratios to keep their spatial features intact.

Being $X \in \mathcal{R}^{M \times N \times D}$, the tensor of activations from a convolutional layer, these are the steps to generate an image representation vector:

1. **Compute spatial weighting factors.** For each location (i, j) in the feature maps calculate a weight, $\alpha_{i,j}$, that will be applied to each channel at that location.
2. **Compute channel weighting factors.** For each channel k, calculate a weight, β_k that will be applied to every location in that channel.
3. **Compute the weighted tensor of activations.** Transform the original tensor of activations X into a new weighted tensor X':

$$X'_{i,j,k} = \alpha_{i,j}\beta_k X_{i,j,k} \tag{1}$$

4. **Perform aggregation.** Aggregate the new X' tensor of activations by channel to obtain a single vector of dimension $1 \times D$.
5. **Perform vector normalization.** The resulting vector is first normalized (l2-normalization). Then, PCA is applied to reduce the dimensionality and increment the discriminative power followed by a second normalization (l2-normalization).

3.1 Image representation from multi scale region partition

In this section we extend the process of computing an image representation from a multi scale partition of a convolutional layer. The process of multi scale region generation of R-MAC [19] is as follows: with a convolutional activation tensor X, sample square regions, R, of specific scales in a sliding window manner with 40% overlap between neighbour windows, for all scales $s = 1, ..., S$. The region size at a specific scale can be calculated as: $R_s = 2min(M, N)/(s+1)$, where M and N are width and height of each channel of X, respectively. To obtain a final vector representation, basically there is a new step 0 to build the regions and steps 3 and 4 of the process described previously are substituted by the following ones:

0. **Generate regions** Rs**.** Generate regions for different scales, usually $s = 1, 2, 3$. For example, a region $R_p = X_{i=x1:x2,j=y1:y2,k=1:D}$ is the following activation tensor:

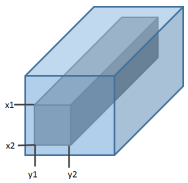


Fig. 2. Region $R_p = X_{i=x1:x2,j=y1:y2,k=1:D}$ from an activation tensor X.

3b. **Compute the weighted tensor of activations.** Transform the original tensor of activations X into a new weighted tensor X':

   a. **Compute the weighted tensor of a region.** Obtain a new weighted tensor Rs' for every Rs of the original activations tensor X:

$$Rs'_{i,j,k} = \alpha_{i,j}\beta_k Rs_{i,j,k} \tag{2}$$

   b. **Perform region aggregation.** Aggregate the new Rs' tensor of region activations by channel to obtain a single vector of dimension $1 \times D$.

4b. **Perform global aggregation.** Aggregate all of the regional vectors of dimension $1 \times D$ into a single one.

Nonetheless, we would like to remark that R-MAC uniformly treats all regions of an image, even though only specific regions would be helpful to construct a discriminative global feature. We aim to address this issue, by weighting the regions by the spatial factor $\alpha_{i,j}$, and studying different global aggregations as the maximum or the average mean.

3.2 Saliency and object detection

It is well known that the accuracy of image search is maximized when object localization of the query Q is done over the original image $\mathbf{I}$. If the similarity is obtained from the region R of maximum similarity, then the corresponding similarity does not take into account all the visual content of image $\mathbf{I}$ and is therefore free from the influence of background clutter. However, the brute-force detection of the optimal region by exhaustive search is really expensive, as the number of possible regions is in $O(M^2N^2)$. Therefore, it is really interesting to obtain a spatial weight $\alpha_{i,j}$, which is able to discard background information.

Taking this consideration into account we propose to use a object detection oriented saliency measure [22], that is able to detect the complete area of the object and discard the background. Such measure, characterizes the spatial layout of image regions with respect to image boundaries and can distinguish between the object and the background at a high precision. The saliency measure is calculated in the original image, and then resized to $M \times N$ (the height and width of the tensor X). We finally apply a gaussian blurring to the resized saliency map to obtain a saliency map $\alpha_{i,j}$ as shown in Fig. 3.

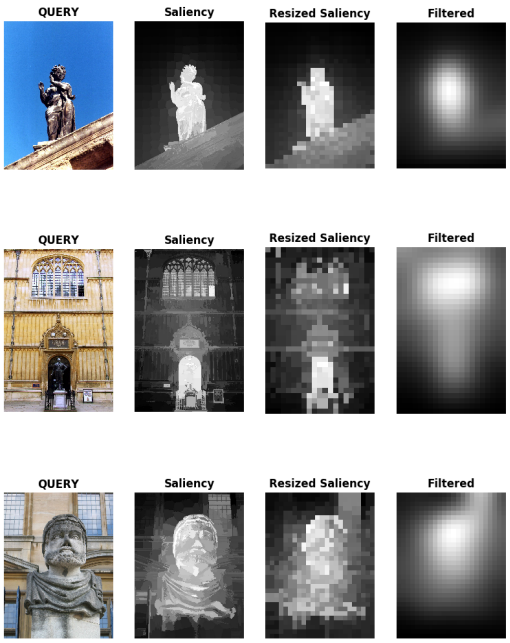


Fig. 3.

4 Experimental results

In this section we present the results obtained in three common image retrieval datasets, Oxford [14], Paris [15], and Holidays dataset [4]. The Oxford Buildings dataset consists of 5062 images with 11 different landmarks, each represented by 5 possible queries, Paris dataset is similar, created with 6412 Paris images, and Holidays dataset with 1491 images and 500 queries.

For each image, the tensor of activations X is the last pooling layer "pool5" obtained from a VGG16 pretrained network in Imagenet dataset (similar to [19], [6], [5] and [13]). The tensor X is of 512 channels, and proportional to the input image size, because we pass input images through the network with their original size. The tensor is processed following the scheme presented in section 3 resulting in a 512 dimensions single vector per image. In Oxford experiments, Paris dataset is used for PCA purposes, whilst in Paris and Holidays dataset Oxford dataset is used.

For each image query, all of the images of the dataset are sorted according to their euclidean distance. Also a simple query expansion (QE) method is applied in Oxford and Paris dataset.

To evaluate the performance in the given queries of these datasets we use the mean average precision metric (mAP), because it is the standard procedure used in the literature.

4.1 Experiment 1: Global pooling aggregation

In this first experiment we evaluate different spatial weighting factors $\alpha_{i,j}$, including: Uniform weights (U), activations based weights [6] (A), our proposal based on saliency measure (SAL) and a Top-Down weighting matrix, in which pixels of the top rows have higher weight than pixels in the botton (TD). Also we evaluate two different channel weighting vectors: Uniform weights per channel (U) and weights based on the sparsity of the map (Sp) [6]. We also test different combination of weighting schemes by directly multiplying their weights.

In Table 1 is presented the performance of different aggregation schemes in the three proposed datasets. We can appreciate that the saliency method (SAL) [22] used as spatial weighting scheme is very helpful to obtain discriminative objects in the images. Moreover saliency can be combined with activations based spatial weights [6] that leads to a great performance improvement. Also we find that a simple weighting scheme (TD) based in the idea that images are taken in vertical position gives a boost when it is combined with any other spatial weight. In fact, the best performance is obtained when the three weighting schemes are joined.

Finally our results are in general better than [13] based also on saliency to weight features, although their approach is not directly comparable because they used a bag of features scheme.

In Fig. 4 and Fig. 5 are depicted the performance in all of the Oxford dataset queries and some queries examples with their similar images respectively.

Method			Oxford		Paris		Holidays
	$\alpha_{i,j}$	β_k	mAp	mAp(QE)	mAp	mAp(QE)	mAp
ucrow	U	U	0.659	0.705	0.757	0.827	0.811
crow [6]	A	S	0.672	0.715	0.787	0.855	0.825
Top-down	TD	U	0.702	0.756	0.781	0.847	0.789
Act-Top-down-Sp	TD*A	S	0.712	0.771	0.807	0.871	0.819
SAL	SAL	U	0.669	0.718	0.776	0.842	0.804
SAL-Sp	SAL	Sp	0.688	0.746	0.794	0.855	0.818
Act-SAL-Sp	A*SAL	Sp	0.688	0.747	0.801	0.861	0.824
SAL-Top-down	SAL*TD	U	0.710	0.778	0.786	0.853	0.799
Act-SAL-Top-down-Sp	Act*SAL*TD	Sp	0.727	**0.791**	**0.813**	**0.873**	**0.830**
BLCF-SalGAN[13]	SalGAN	U	**0.746**	0.778	0.812	0.830	—

Table 1. Comparison of different global pooling aggregations in Oxford, Paris and Holidays datasets.

4.2 Experiment 2: Region pooling aggregation

In this second experiment, we compare the performance of the spatial weighting methods when they are used in a multi region pooling scheme (subsection 3.1). Regions are extracted in three different scales, this results in about 20 regions per image (depending on the size of the image). Each region is transformed in a representation vector and then those vectors are combined. We test two different global aggregations, maximum and average mean to combine the region vectors.

The results obtained are shown in Table 2. We can observe that using the multi scale representation in this aggregation scheme do not improve the performance in all of the cases studied. Furthermore it seems that depends on the dataset and the partial weights used. Also we have not a clear conclusion about using the maximum or the average mean as the global aggregation. However the best performance is obtained with the spatial weight as the combination of the proposed saliency, the activations based and the top down mask when the region vectors are combined with the maximum.

Method		Oxford		Paris		Holidays
	Aggr.	mAp	mAp(QE)	mAp	mAp(QE)	mAp
RMAC[4]	Avg	0.646	0.747	0.763	0.828	0.803
RA-ucrow	Avg	0.660	0.717	0.779	0.843	0.818
RA-crow [6]	Avg	0.674	0.730	0.805	0.866	**0.841**
RA-SAL	Avg	0.650	0.710	0.775	0.839	0.794
RA-SAL-Top-down	Avg	0.710	0.778	0.785	0.848	0.789
RA-Act-SAL-Sp	Avg	0.662	0.730	0.798	0.856	0.816
RA-Act-SAL-Top-down-Sp	Avg	0.697	0.785	0.811	0.867	0.826
RM-crow [6]	Max	0.666	0.724	0.789	0.857	0.825
RM-SAL	Max	0.665	0.733	0.775	0.838	0.799
RM-SAL-Top-down	Max	0.710	0.778	0.783	0.851	0.794
RM-Act-SAL-Sp	Max	0.682	0.756	0.800	0.858	0.818
RM-Act-SAL-Top-down-Sp	Max	**0.724**	**0.799**	**0.813**	**0.871**	0.821
RMAC [19]	Avg	0.661	—	0.83	—	—

Table 2. Comparison of different regional pooling aggregations in Oxford, Paris and Holidays datasets.

Fig. 4. Performance comparison for Oxford dataset between Act-SAL-Top-down-Sp and ucrow aggregation schemes. In the title of each image is shown the performance obtained with uniform aggregation (ucrow) and the current performance of the Act-SAL-Top-down-Sp method. The square in the image represents the query region.

4.3 State-of-the-art-results

Finally in Table 3, we compare our proposal with state-of-the-art results using off-the-self models. We achieve the state-of-the-art performance in similar experiments with crow [6], BLCF-SalGAN [13], RMAC [19] or CAM [5] or [16]. Another conclusion is that our proposed method is fitted to the Query Expansion process, such a way we obtain higher performance with QE than others methods which are better than ours without QE.

Method		Oxford		Paris		Holidays
	Dimensions	mAp	mAp(QE)	mAp	mAp(QE)	mAp
Act-SAL-Top-down-Sp	512	0.727	0.791	0.813	**0.873**	**0.830**
RM-Act-SAL-Top-down-Sp	512	0.724	**0.799**	0.813	0.871	0.820
crow [6]	512	0.672	0.715	0.787	0.855	0.826
BLCF-SalGAN [13]	336	**0.746**	0.778	0.812	0.830	—
RMAC [19]	512	0.660	—	0.830	—	—
CAM [5]	512	0.736	0.760	0.855	0.873	—
SPoC [20]	256	0.531	—	—	—	—
Razavian [16]	32k	**0.843**	—	**0.853**	—	—

Table 3. State-of-the-art results.

Fig. 5. Queries examples, In first row of each example is shown the nearest images to the query obtained with Act-SAL-Top-down-Sp, also in the title o each image is shown its previous rank position with ucrow aggregation. In the second row is shown the spatial weight as the product of saliency and the top-down weight.

5 Conclusions

In this work we have presented a method to weight most important image regions in the process of obtaining a global descriptor for image retrieval. It is based on robust methodology that is able to split the image into object and background. We have showed that this spatial weighting scheme combined with an activation based measure and a simple top-down filter achieves higher performance (specially in Oxford and Paris) than other pre-trained single-pass methods. However, some future research should be done with respect to the aggregation method followed with a multi regional image representation.

ACKNOWLEDGMENT

This work is partially supported by the research services of Universidad Pública de Navarra and by the project TIN2016-77356-P (AEI/FEDER, UE).

References

1. Cao, J., Liu, L., Wang, P., Huang, Z., Shen, C., Shen, H.T.: Where to Focus: Query Adaptive Matching for Instance Retrieval Using Convolutional Feature Maps pp. 1–10 (2016), http://arxiv.org/abs/1606.06811
2. Gordo, A., Almazán, J., Revaud, J., Larlus, D.: End-to-End Learning of Deep Visual Representations for Image Retrieval. International Journal of Computer Vision (2017). https://doi.org/10.1007/s11263-017-1016-8
3. Itti, L., Koch, C.: A saliency-based search mechanism for overt and covert shifts of visual attention. Vision Research **40**, 1489–1506 (2000)
4. Jegou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: Proceedings of the 10th European Conference on Computer Vision: Part I. pp. 304–317. ECCV '08, Springer-Verlag, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88682-224, http://dx.doi.org/10.1007/978-3-540-88682-224
5. Jimenez, A., Alvarez, J.M., Giro-i Nieto, X.: Class-Weighted Convolutional Features for Visual Instance Search. In: In 28th British Machine Vision Conference (BMVC) (jul 2017), https://arxiv.org/abs/1707.02581
6. Kalantidis, Y., Mellina, C., Osindero, S.: Cross-dimensional weighting for aggregated deep convolutional features. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **9913 LNCS**, 685–701 (2016). https://doi.org/10.1007/978-3-319-46604-048
7. Kim, J.: Regional Attention Based Deep Feature for Image Retrieval. Bmvc pp. 1–13 (2018)
8. Krizhevsky, A., Sutskever, I., Hinton., G.E.: AlexNet. Advances in neural information processing systems (2012). https://doi.org/10.1016/B978-008046518-0.00119-7
9. Laskar, Z., Kannala, J.: Context aware query image representation for particular object retrieval. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **10270 LNCS**, 88–99 (2017). https://doi.org/10.1007/978-3-319-59129-28

10. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition **2**, 2169–2178 (2006). https://doi.org/10.1109/CVPR.2006.68
11. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the Seventh IEEE International Conference on Computer Vision. vol. 2, pp. 1150–1157 vol.2 (Sep 1999). https://doi.org/10.1109/ICCV.1999.790410
12. Mohedano, E., McGuinness, K., Giro-I-Nieto, X., O'Connor, N.E.: Saliency weighted convolutional features for instance search. In: Proceedings - International Workshop on Content-Based Multimedia Indexing. vol. 2018-Septe (2018). https://doi.org/10.1109/CBMI.2018.8516500
13. Mohedano, E., McGuinness, K., Giro-i Nieto, X., O'Connor, N.E.: Saliency Weighted Convolutional Features for Instance Search (nov 2017), http://arxiv.org/abs/1711.10795
14. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2007)
15. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2008)
16. Razavian A. S., Azizpour H., S.J., S., C.: CNN features off-the-shelf: an astounding baseline for recognition. Proceedings of the IEEE International Conference on Computer Vision (2015)
17. Simeoni, O., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Unsupervised object discovery for instance recognition. Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018 **2018-January**, 1745–1754 (2018). https://doi.org/10.1109/WACV.2018.00194
18. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition (sep 2014), http://arxiv.org/abs/1409.1556
19. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of CNN activations. ICL 2016 - International Conference on Learning Representations, May 2016, San Juan, Puerto Rico. pp. 1–12 (2015), http://arxiv.org/abs/1511.05879
20. Yandex, A.B., Lempitsky, V.: Aggregating local deep features for image retrieval. Proceedings of the IEEE International Conference on Computer Vision pp. 1269–1277 (2015). https://doi.org/10.1109/ICCV.2015.150
21. Zheng, L., Yang, Y., Tian, Q.: SIFT Meets CNN: A Decade Survey of Instance Retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(5), 1224–1244 (2018). https://doi.org/10.1109/TPAMI.2017.2709749
22. Zhu, W., Liang, S., Wei, Y., Sun, J.: Saliency optimization from robust background detection. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition pp. 2814–2821 (2014). https://doi.org/10.1109/CVPR.2014.360

4. Co-ocurrencia de características de redes neuronales convolucionales aplicadas a recuperación de imágenes

J.I. Forcén, M. Pagola, E. Barrenechea, H. Bustince

Co-occurrence of deep convolutional features for image search

Estado: ***Publicado***

Revista: ***Image and Vision Computing***

DOI: `https://doi.org/10.1016/j.imavis.2020.103909`

Índice de impacto: ***3.103 - Q1 JCR.***

Áreas de conocimiento: ***Computer Science, Theory*** & ***Methods***

Image and Vision Computing 97 (2020) 103909

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Co-occurrence of deep convolutional features for image search

J.I. Forcen [a,b,*], Miguel Pagola [b], Edurne Barrenechea [b], Humberto Bustince [b,c]

[a] das-Nano — Veridas, 31192 Tajonar, Spain
[b] Dpt. Estadística, Informática y Matemáticas, Institute of Smart Cities, Universidad Pública de Navarra, Campus Arrosadía, 31006 Pamplona, Spain
[c] King Abdullazih University, Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 7 February 2020
Accepted 19 March 2020
Available online 25 March 2020

Keywords:
Co-occurrence
Image retrieval
Feature aggregation
Pooling

ABSTRACT

Image search can be tackled using deep features from pre-trained Convolutional Neural Networks (CNN). The feature map from the last convolutional layer of a CNN encodes descriptive information from which a discriminative global descriptor can be obtained. We propose a new representation of *co-occurrences* from deep convolutional features to extract additional relevant information from this last convolutional layer. Combining this *co-occurrence* map with the feature map, we achieve an improved image representation. We present two different methods to get the *co-occurrence* representation, the first one based on direct aggregation of activations, and the second one, based on a trainable *co-occurrence* representation. The image descriptors derived from our methodology improve the performance in very well-known image retrieval datasets as we prove in the experiments.

1. Introduction

Visual image search has rapidly evolved from variants of the bag-of-words model [1] based on local features, typically SIFT [2], to approaches focused on deep Convolutional Neural Network (CNN) features [3]. The first contributions in image retrieval using deep features were proposed by Razavian et al. [4] and Babenko et al. [5]. Basically, they established different aggregation strategies for deep features and demonstrated state-of-the-art performance in popular benchmarks. According to these results, taking into account that representations for image retrieval need to be compact, i.e., around a few hundred dimensions, recent contributions have been made, in order to improve the quality of the final image representation. Relevant works as Tolias et al. [6] or Kalantidis et al. basically in Ref. [7], have focused in the methodology of feature extraction from the layers of the network into compact feature vectors using an off-the-shelf CNN, commonly known as a general feature extractor. Kalantidis et al. [7] established a straightforward way of creating powerful image representations by means of multidimensional aggregation and weighting: an image is feed in a CNN, obtaining a tensor A, i.e. the activation maps of the last convolution layer, then these deep convolutional features are aggregated to derive a final feature vector, i.e., the global image representation. Zheng et al. [3] define this category as *pre-trained single-pass* category of CNN-based approaches.

Other approaches have tried to fine-tune the CNN with training datasets related to test datasets [8]. These approaches improve results in particular datasets, but have the drawback of requiring training datasets with expensive annotations depending on a category of each test set.

In computer vision has been widely used the concept of *co-occurrence* matrix to represent textures among other visual features. A *co-occurrence* matrix [9] is defined from an image being the distribution of co-occurring pixel values at a given spatial offset. Recently, this concept of *co-occurrence* has been extended to the *co-occurrence* of features activations in convolutional layers [10]. In this approach, said *co-occurrence* layer calculates the correlation between each pair of feature maps by means of the maximum product of the activations given a set of spatial offsets. This *co-occurrence* representation obtains a 1-dimension vector which contains one correlation value for each possible pair of channels, therefore does not contain spatial information. Our proposed *co-occurrences* representation is a tensor with the same dimension of the original activation tensor, which contains for each location their *co-occurrences* information. In this paper, we applied the *co-occurrence* representation to obtain an improved image representation

* Corresponding author at: das-Nano — Veridas, 31192 Tajonar, Spain.
E-mail addresses: jiforcen@das-nano.com (J.I. Forcen), miguel.pagola@unavarra.es (M. Pagola).

https://doi.org/10.1016/j.imavis.2020.103909
0262-8856/

for image retrieval applications. The contributions of this paper can be summarized as follows:

- We propose a new definition for *co-occurrence* representation of convolutional deep features.
- We introduce a new concept of *co-occurrence* filter. This filter is able to capture the dependencies between channels activations to obtain an improved *co-occurrence* representation.
- We propose a linear and a bilinear pooling approach based on *co-occurrence*, over off-the-shelf CNN convolutional features, for image retrieval.
- We demonstrate in the experimental results the effectiveness of our method to well-known image retrieval datasets, and we compare our method with the state-of-the-art techniques.

The rest of this paper is organized as follows. In Section 2, related work with our proposal is recalled. In Section 3, our new definition of *co-occurrence* and its implementation by means of a *co-occurrence* filter is proposed. Next, in Section 4 are explained two pooling methodologies used to obtain a final image descriptor from our deep *co-occurrences* tensor, followed by Section 5 where a methodology to learn the best *co-occurrence* representation is presented. Finally, in Section 6 our method is compared with other *co-occurrence* representation and with state of the art image retrieval methods. Finally, the conclusions and future work are highlighted in Section 7.

2. Related work

CNN-based retrieval methods have been emerged in recent years and are replacing the classical local detectors and descriptors methods. Several CNN models pre-trained in giant datasets like ImageNet [11], serve as good choices for extracting features, including VGG [12] or ResNet [13]. Based on the transfer learning principle, the first idea was to extract an image descriptor from a fully-connected layer of the network, however, it has been observed that the pooling layer after the last convolutional layer (e.g., pool5 in VGGNet), usually yields superior accuracy than the fully-connected descriptors and other convolutional layers [6]. Basically, [4,6] proposed a feature aggregation pipeline using max-pooling that, in combination with normalization and whitening, obtained state-of-the-art results for low dimensional image codes. Following these results, research efforts have been focused on the aggregation of the features from the pre-trained CNNs. This means, to identify proper spatial regions or weighting functions to obtain a low dimensional image representation. For example, Babenko et al. [5] used a global sum pooling with a center priority, and Kalantidis et al. [7] proposed a non-parametric spatial weighting method focusing on activation regions and a channel weighting related to activation sparsity with global sum pooling. Similarly, Cao et al. [14] proposed a method to derive a set of base regions directly from the activations of the convolutional layer. Jimenez et al. [15] studied the class activation maps for spatial weighting and Mohedano et al. [16] and Simeoni et al. [17] proposed to use different human-based saliency measures for spatial pooling. However, all of these methods do not take into account the correlation between the features of the convolutional layers. This concept of correlation between image features was introduced by Yang and Newsam [18], named as feature *co-occurrence*, characterizes the spatial dependency of the visual features in a given image. Recently, Shih et al. [10] introduce a new *co-occurrence* representation for deep convolutional networks, demonstrating its effectiveness to exploit the information of visual features in the field of visual recognition. However, in the wrong side this method loses spatial information and its execution is very slow.

To overcome these problems, in this paper we consider that the correlation or interdependence between feature maps contains useful information, so in order to improve the final accuracy in the image search problem, we propose to add the *co-occurrence* information to the final image descriptor using a new representation of deep *co-occurrence*.

3. Deep *co-occurrence* tensor of deep convolutional features

In convolutional neural networks when an image is feed in the CNN, the result after the last convolutional layer is an activation map A of size $M \times N$ and D channels, $A \in \mathbb{R}^{M \times N \times D}$. This activations map represents how much is activated each feature (represented in channels) for a given spatial position.

The goal of *co-occurrence* representation is to characterize the spatial dependency of the image features. Yang and Newsam [18] call it *"Spatial co-occurrence Kernel"* and considered it as a count of the times that two visual features satisfy a spatial condition. Shih et al. [10] present a new idea behind the *co-occurrence* representation, recording the spatial correlation c between a pair of feature maps k and w, seeking the maximal correlation response for a set of spatial offsets $o_{ij} = [o_{ij,x}, o_{ij,y}]^T \in \mathbb{R}^2$ i. e.

$$c(k, w) = \max_{o_{ij}} \sum_{p \in [1,m] \times [1,n]} a_p^k a_{p+o_{ij}}^w \tag{1}$$

where a_p^k is the k_{th} channel of A at location p, and a_{p+oij}^w is the w_{th} channel of a at location $p + o_{ij}$.

In this approach, the spatial information is lost in the resultant *co-occurrence* vector c of size D^2, due to the *co-occurrence* between two channels is a single value. Moreover, this high dimensionality makes it unaffordable in deep tensors like VGG with 512 channels. For this reason, Shih et al. [10] add a $1 \times 1 \times N$ convolution filter to reduce the number of channels before the *co-occurrence* layer. As a side effect this channel reduction causes a reduction of performance as demonstrated in Ref. [10]; this representation was also used in Ref. [19].

3.1. Deep-co-occurrence-tensor

In this section, we propose a new method to obtain a *co-occurrence* representation from an activation tensor of a deep convolutional layer. This *co-occurrence* representation is a tensor with equal dimensions than the activation tensor which encodes the correlation between feature maps for each tensor location.

We define that a *co-occurrence* happens when the value of two different activations, $a_{i,j}^k$ and $a_{u,v}^w$, are greater than a threshold, t, and both are spatially located inside of a given region. (graphical interpretation is depicted in Fig. 1).

Given a convolutional map $A \in \mathbb{R}^{M \times N \times D}$, containing a set of activations, given a distance r and a threshold t, we define a positive *co-*

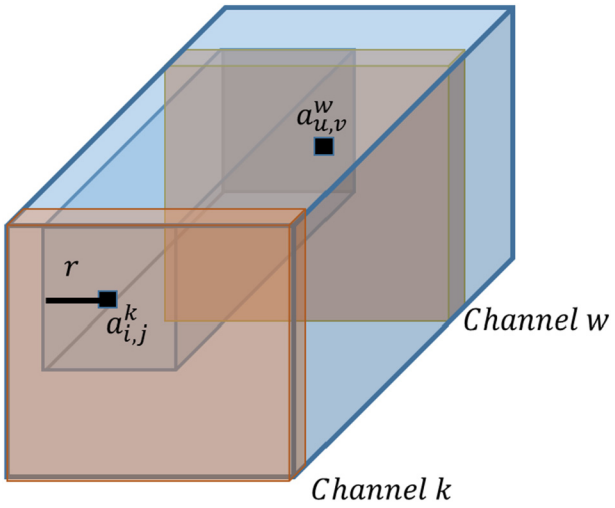


Fig. 1. A *co-occurrence* occurs when the activation of two different activations $a_{i,j}^k$ and $a_{u,v}^w$ are greater than a threshold t and both are spatially located inside of a given region.

occurrence between two activations $a^k_{i,j}$ and $a^w_{u,v}$ as:

$$\rho\left(a^k_{i,j}, a^w_{u,v}\right) = \begin{cases} 1, & \text{if } |i-u| \leq r \text{ and } |j-v| \leq r \text{ and } a^k_{i,j} > t \text{ and } a^w_{u,v} > t \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

The resultant elements of the *co-occurrence* tensor are the sum of all the activations at the positive *co-occurrences* divided by the number of channels. So, the *co-occurrence* tensor is represented as $C_T \in \mathbb{R}^{M \times N \times D}$, calculated by:

$$C_T(i,j,k) = \sum_{u=1}^{M} \sum_{v=1}^{N} \frac{1}{D-1} \sum_{w=1}^{D} \rho\left(a^k_{i,j}, a^w_{u,v}\right) \cdot a^w_{u,v} \quad (3)$$

Remark 1. : The *co-occurrence* of a channel with itself is not considered for the calculation of the *co-occurrence*, for this reason all the activations aggregated are divided by $D-1$.

3.1.1. Co-occurrence and image representativeness

In order to study the representativeness of our proposed *co-occurrence* tensor, we visualize the pair-wise correlation of the query images of Oxford [20] and Paris [21] datasets. We calculate the total *co-occurrence* vector $C_V \in \mathbb{R}^{1 \times D}$ as the sum of all the *co-occurrences* per channel:

$$C_V(k) = \sum_{i=1}^{M} \sum_{j=1}^{N} C_T(i,j,k) \quad (4)$$

We use these vectors C_V of dimension $1 \times D$ to compute the pair-wise correlation between images. The query-sets for both datasets contain 55 images in total, with 5 images of 11 classes of landmarks. We calculate the C_T for each query image with $r=4$ and threshold t as the average mean of all the activations at A (being A the last pooling layer called 'pool5' obtained from a VGG16 pre-trained network in ImageNet dataset).

In Fig. 2, we observe that the *co-occurrence* representation is highly correlated for images of the same landmark and less correlated for images of different landmarks. These figures evidence that the *co-occurrence* tensor contains discriminative information and therefore could be useful to obtain a representative vector applied to image search.

3.1.2. Co-occurrence filter representation

An important advantage of our *co-occurrence* representation is that it can be implemented using convolutional filters. We define the *co-occurrence* filter as a convolutional filter: $F \in \mathbb{R}^{D \times D \times S \times S}$, where D is de number of channels in the activation tensor A, S the window size being $S = 2 \cdot r + 1$, with r the radius that defines the *co-occurrence* region (see Fig. 1).

Note that activations do not compute to their channel *co-occurrence* calculation. So all filters elements are initialized to one except the related with itself channel, that are initialized with zero or a small value ε, for example i.e. $1e-10$.

$$F_{a,b,c,d} \in \mathbb{R}^{D \times D \times S \times S} = \begin{cases} 1, & \text{if } a = b \\ 0. & \text{otherwise} \end{cases} \quad (5)$$

Given an activation tensor, $A \in \mathbb{R}^{M \times N \times D}$, of last convolution operator in a neural network, the *co-occurrence* tensor $C_T \in \mathbb{R}^{M \times N \times D}$ can be obtained as a convolution between a thresholded activation tensor and the *co-occurrence* filter:

$$C_T = \left(A_{\rho_A} * F\right) \cdot \rho_A \quad (6)$$

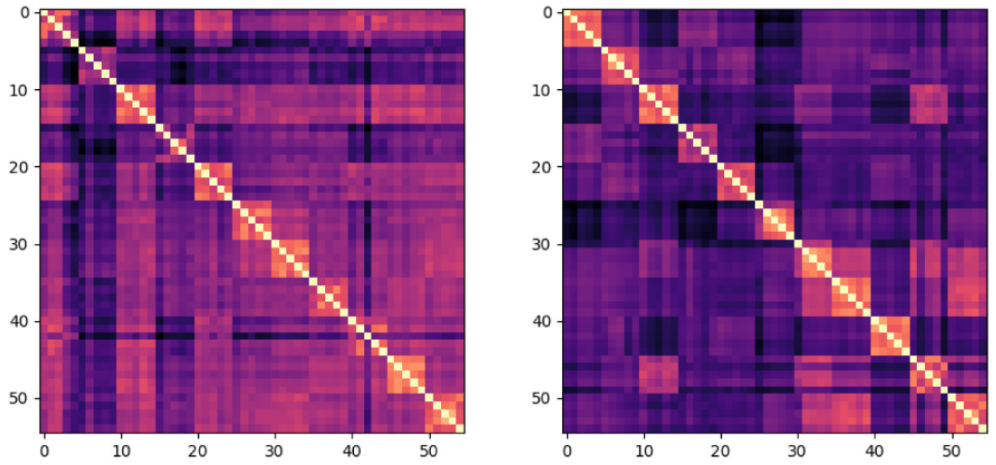
where $A_{\rho_A} = A \cdot \rho_A$ and $\rho_A \in \mathbb{R}^{M \times N \times D}$, $\rho_A = A > \overline{A}$, with $\overline{A}$ the average mean of the activation map, i.e.:

$$\rho_A(i,j,k) = 1, \quad \text{if } a^k_{i,j} > \frac{1}{M \cdot N \cdot D} \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{k=1}^{D} a^k_{i,j} 0. \text{otherwise} \quad (7)$$

The pseudo-code of the implementation is shown in Algorithm 1 and in Fig. 3 is depicted an illustration of the *co-occurrence* tensor calculation.

Algorithm 1. *Co-occurrence* tensor.

```
calcCooc (A, S);
Input  : A : Tensor of activations with shape D × M × N
         S : window size
Output: C_T : Tensor of co-occurrences
filters = ones(D, D, S, S)
For i = 1 to D :
    filters[i, i, :, :] = 1e-10
ρ_A = A > mean(A)
A_ρA = A · ρ_A
C_T = conv2d(A_ρA, filters, padding = r) / (D − 1)
C_T = C_T · ρ_A
return C_T
```



(a) Oxford (b) Paris

Fig. 2. Correlation of *co-occurrence* vector *CV* for the 55 images in the query-set of Oxford (a) and Paris (b) datasets. Images are sorted by landmark.

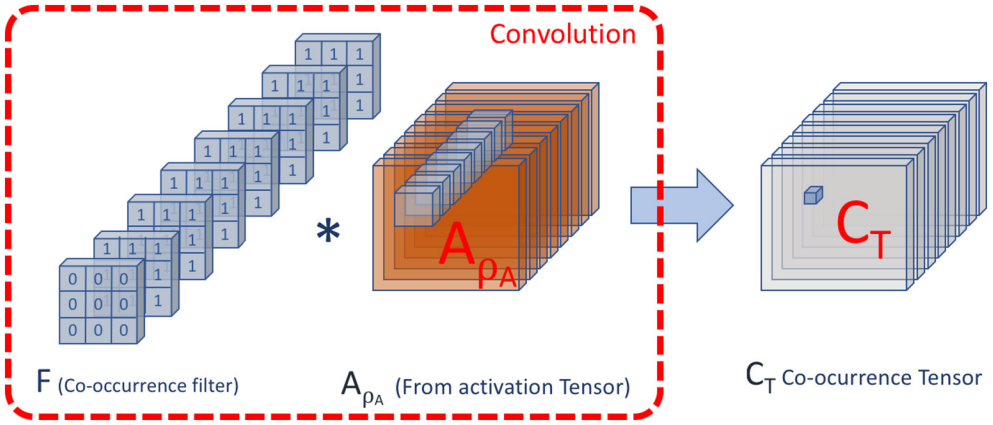


Fig. 3. Example of the *co-occurrences* filter $F_0 \in \mathbb{R}^{D\times S\times S}$, convolved with $A_{\rho A}$ to obtain a value in position (i, j) in the *co-occurrences* tensor (best viewed in color). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

This *co-occurrence* implementation based on convolutions, makes possible to calculate the *co-occurrence* representation with no channel reduction. Moreover, it is simple and straightforward to learn an improved *co-occurrence* representation by adding a *co-occurrence* layer in a trainable architecture.

In the next sections are introduced two different ways to use the *co-occurrence* tensor applied to image retrieval. The first one, called *Direct co-occurrences* uses a straightforward approach to calculate *co-occurrence* (see Algorithm 1) and in the second one called *Learnable co-occurrences* a *co-occurrence* filter is learned to obtain an improved *co-occurrence* representation.

Implementation of the proposed method and some examples, in PyTorch are publicly available: https://github.com/jiforcen/co-occurrence.

4. Direct *co-occurrences*

Off-the-shelf methods of image retrieval obtain a compact representation of the images and queries to perform the query search, typically in three steps: (1) to feed the image in a pre-trained CNN to extract its activation tensor, (2) to apply a pooling function to obtain a compact representation and (3) to apply l2-normalization and pca/whitening to reduce the dimensionality and increment the discriminative power.

In our method, the compact representation, the second step, is obtained by pooling the activation tensor with the *co-occurrence* tensor (see Fig. 4). We have used two different methods, linear and bilinear pooling, to perform this aggregation and demonstrate the effectiveness of the *co-occurrence* representation. Next, these two approaches are explained in detail.

4.1. Linear weighted pooling

Following the lineal weighted pooling methodology proposed by Kalantidis et al. [7], we transform the original tensor of activations into a new weighted tensor:

$$A'_{i,j,k} = \alpha_{i,j}\beta_k A_{i,j,k} \tag{8}$$

where $A \in \mathbb{R}^{M\times N\times D}$ is the tensor of activations from the last convolutional layer, $\alpha_{i,j}$ are the spatial *co-occurrences* obtained from the *co-occurrence* tensor and β_k are channel *co-occurrences*, also obtained from the *co-occurrence* tensor. The final step of lineal pooling is

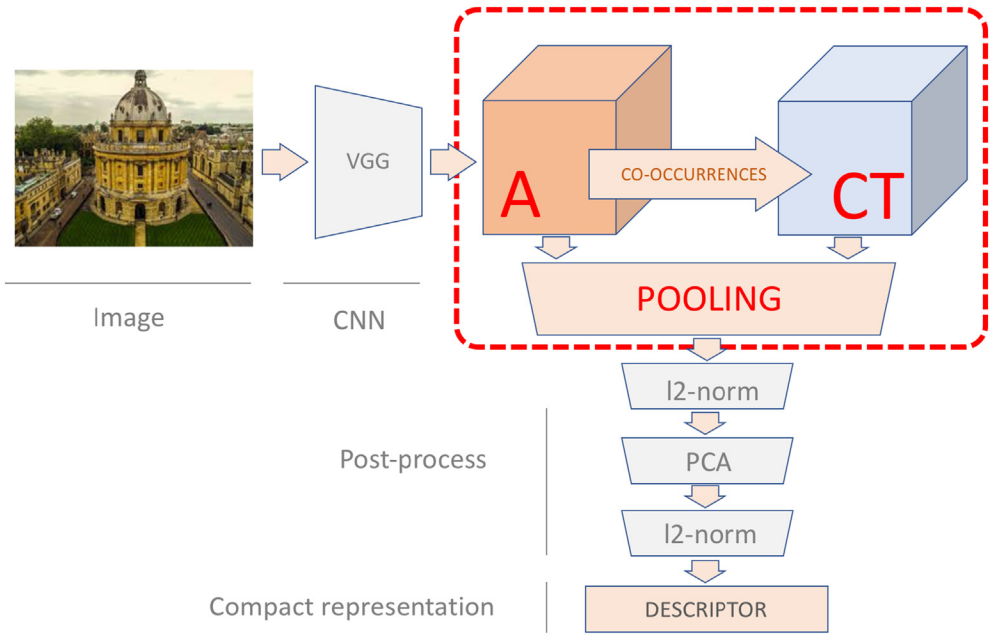


Fig. 4. Aggregation pipeline to obtain the final compact image representation.

the sum the new A' tensor of activations by channel to obtain a single vector of dimension $1 \times D$.

The spatial *co-occurrences* $\alpha_{i,j}$ basically are the normalized total *co-occurrence* across all channels for every image location (i, j), taking into account that spatial locations with large *co-occurrences* across channels should correspond to discriminative locations of the image.

We apply a power normalization to obtain the final spatial *co-occurrences* matrix:

$$\alpha_{i,j} = \left(\frac{S(i,j)}{\left(\sum_{i=0}^{i=M} \sum_{j=0}^{j=N} S(i,j)^a \right)^{1/a}} \right)^{1/b} \tag{9}$$

where $S \in \mathcal{R}^{M \times N}$ is the matrix of aggregated *co-occurrences* from all channels per spatial location:

$$S(i,j) = \sum_{k=1}^{D} CT(i,j,k) \tag{10}$$

Fig. 5 depicts spatial *co-occurrences* $\alpha_{i,j}$ for several images of the Oxford dataset (*co-occurrence* tensor C_T is obtained by Eq. (3)). We visualize that our spatial *co-occurrences* tends to give large values to locations with salient visual content and reject background information.

To calculate the importance of each channel, the inverse of its *co-occurrence* value is used, such a way we boost the contribution of rare features (in a similar way as term frequency inverse document frequency):

$$\beta_k = \log \left(\frac{\sum_{l=0}^{D} VC(l)}{\varepsilon + VC(k)} \right) \tag{11}$$

where ε is a constant to avoid division by zero.

4.2. Bilinear pooling

Bilinear, or second order pooling, introduced by Tenenbaum and Freeman in Ref. [22], has the ability to capture pairwise correlations between channels of a descriptor. It have been successfully applied for different tasks, like semantic segmentation [23], fine grained visual recognition [24] or face recognition [25]. The bilinear pooling of a single dimension descriptor x can be calculated as the outer product of x an its transpose x^T. The outer product captures pairwise correlations between the feature channels and can model feature interactions. In our approach we combine the tensor of activations with the *co-occurrence* tensor:

$$B(A, C_T) = \sum_{i}^{M} \sum_{j}^{N} A_{ij} \times C_{Ti,j}^{T} \tag{12}$$

where $A_{i,j} \in \mathbb{R}^{1 \times D}$ is the vector of activations and $C_{Ti,j} \in \mathbb{R}^{1 \times D}$ the vector of *co-occurrences* at position (i,j). The main disadvantage of bilinear pooling is the large size of the resultant descriptor $B = \mathbb{R}^{D \times D}$. For example, an activation tensor with $D = 512$ channels will produce a final descriptor with about $512 \times 512 \approx 250K$ values, which is excessive in most cases. To tackle with that problem, Y. Gao et al. [26] proposed compact bilinear pooling which is a kernelized view of bilinear pooling, achieving almost equal results of bilinear pooling with a final vector of 8 K values which means a reduction of two orders of magnitude. We will use the compact bilinear pooling implementation in the experiments to reduce the $D \times D$ bilinear matrix into a $1 \times 8K$ vector.

4.3. Experiments

In this section, we present the results obtained with our proposal of direct *co-occurrences* combined with linear and bilinear pooling. To perform the experiments we use common image retrieval datasets: Oxford

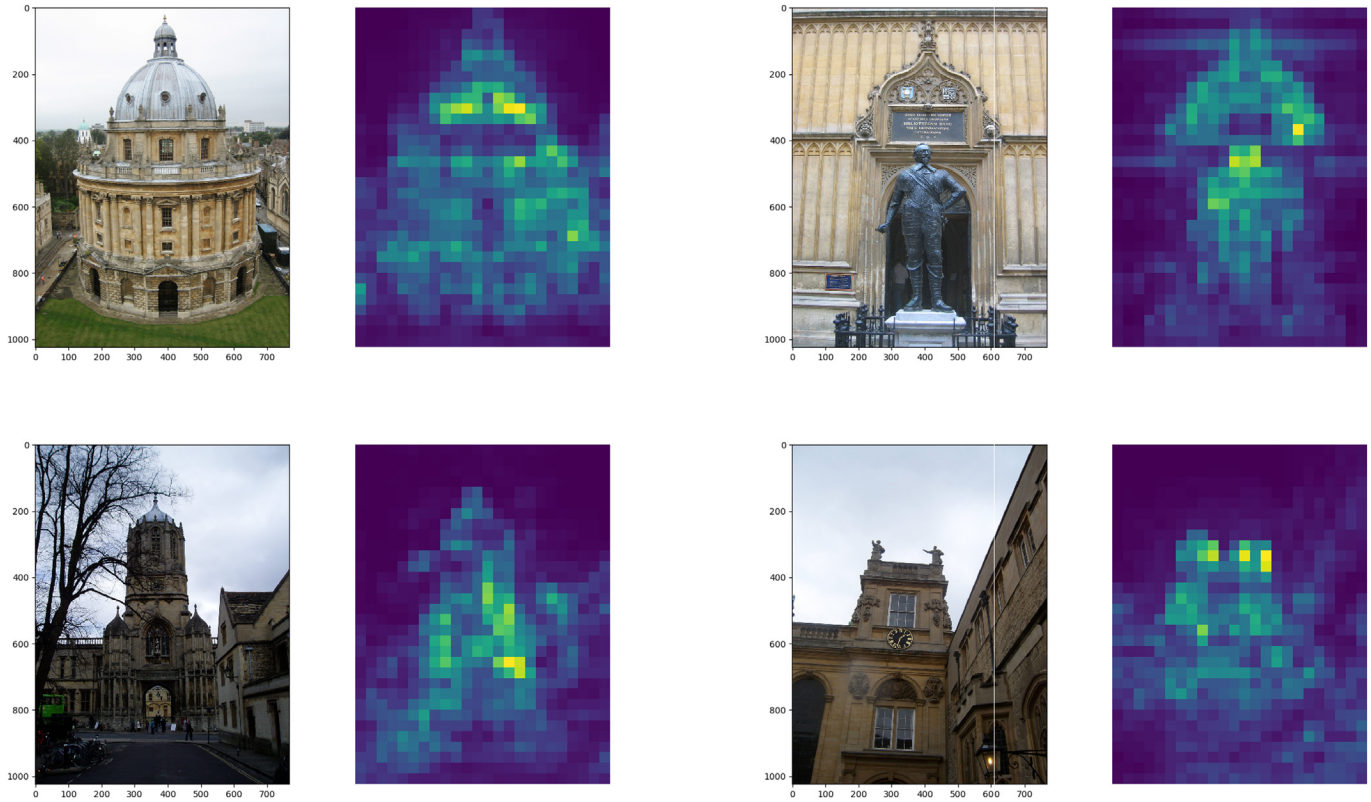


Fig. 5. Spatial *co-occurrences* obtained from Eq. (9) for images of Oxford dataset.

Table 1
Results of linear weighted pooling aggregation based on the *co-occurrence* tensor for the following datasets: Oxford, Paris, Holidays, ROxford and RParis (*co-occurrences* are calculated with r = 4).

Method	Oxford	ROxford			Paris	RParis			Holidays
		Easy	Medium	Hard		Easy	Medium	Hard	
	mAP	mAP	mAP	mAP	mAP	mAP	mAP	mAP	mAP
ucrow	66.0	60.53	41.15	11.98	75.8	74.75	57.69	30.20	81.1
crow [7]	67.2	61.92	44.66	17.94	78.7	76.13	60.17	33.38	82.5
ChCO-SC_T	67.05	61.31	44.52	18.71	79.17	77.16	60.69	33.89	83.22
ChCO-SC_T + *SpTD*	**71.68**	**63.37**	**47.63**	**21.96**	**80.89**	**78.79**	**62.59**	**36.84**	83.04
ChCO-SC_T + *SpCt*	66.82	61.67	43.05	14.91	80.18	77.32	61.05	33.79	**83.96**
ucrow + *AQE*	70.5	55.32	39.94	13.08	82.7	81.32	65.29	38.72	–
crow + *AQE* [7]	71.5	55.75	42.11	17.89	85.5	81.82	67.93	43.16	–
ChCO-SC_T + *AQE*	71.30	56.41	42.46	18.93	85.31	83.59	69.03	44.12	–
ChCO-SC_T + *SpTD* + *AQE*	**77.48**	60.27	**46.58**	**21.19**	**87.13**	85.19	**71.01**	**46.43**	–
ChCO-SC_T + *SpCt* + *AQE*	72.99	**61.83**	44.57	16.79	86.08	**86.04**	70.93	44.93	–

Table 2
Results of bilinear weighted pooling aggregation based on the *co-occurrence* tensor for the following datasets: Oxford, Paris, Holidays, ROxford and RParis (final vector size used is 512).

Method	Oxford	ROxford			Paris	RParis			Holidays
		Easy	Medium	Hard		Easy	Medium	Hard	
	mAP	mAP	mAP	mAP	mAP	mAP	mAP	mAP	mAP
BP(*AA*)	65.94	60.410	45.99	21.66	75.88	73.66	57.80	30.67	80.42
BP(AC_T)	64.23	59.76	44.12	19.75	77.77	77.18	60.09	33.080	**82.62**
BP(AC_T) + *SpTD*	**71.00**	**63.48**	**47.62**	**22.83**	**79.56**	**77.72**	**62.14**	**37.09**	79.51
BP(*AA*) + *AQE*	67.66	48.670	37.76	15.76	81.00	77.71	63.76	37.87	–
BP(AC_T) + *AQE*	66.08	51.24	38.22	15.9	82.97	83.03	67.06	40.85	–
BP(AC_T) + *SpTD* + *AQE*	**77.25**	**56.01**	**43.69**	**19.33**	**84.84**	**83.17**	**68.88**	**44.38**	–

[20], Paris [21], ROxford [27], RParis [27] and Holidays [28]. The Oxford Buildings Dataset consists of 5062 images with 11 different landmarks, each represented by 5 possible queries, Paris dataset is similar, created with 6412 Paris images. ROxford and RParis are a review of Oxford and Paris with three different difficulty levels. The Holidays dataset has 1491 images and 500 queries of holiday photos.

In addition, we have used image retrieval common strategies, like spatial masks and query expansion techniques to reach improved performance results.

4.3.1. Spatial masks

Yandex and Lempitsky [5] supposed that the objects tend to be located close to the geometrical center of an image. Therefore, they incorporate such centering prior with a simple heuristic, which assigns larger weights to the features from the center of the feature map. In the experimental study we evaluate this mask and we also try a new mask configuration, Top-Down, with larger weights to the features in the top of the feature map. This can be useful in datasets like Oxford and Paris which contains discriminant features in the top of the image such as high buildings and less discriminant objects in the bottom like floor, grass, trees, roads or even persons.

4.3.2. Query expansion

Query expansion repeats the search process based on a new query composed by the top ranked results on the first query [7,8,29,30]. We use two different query expansion methods in our experiments. The simplest method, called average query expansion *AQE*, is the average of the *n* nearest descriptors after the first query. The second method, denoted alpha query expansion *αQE*, presented by Radenovic et al. [31], which is a weighted aggregation of the nearest descriptors.

4.3.3. Linear weighted pooling results

For each image, the tensor of activations *A* is the result of the last pooling layer "pool5" obtained from a VGG16 pre-trained network in ImageNet dataset (similar to Refs. [6,7,15,32]). The tensor *A* has 512 channels, and its dimensions (height and width) are proportional to the input image size, due to we pass input images through the network with their original size. The tensor is processed following the scheme presented in Section 4.1 resulting in a 512 dimensions single vector per image. In Oxford experiments, Paris dataset is used for PCA purposes, whilst in Paris and Holidays dataset Oxford dataset is used. For each image query, all of the images of the dataset are sorted according to their Euclidean distance. In addition, a simple query expansion *AQE* method is applied in Oxford and Paris datasets. To evaluate the performance in the given queries of these datasets, we use the mean Average Precision metric (*mAP*), which is the standard procedure used in the literature.

In Table 1, we present the comparison of our *co-occurrence* based linear weighting pooling with other linear weighting methods. Ucrow [7] is the simplest way to aggregate the activation tensor in a compact descriptor, being calculated as the average of $A \in \mathbb{R}^{M\times N\times D}$ over the dimensions *M* and *N*. In addition, we compare with the method proposed in Kalantidis et al. [7] called crow.

These two methods are compared with out method ChCO-SC_T which is a combination of channel weighting *ChCO* and spatial weighting[1] SC_T to perform the linear weighted pooling aggregation of the activation tensor and *co-occurrence* tensor.

Regarding the spatial masks (Section 4.3.1), we have test a Top-Down weighting matrix, in which pixels of the top rows have higher weight than pixels in the bottom (*SpTD*) and a center prior weighting matrix [5], in which pixels of the center have higher value than pixels in the borders (*SpCt*).

The *co-occurrence* tensor C_T was calculated with $r = 4$[2] and the threshold *t* is the average mean of all the activations of the tensor *A*.

Analyzing Table 1, we can appreciate that the *co-occurrence* based pooling is very helpful to obtain representative image vectors (ChCO-

[1] As in Ref. [7], the parameters used in the power normalization were $a = 2$ and $b = 2$.
[2] Previous experiments with $r = 4$, $r = 6$ and $r = 8$ showed us that size influence was low, with the best case $r = 4$.

Table 3
Comparison of bilinear accuracy incrementing the final vector size.

Method	FV size	Oxford	ROxford			Paris	RParis			Holidays
			Easy	Medium	Hard		Easy	Medium	Hard	
		mAP	mAP	mAP	mAP	mAP	mAP	mAP	mAP	mAP
$BP(AC_T)$ + *SpTD*	512	71.00	63.48	47.62	22.83	79.56	77.72	62.14	37.09	79.51
$BP(AC_T)$ + *SpTD*	1024	72.74	65.22	48.83	23.93	82.19	79.88	64.15	39.34	
$BP(AC_T)$ + *SpTD*	2048	73.76	65.98	50.14	26.14	83.08	81.46	65.49	40.76	81.10
$BP(AC_T)$ + *SpTD*	4096	**76.56**	**67.86**	**51.92**	**27.15**	**83.68**	**81.93**	**64.95**	**39.92**	**81.75**
$BP(AC_T)$ + *SpTD* + *AQE*	512	77.25	56.01	43.69	19.33	84.84	83.17	68.88	44.38	–
$BP(AC_T)$ + *SpTD* + *AQE*	1024	77.49	58.67	46.21	21.99	87.45	85.40	71.28	47.24	–
$BP(AC_T)$ + *SpTD* + *AQE*	2048	79.03	59.38	46.68	22.19	88.65	86.94	72.99	49.75	–
$BP(AC_T)$ + *SpTD* + *AQE*	4096	**81.71**	**64.33**	**51.28**	**26.64**	**89.16**	**88.11**	**73.74**	**50.23**	–

SC_T), because weighted vectors improve uniform aggregation (ucrow). Moreover, using ChCO-SC_T our performance is similar in Paris and Oxford and better in Holidays than the crow [7] method, which is a state-of-the-art in pre-trained single pass methodologies. Therefore, the *co-occurrence* tensor captures feature correlations and is able to provide better image representations. We have found that we get a large improvement in our results if the feature tensor A is multiplied by the spatial masks, top-down *SpTD* or center prior *SpCt*. Basically, top-down mask assumes upright images, which is the case of Oxford and Paris datasets due to all of the queries are buildings, and confirms that the spatial structure of the image is relevant in image retrieval.

4.3.4. Bilinear pooling results

In this experiment are evaluated the results obtained using the bilinear pooling method (Section 4.2). All parameters and measures are equal to the previous experiments (Section 4.3.3). In Table 2, we compare the results of the bilinear pooling of activations A and *co-occurrences* C_T ($BP(AC_T)$) with the performance of bilinear pooling with itself A ($BP(AA)$). So, we can evaluate if the *co-occurrence* provides useful information, also combined with the spatial masks. We conclude that the combination of *co-occurrences* and activations is better in almost all the cases than the multiplication of the activations by itself ($BP(AA)$). Therefore, as in the previous experiment, we have proved that the *co-occurrences* reflect the correlations of the features and provide better final image representations. The performance of the bilinear pooling is similar than the previous linear combination for Oxford and Paris datasets but worse in Holidays, and the Top-Down spatial mask influences in a similar way than the previous experiment.

Compact bilinear pooling aggregation of 512 depth vectors result in a vector of 8192 features, in experiments showed in Table 2 we have reduced said vector in the PCA step to 512 features, in order to compare it with linear weighted pooling.

In Table 3 are shown the results with larger final representation size. As we can observe larger vectors produce a great improvement in the *mAP* results, being much better than the linear approach.

5. Learning *co-occurrences*

In this section is explained how a *co-occurrence* filter, F, can be learned within a trainable architecture to obtain a better *co-occurrence*, C_T, representation. We have used a Siamese architecture due to it is known to produce highly discriminative embeddings [33–35].

5.1. Siamese learning

The Siamese architecture is trained with paired samples, maximizing the similarity of positive pairs representation and also maximizing the dissimilarity of negative pairs. In Fig. 6 is presented the Siamese architecture adopted. It contains two equal branches that share parameters; each branch is represented as "CoOc-NET". The output of the "CoOc-NET" is the l2-normalization of the resultant vector after the bilinear pooling operation between the activation tensor and its *co-occurrence* tensor. In our experiments, only the *co-occurrence* filter is trained, and the rest of VGG layers are freeze, so the *co-occurrence* filter weights change to obtain more discriminative representations.

This Siamese architecture is trained in combination with contrastive loss [33] using as train input a pair of images $P_{lm} = [Im_a, Im_b]$ which produces descriptors f_a and f_b. Label $Y(P_{lm})$ will be 1 if the images of the pair correspond to the same class or 0 if not. Following this notation, contrastive loss will be calculated as:

$$d = \|f_a - f_b\|_2 \tag{13}$$

$$Ł(P_{lm}) = Y(P_{lm}) * d^2 + (1 - Y(P_{lm})) * \max(\tau - d, 0)^2 \tag{14}$$

where τ is a parameter to establish a distance margin where dissimilar pairs influence the loss or not.

5.2. Experiments

In this section, we present the results obtained using the trainable *co-occurrence* filter, the experiments done are equivalent to Section 4.3.

5.2.1. Training step

In the training process of Siamese architecture, we have used an image dataset called retrieval-SfM published by Radenovic et al. [31]. This dataset contains 163 k images grouped in 713 clusters of images. We have used the pairing images selection procedure of [31]. Other parameters are: batch size equal to 5 and Adam [36] optimizer with momentum 0.85 and learning rate $1e-9$.

As we have explained previously, only the *co-occurrence* filter is learned, freezing the rest of the network. This *co-occurrence* filter fine-tuning process takes only around 30 epochs to achieve the best result in the validation set. In Fig. 7 are shown four 9×9 filters after the training process. Each filter represents the correlation between a pair of channels.

In Fig. 7, direct and learned *co-occurrence* representations are compared. After the learning process the *co-occurrence* representation shows the ability to put more emphasis in representative regions to discern between queries, for example buildings have higher activation values and a irrelevant objects or persons are dismissed from the *co-occurrence* representation. In Fig. 8(h) and (i), we can see that the *co-occurrence* between features representing humans are avoided, and only the *co-occurrence* of the features representing buildings are taken into account.

5.2.2. Evaluation

In this section is evaluated the *co-occurrence* representation after the *co-occurrence* filter training process in a CoOcNET pipeline. The evaluation is performed similar to [31], with its same whitening procedure, alpha query expansion method *αQE*, and testing also each query in

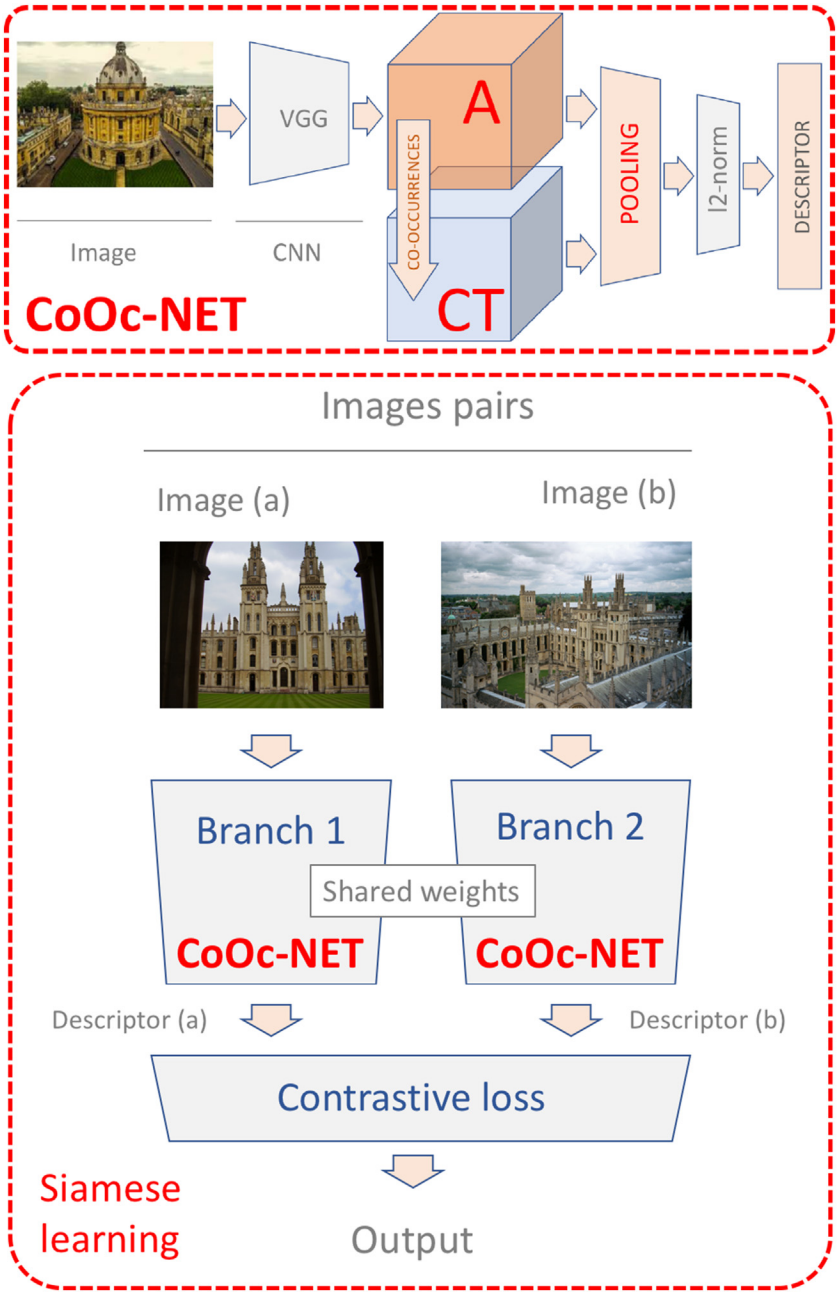
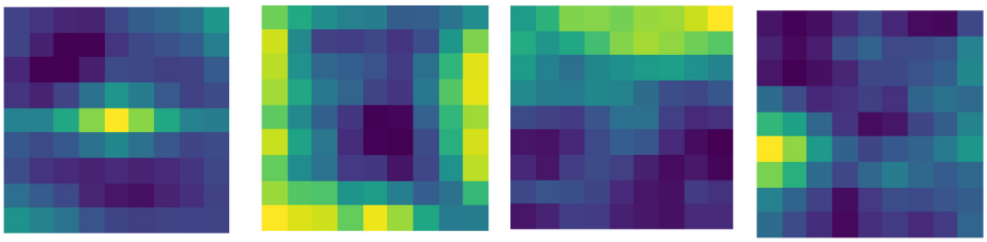


Fig. 6. In the top, the CoOc-NET pipeline used to obtain a compact representation from an image combining the *co-occurrences* tensor. In the bottom, the Siamese architecture based in two equal branches sharing weights and contrastive loss.



(a) 5 - 5 (b) 12 - 8 (c) 5 - 9 (d) 1 - 9

Fig. 7. Filter examples of size 9×9 pixels where each spatial offset is weighted to obtain the best *co-occurrence* representation. Each image represents the filter between a pair of channels (best viewed in color). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

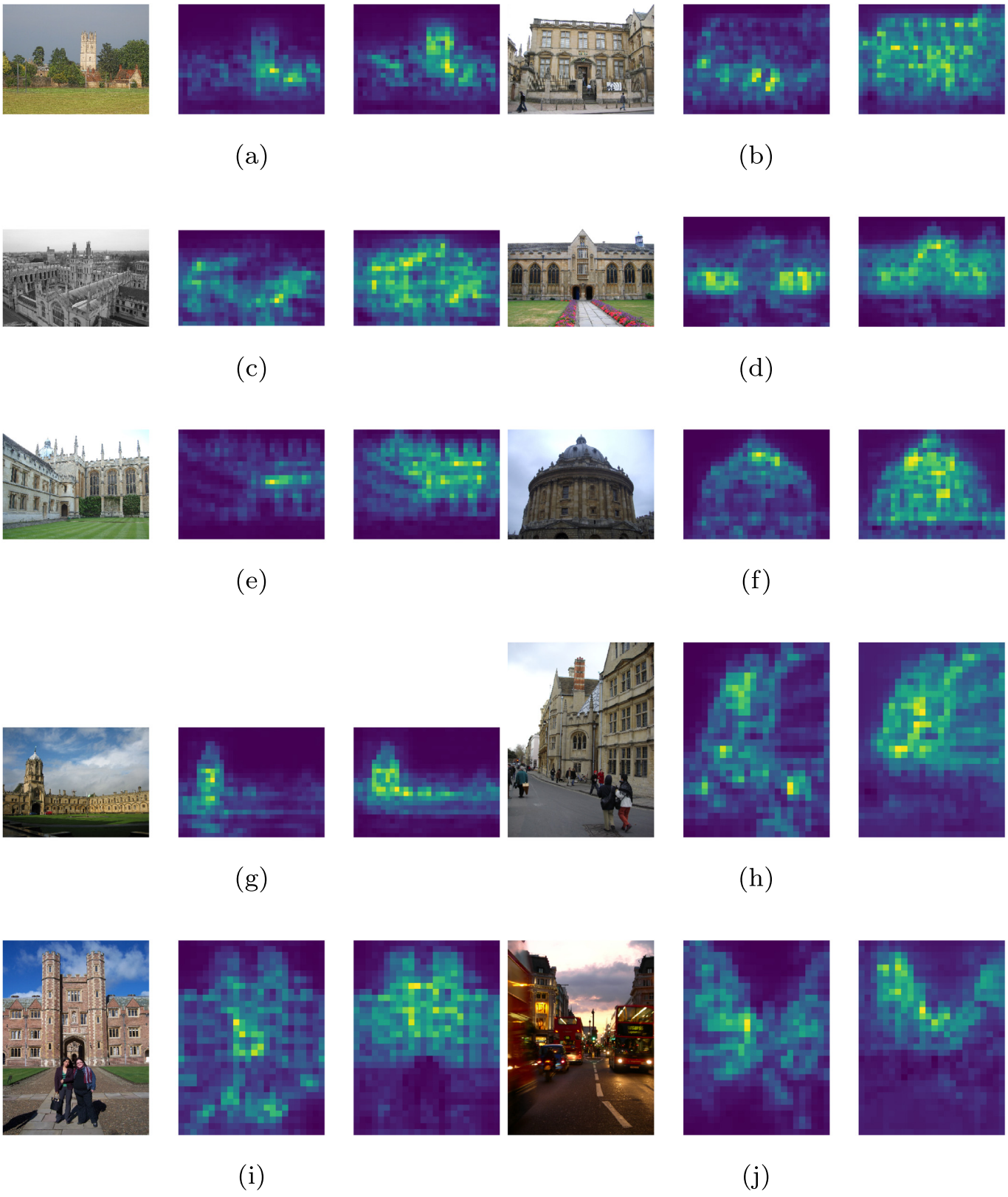


Fig. 8. *Co-occurrence* representation of the *co-occurrence* tensor (sum over channels) for some example images, showing for each one its *co-occurrence* representation with fixed weights (left) and with trainable weights (right), (best viewed in color). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

multiscale, ms, $(1, \sqrt{\frac{1}{2}}, \frac{1}{2})$.

In Table 4 are shown the results comparing bilinear pooling with the itself activation tensor $BP(AA)$, bilinear pooling combining the activation tensor and the *co-occurrence* tensor with fix-weights $BP(AC_T)$ and with the learned weights $BP(AC_T)_{learn}$. It is easy to observe that learning *co-*

Table 4
Results of bilinear weighted pooling aggregation using the pipeline proposed in 5 for learnable *co-occurrences* in the following datasets: Oxford, Paris, Holidays, ROxford and RParis (final vector size used is 8192).

Method	Oxford	ROxford			Paris	RParis		
		Easy	Medium	Hard		Easy	Medium	Hard
	mAP	mAP	mAP	mAP	mAP	mAP	mAP	mAP
$BP(AA)$ + ms	76.67	72.97	55.67	31.24	90.31	89.69	72.22	48.0
$BP(AC_T)$ + ms	76.79	72.53	56.34	32.93	89.55	90.13	71.86	48.22
$BP(AC_T)_{learn}$ + ms	**80.75**	**74.65**	**57.94**	**33.68**	**91.02**	**90.18**	**72.44**	**48.37**
$BP(AA)$ + αQE + ms	80.11	76.08	59.65	33.95	93.42	93.80	80.17	59.36
$BP(AC_T)$ + αQE + ms	81.13	74.53	60.86	38.50	93.23	93.93	80.35	60.44
$BP(AC_T)_{learn}$ + αQE + ms	**85.76**	**82.25**	**67.04**	**42.23**	**94.84**	**94.21**	**80.97**	**61.42**

Table 5
Results comparison between our proposed *co-occurrences* and an adaptation of Shih et al. method.

		ROxford and RParis datasets					
		ROxford			RParis		
		Easy	Medium	Hard	Easy	Medium	Hard
Method	Size	mAP	mAP	mAP	mAP	mAP	mAP
Shih ChCO-SC_T + *SpTD*	512	64.3	47.64	20.86	79.45	62.92	37.15
ChCO-SC_T + *SpTD*	512	64.28	47.02	20.30	79.01	62.18	35.79
ChCO-SC_T + *SpCt*	512	62.06	42.81	14.52	77.52	61.06	33.39
Shih $BP(AC_T)$ + *SpTD*	512	56.27	44.51	21.94	73.15	61.25	37.95
$BP(AC_T)$ + *SpTD*	512	63.48	47.62	22.83	77.72	62.14	37.09
Shih $BP(AC_T)$ + *SpTD*	4096	63.15	48.61	24.5	76.1	62.84	39.78
$BP(AC_T)$ + *SpTD*	4096	**67.86**	**51.92**	**27.15**	**81.93**	**64.95**	**39.92**
		mAP (AQE)	mAP (AQE)	mAP (AQE)	mAP (AQE)	mAP (AQE)	mAP (AQE)
Shih ChCO-SC_T + *SpTD*	512	56.7	44.29	18.2	85.04	71.56	47.55
ChCO-SC_T + *SpTD*	512	57.98	44.32	16.56	84.57	70.37	45.45
ChCO-SC_T + *SpCt*	512	61.23	44.34	17.44	86.7	71.24	44.91
Shih $BP(AC_T)$ + *SpTD*	512	53.1	43.66	19.31	79.49	68.04	45.55
$BP(AC_T)$ + *SpTD*	512	56.01	43.69	19.33	83.17	68.88	44.38
Shih $BP(AC_T)$ + *SpTD*	4096	59.82	49.33	25.34	85.36	74.45	52.37
$BP(AC_T)$ + *SpTD*	4096	**64.33**	**51.28**	**26.64**	**88.11**	**73.74**	**50.23**

occurrences $BP(AC_T)_{learn}$ obtains the best result, showing its ability to capture even better the relations between the features of the activation map.

6. Comparison with state-of-the-art results

In this section, we compare the results of our method with the Shih et al. [10] *co-occurrence* interpretation method and also with other state-of-the-art methods in image retrieval.

6.1. Co-occurrence representation comparison

Shih et al. [10] define *co-occurrences* as the maximal correlation between a pair of feature maps, for a set of spatial offsets, whilst we define *co-occurrences* as the sum of the activations inside a region, being the activations value above a threshold (Section 3.1).

In order to compare these two different interpretations, we have modified Shih et al. [10] method to return a *co-occurrence* tensor instead of a *co-occurrences* vector. This modification consists in picking the summed maximum correlation map instead of the own value. This produces a tensor 3D tensor $C_T^{'} = \mathbb{R}^{M\times N\times D^2}$ with D^2 channels. Each channel of this tensor represents the correlation between a pair of channels. The aggregation of all the correlations of each channel with the rest produce a tensor $C_T{}'' = \mathbb{R}^{M\times N\times D}$, with the same size than the original activation tensor A as in our method.

In Table 5, we present the comparison for linear and bilinear aggregation schemes using both *co-occurrence* representation methods. The *mAP* results obtained with the lineal aggregation are quite similar for both methods, but when bilinear aggregation is used our method outperforms significantly Shih et al. method. Furthermore, Shih et al. method has an efficiency drawback in its implementation, because it is necessary to find the maximum correlation value for each pair of channels.

For this reason, we have compared the execution time of the *co-occurrence* tensor C_T generation of both methods, with a subset of one hundred images of Paris6k. The feature map of each image was extracted with two pre-trained networks VGG [12] and ResNet [13], resulting in tensors of 32 × 24 (width × height) with 512 channels (VGG) and 2048 channels (ResNet). In addition, we have studied the

Table 6
Performance comparison between *co-occurrence* methods.

Tensor size	Ours (single)	DeepCooc [10] (single)	Ours (batch 5)	DeepCooc [10] (batch 5)
32 × 24 × 512 (VGG)	0.977 ms	464.699 ms	0.695 ms	495.66 ms
32 × 24 × 2048 (ResNet)	16.678 ms	2417.895 ms	5.578 ms	*Out-of-memory*
32 × 24 × 32	0.258 ms	29.321 ms	0.318 ms	35.227 ms

Table 7
Comparison with state-of-the-art results for ROxford and RParis datasets.

		ROxford			RParis		
		Easy	Medium	Hard	Easy	Medium	Hard
Method	Size	mAP	mAP	mAP	mAP	mAP	mAP
Pre-trained single pass							
ucrow	512	60.53	41.15	11.98	74.75	57.69	30.20
ucrow + A	512	55.32	39.94	13.08	81.32	65.29	38.72
crow [7]	512	61.92	44.66	17.94	76.13	60.17	33.38
crow + A [7]	512	55.75	42.11	17.89	81.82	67.93	43.16
SPoC [5]	512		38.0	11.4		59.8	32.4
MAC [6]	512		37.8	14.6		59.2	35.9
R-MAC [6]	512		42.5	12.0		66.2	40.9
GeM [31]	512		40.5	15.7		63.2	38.8
ChCO- + (ours)	512	63.37	47.63	21.96	78.79	62.59	36.84
ChCO- + + A (ours)	512	60.27	46.58	21.19	85.19	71.01	46.43
+ ms (ours)	8192	72.53	56.34	32.93	90.13	71.86	48.22
+ + ms (ours)	8192	**74.53**	**60.86**	**38.50**	**93.93**	**80.35**	**60.44**
Fine-tuning							
Radenovic VGG16-GeM [31] + ms	512		61.9	33.7		69.3	44.3
Radenovic VGG16-GeM [31] ++ ms	512		66.6	38.9		74.0	51.0
+ ms (ours)	8192	74.65	57.94	33.68	90.18	72.44	48.37
+ + ms (ours)	8192	**82.25**	**67.04**	**42.23**	**94.21**	**80.97**	**61.42**

performance with a smaller tensor of 32 channels depth (because is the depth of tensors used in the Shih et al. implementation).

Table 6 shows the average time after fifty executions of each experiment using a computer with a CPU i7-7700K@4.20 GHz, GPU GeForce GTX1080Ti, and 32 GB of RAM.

As we can see, our implementation is more than hundred times faster than the Shih et al. method. Therefore, we have demonstrated that our method allows the use of *co-occurrences* representations, breaking the performance barrier that made *co-occurrences* calculation out of the reach for many applications.

6.2. Comparison with state-of-the-art results

In Table 7, we present the results in ROxford and RParis datasets of state-of-the-art methods which uses VGG as feature extractor. In the pre-trained single pass category we improve the state-of-the-art performance with ChCO-SC_T + *SpTD* based in the linear aggregation of *co-occurrences* against well-known image retrieval methods like crow [7], SPoC [5], MAC and R-MAC [6] and GeM [31]. Moreover, with bilinear pooling we can obtain a final vector representation with higher dimensions than the number of channels of the last VGG layer. Using Off-The-Shelf VGG and $BP(AC_T)$ with a final vector size of 8192, a great *mAP* improvement is achieved.

Finally, we compare our *co-occurrence* representation based on trainable *co-occurrence* filter against GeM pooling [37] using the same training and fine-tuning procedure for both methods. Again, we demonstrate a huge improvement as consequence of adding *co-occurrence* information to the final vector representation, even when only the *co-occurrence* filter is trained.

7. Conclusions

In this work, we have presented a new definition for *co-occurrence* tensor of deep convolutional features. This *co-occurrence* representation embeds relevant information of the image, allowing us to add discriminative information to the compact final image representations for image retrieval. In addition, our *co-occurrence* implementation allows to learn the *co-occurrence* filter to have better *co-occurrence* representations.

In our approach, we combine the proposed *co-occurrence* tensor by means of weighted linear pooling and bilinear pooling with the original tensor of activations in a simple a straight forward pipeline. In the experimental results, we have evidenced that the *co-occurrence* tensor improve the results over the standard procedure, so the ability of *co-occurrences* to capture additional information and create powerful image representations was demonstrated.

For future research, we plan to study other aggregation and normalization schemes for *co-occurrence* and adapt our methodology on multi regional image representation [5].

Declaration of competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Juan I. Forcen: Investigation, Software, Methodology, Validation, Writing - original draft.**Miguel Pagola:**Investigation, Conceptualization, Formal analysis, Writing - original draft.**Edurne Barrenechea:**Supervision, Writing - review & editing.**Humberto Bustince:**Supervision, Writing - review & editing, Funding acquisition.

Acknowledgments

We are grateful to the NVIDIA corporation for supporting our research in this area donating a TITAN V.

References

[1] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: spatial pyramid matching for recognizing natural scene categories, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. 2 (2006) 2169–2178, https://doi.org/10.1109/CVPR.2006.68.
[2] D.G. Lowe, Object recognition from local scale-invariant features, Proceedings of the Seventh IEEE International Conference on Computer Vision, Vol. 2, 1999, pp. 1150–1157, https://doi.org/10.1109/ICCV.1999.790410.
[3] L. Zheng, Y. Yang, Q. Tian, SIFT meets CNN: a decade survey of instance retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 40 (5) (2018) 1224–1244, https://doi.org/10.1109/TPAMI.2017.2709749.
[4] A.S. Razavian, H. Azizpour, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition, Proceedings of the IEEE International Conference on Computer Vision, 2014.
[5] A.B. Yandex, V. Lempitsky, Aggregating local deep features for image retrieval, Proceedings of the IEEE International Conference on Computer Vision 2015 Inter 2015, pp. 1269–1277, https://doi.org/10.1109/ICCV.2015.150.
[6] G. Tolias, R. Sicre, H. Jégou, Particular Object Retrieval with Integral Max-Pooling of CNN Activations, ICL 2016 — International Conference on Learning Representations, May 2016, San Juan, Puerto Rico, 2015 1–12.
[7] Y. Kalantidis, C. Mellina, S. Osindero, Cross-dimensional weighting for aggregated deep convolutional features, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Lect. Notes Comput. Sci 9913 (2016) 685–701, https://doi.org/10.1007/978-3-319-46604-048.
[8] A. Gordo, J. Almazán, J. Revaud, D. Larlus, End-to-end learning of deep visual representations for image retrieval, Int. J. Comput. Vis. (2017) https://doi.org/10.1007/s11263-017-1016-8.
[9] R.M. Haralick, K. Shanmugam, I. Dinstein, Textural features for image classification, IEEE Trans. Syst. Man Cybern. SMC-3 (6) (1973) 610–621, https://doi.org/10.1109/TSMC.1973.4309314.
[10] Y.F. Shih, Y.M. Yeh, Y.Y. Lin, M.F. Weng, Y.C. Lu, Y.Y. Chuang, Deep co-occurrence feature learning for visual object recognition, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. 2017 (2017-January) 7302–7311, https://doi.org/10.1109/CVPR.2017.772.
[11] J. Deng, W. Dong, R. Socher, L. Li, Imagenet: A large-scale hierarchical image database, Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (2009) 248–255, https://doi.org/10.1109/CVPR.2009.5206848.
[12] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, International Conference on Learning Representations, 2015.
[13] A. Krizhevsky, I. Sutskever, G.E. Hinton, AlexNet, Advances in Neural Information Processing Systems, 2012https://doi.org/10.1016/B978-008046518-0.00119-7.
[14] J. Cao, L. Liu, P. Wang, Z. Huang, C. Shen, H.T. Shen, Where to Focus: Query Adaptive Matching for Instance Retrieval Using Convolutional Feature Maps, 2016 1–10.
[15] A. Jimenez, J.M. Alvarez, X. Giro-i Nieto, Class-weighted convolutional features for visual instance search, 28th British Machine Vision Conference (BMVC), 2017.
[16] E. Mohedano, K. McGuinness, X. Giro-I-Nieto, N.E. O'Connor, Saliency weighted convolutional features for instance search, Proceedings - International Workshop on Content-Based Multimedia Indexing, Vol. 2018, , Septe, 2018https://doi.org/10.1109/CBMI.2018.8516500.

[17] O. Simeoni, A. Iscen, G. Tolias, Y. Avrithis, O. Chum, Unsupervised Object Discovery for Instance Recognition, Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, 2018, 2018-January 1745–1754, https://doi.org/10.1109/WACV.2018.00194.
[18] Y. Yang, S. Newsam, Bag-of-visual-words and spatial extensions for land-use classification, Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10 2010, p. 270, https://doi.org/10.1145/1869790.1869829.
[19] S. Elkerdawy, N. Ray, H. Zhang, Fine-grained vehicle classification with unsupervised parts co-occurrence learning, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Lect. Notes Comput. Sci 11132 (2019) 664–670, https://doi.org/10.1007/978-3-030-11018-5_54.
[20] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2007.
[21] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Lost in quantization: improving particular object retrieval in large scale image databases, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2008.
[22] J.B. Tenenbaum, W.T. Freeman, Separating style and content with bilinear models, Neural Comput. 12 (6) (2000) 1247–1283, https://doi.org/10.1162/089976600300015349.
[23] J. Carreira, R. Caseiro, J. Batista, C. Sminchisescu, Semantic segmentation with second-order pooling, in: A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, C. Schmid (Eds.), Computer Vision – ECCV 2012, Springer, Berlin, Heidelberg 2012, pp. 430–443.
[24] C. Guillot-Soulez, S. Soulez, L'analyse conjointe: présentation de la méthode et potentiel d'application pour la recherche en GRH, Revue de Gestion Des Ressources Humaines 80 (2) (2014) 33, https://doi.org/10.3917/grhu.080.0033.
[25] A.R. Chowdhury, T. Lin, S. Maji, E. Learned-Miller, One-to-many face recognition with bilinear cnns, 2016 IEEE Winter Conference on Applications of Computer Vision (WACV) 2016, pp. 1–9, https://doi.org/10.1109/WACV.2016.7477593.
[26] Y. Gao, O. Beijbom, N. Zhang, T. Darrell, Compact bilinear pooling, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016, pp. 317–326, https://doi.org/10.1109/CVPR.2016.41.
[27] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, O. Chum, Revisiting oxford and paris: Large-scale image retrieval benchmarking, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018, pp. 5706–5715.
[28] H. Jegou, M. Douze, C. Schmid, Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search, in: Proceedings of the 10th European Conference on Computer Vision: Part I, ECCV '08, Springer-Verlag, Berlin, Heidelberg, 2008 304–317, https://doi.org/10.1007/978-3-540-88682-224.
[29] R. Arandjelović, A. Zisserman, Three things everyone should know to improve object retrieval, 2012 IEEE Conference on Computer Vision and Pattern Recognition 2012, pp. 2911–2918, https://doi.org/10.1109/CVPR.2012.6248018.
[30] G. Tolias, H. Jégou, Visual query expansion with or without geometry: refining local descriptors by feature aggregation, Pattern Recogn. 47 (10) (2014) 3466–3476. doi: https://doi.org/https://doi.org/10.1016/j.patcog.2014.04.007.
[31] F. Radenovic, G. Tolias, O. Chum, Fine-tuning CNN image retrieval with no human annotation, IEEE Trans. Pattern Anal. Mach. Intell. (2018) 1–14, https://doi.org/10.1109/TPAMI.2018.2846566.
[32] E. Mohedano, K. McGuinness, X. Giro-i Nieto, N. E. O'Connor, Saliency Weighted Convolutional Features for Instance Search.
[33] S. Chopra, R. Hadsell, Y. LeCun, Learning a similarity metric discriminatively, with application to face verification, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), Vol. 1, 2005, pp. 539–546, https://doi.org/10.1109/CVPR.2005.202.
[34] G.R. Koch, Siamese Neural Networks for One-Shot Image Recognition, 2015.
[35] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, P.H.S. Torr, Fully-convolutional siamese networks for object tracking, in: G. Hua, H. Jégou (Eds.), Computer Vision – ECCV 2016 Workshops, Springer International Publishing, Cham 2016, pp. 850–865.
[36] D. Kingma, J. Ba, Adam: a method for stochastic optimization, International Conference on Learning Representations.
[37] F. Radenović, G. Tolias, O. Chum, Cnn image retrieval learns from bow: unsupervised fine-tuning with hard examples, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham 2016, pp. 3–20.