University Name

Doctoral Thesis

The fishy thing

Author:
Marco Herrero

Supervisor:
Dr. James Smith

A thesis submitted in fulfillment of the requirements for the degree of Máster Universitario de Lógica, Computación e Inteligencia Artificial

in the

Research Group Name
Department or School Name

June 6, 2017

Declaration of Authorship

I, Marco Herrero, declare that this thesis titled, "The fishy thing" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"*Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism.*"

Dave Barry

University Name

Abstract

Faculty Name

Department or School Name

Máster Universitario de Lógica, Computación e Inteligencia Artificial

The fishy thing

by Marco Herrero

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Contents

List of Figures

List of Tables

List of Abbreviations

LAH **L**ist **A**bbreviations **H**ere
WSF **W**hat (it) **S**tands **F**or

Physical Constants

Speed of Light $\quad c_0 = 2.997\,924\,58 \times 10^8\,\mathrm{m\,s^{-1}}$ (exact)

List of Symbols

a	distance	m
P	power	W ($\mathrm{J\,s^{-1}}$)
ω	angular frequency	rad

For/Dedicated to/To my...

Chapter 1

Introducción

1.1 La plataforma

Dentro de los muchos avances que internet ha hecho posible se encuentra la capacidad de que varias personas colaboren solucionando el mismo problema. Esto se ha notado especialmente en el mundo del software, donde grandes proyectos de software libre han salido adelante gracias al aporte de muchas personas de diferente nacionalidad.

Chapter 2

Conceptos básicos

2.1 Redes neuronales

Introducción a las redes neuronales, back-propagation, capas, input-outpus, etc

2.2 Convoluciones

Una imagen en el ámbito digital se entiende como una matriz de puntos. Cada uno de estos puntos puede ser interpretado como un número, que representa la localización de este punto en la escala de grises. Para representar la posición del punto en la escala de grises usaremos los valores menores para los puntos más oscuros y los mayores para los más claros.

Por lo tanto, una imagen como la siguiente:

Sería representada por esta matriz.

```
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.3529, 0.5412, 0.9216, 0.9216, 0.9216, 0.9216, 0.9216]
[ 0.    , 0.    , 0.549 , 0.9843, 0.9961, 0.9961, 0.9961, 0.9961, 0.9961, 0.9961]
[ 0.    , 0.    , 0.8863, 0.9961, 0.8157, 0.7804, 0.7804, 0.7804, 0.7804, 0.5451]
[ 0.    , 0.    , 0.149 , 0.3216, 0.051 , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
[ 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    , 0.    ]
```

Para simplificar, la mayoría de los ejemplos van a usar una escala de grises, pero son aplicables a imágenes RGB aplicando las operaciones a las tres diferentes capas al mismo tiempo.

2.2.1 Filtros

Al trabajar con esta interpretación de lo que es una imagen, se pueden usar operaciones sobre la matriz de la imagen para transformarla de diferentes maneras.

Imaginemos una matriz filtro de 3×3 (también llamada matriz de convolución):

$$F = \begin{bmatrix} -1 & -1 & -1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Se puede usar la matriz como un filtro para una imagende la siguiente manera: Primero, superponemos la matriz en algún punto de la imagen. Esto modificará el pixel donde ha quedado colocado el valor central de la matriz F. Multiplicamos cada uno de los valores superpuestos, sumamos los resultados y los sustituimos en el valor central. Esto se hace para cada píxel de la imagen original (superponer el filtro en ese píxel y sustituir el valor por la operación).

En el caso de la matriz F, la fila superior son todos valores negativos, la intermedia son todos 1 y la inferior todos ceros. Si aplicamos la operación descrita con la matriz F sobre una imagen, los píxeles más brillantes (aquellos con mayor valor) serán los que su fila superior es cero, eliminando los valores negativos y la fila intermedia es 1. Esto ocurrirá con más frecuencia en los bordes superiores de objetos claros con fondo oscuro.

Para ver la utilidad vamos a verlo aplicado a la imagen de un dígito escrito a mano, sacado del dataset MNSIT (cita req).

Si aplicamos el filtro a la imagen podemos observar como resalta en blanco los bordes superiores y en negro los inferioes. Filtros similares, rotando los valores del filtro F, son capaces de resaltar bordes lateras u oblicuos.

Lo interesante de este método es que hemos conseguido resaltar características del objeto representado en la imagen solo multiplicando matrices.

Las matrices de convolución pueden ser de mayor tamaño, permitiendo capturar características más complejas. La matrix de 3×3 es la menor matriz que permite definir en su totalidad el concepto de espacio, pudiendo extraer características espaciales en dos dimensiones.

A la hora de trabajar con imágenes en color es necesario usar un modelo de color. Uno de los más usados, RGB, se compone de tres capas, una dedicada a la intensidad roja, otra a la verde y la última a la azul, de ahí su nombre. Cada filtro se aplicaría a cada capa por separado, permitiendo de esta manera detectar diferentes características de la imagen que ocurran solo en uno de los colores.

Para ver como afectan diferentes filtros a una imagen, existe una página (http://setosa.io/ev/image-kernels/ mover a biblio) donde se pueden probar ejemplos con filtros personalizados, haciendo el concepto mucho más sencillo de comprender.

2.3 Redes neuronales convolucionales

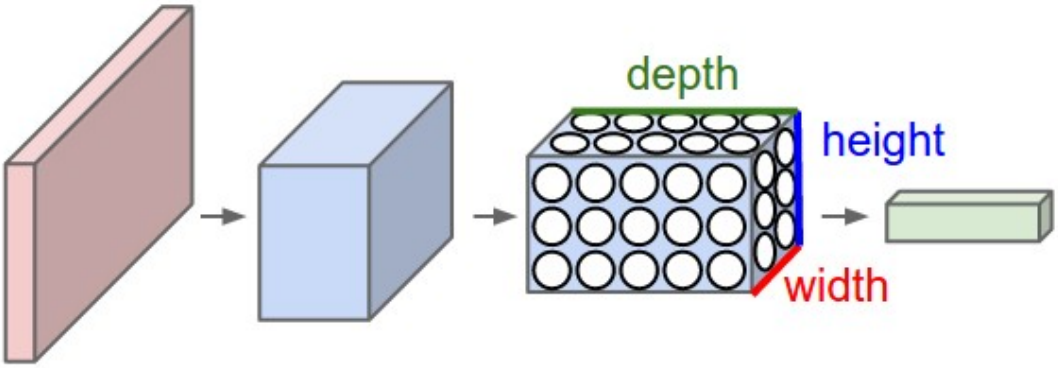
Hemos explorado la idea de que determinados filtros son capaces de extraer información localizada sobre características de la imagen. En el ejemplo del apartado anterior, dada una imagen podíamos saber si había bordes superiores y dónde se podían encontrar. Esto puede ser de gran utilidad en el campo de reconocimiento de imágenes, ya que podemos usar esa información localizada para categorizar o aplicar otro tipo de técnicas en esas áreas señaladas.

El problema está en cómo encontrar los mejores filtros para sacar las características más relevantes de una imagen.

Analizando como funcionan los filtros convolucionales vemos su parecido con las redes neuronales. Al igual que las redes neuronales, los filtros son matrices que estamos aplicando a los datos de entrada que producirán unos datos de salida relevantes con la función buscada. El entrenamiento de una red neuronal va modificando los pesos de las diferentes capas hasta que produce una salida relevante con los ejemplos del conjunto de entrenamiento.

Si entendemos los pesos como la matriz convolucional, podemos hacer que sea la misma red la que busque el mejor filtro para nuestro problema de clasificación. De hecho, ya que las redes neuronales son capaces de componer diferentes funciones en diferentes capas de la red para imitar funciones no lineales, podemos aplicar la misma idea a las redes con filtros: componer diferentes filtros para poder extraer características más complejas.

En esta idea se basan las redes convolucionales que trabajan con imágenes. Cada capa de la red va a aplicar varios filtros a la imagen y a devolver las imágenes tratadas por dicho filtro para pasarlo a la siguiente capa. Ya que lo que importa es la composición de filtros, cada capa deberá entrenar varios filtros al mismo tiempo, permitiendo así aumentar la combinatoria final.



La manera de representar esto es entendiendo los pesos de cada capa como una matriz de tres dimensiones. Estas tres dimensiones se pueden entender como d (profundidad) filtros de dimensión $w \times h$ (anchura y altura). Cada capa tendrá su conjunto de filtros.

2.3.1 Estructura de una capa convolucional

El concepto de capa en una red convolucional incluye una agrupación de diferentes capas que efectúan diferentes operaciones. Como en las redes neuronales clásicas, la capa precisará de un tratamiento de las entradas con los pesos (en este caso los filtros convolucionales), y una función de activación aumentar la relevancia de las activaciones.

Capa convolucional - CONV

Las capas convolucionales son las que hemos visto hasta ahora. Tendrán como pesos n filtros convolucionales (todos del mismo tamaño), y producirán las n salidas de aplicar la imagen de entrada a estos filtros. Una imagen de tamaño 32×32 aplicada sobre una capa convolucional de 8 filtros producirá una salida de tamaño $32 \times 32 \times 8$, o lo que es lo mismo, 8 imágenes de 32×32

Capa de activación - RELU

Al igual que en las redes clásicas es necesario tratar la salida de la aplicación de los pesos sobre las entradas con una función de activación.

La manera estándar de modelar la salida de una neurona en las RNAs es a través de $f(x) = tanh(x)$ o $f(x) = (1 + e^{-x})^{-1}$ (función sigmoide). A la hora de entrenar la red con descenso del gradiente, estas funciones mucho más lentas que otra función que también evita la linealidad: $f(x) = max(0, x)$, llamada ReLU (*Rectifier Linear*). Las redes neuronales convolucionales entrenan varias veces más rápido usando ReLU (ver figura 2.1) que las equivalentes usando la función *tanh* (Krizhevsky, Sutskever, and Hinton, 2012)

FIGURE 2.1: Evolución de un entrenamiento para una red convolucional de cuatro capas usando ReLUs (línea sólida) vs usando tanh (línea discontínua) (Krizhevsky, Sutskever, and Hinton, 2012)

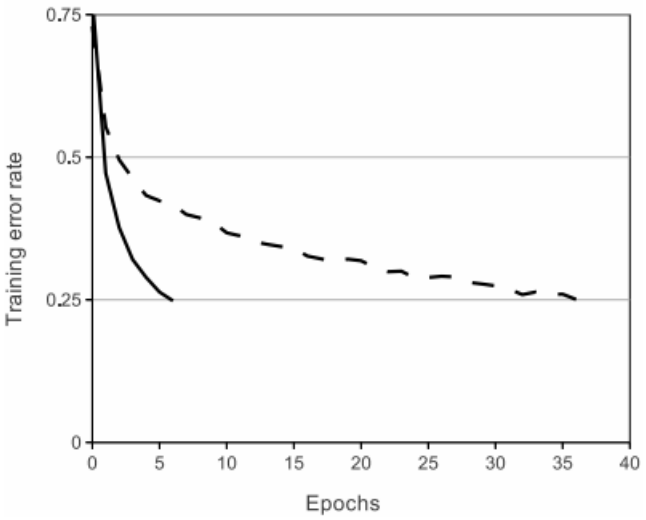
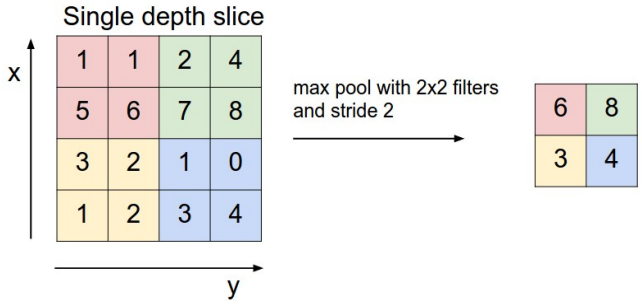


FIGURE 2.2: *Max-pooling sobre una imagen de* 4×4



Capa de muestreo - POOL

Al usar convoluciones en la imagen sobre el entorno de cada punto la información relevante de este queda difuminada y redundante. Gracias a esto se puede reducir el tamaño del problema mediante muestreo. Las capas de muestreo (*pooling layers*) resumen las salidas del entorno de cada grupo de neuronas. Un ejemplo de función de muestreo sería elegir el valor máximo de cada cuadrado de 4 píxeles (figura 2.2)

Si la salida de una capa CONV + RELU es $32 \times 32 \times 8$, al aplicar la capa de muestreo, la salida se convertirá en $16 \times 16 \times 8$. Al eliminar el 75% de las activaciones se reducen la cantidad de parámetros y el tiempo de computación, y además ayuda a reducir el sobreajuste.

Capa densa - FC

Esta capa es una capa clásica de red neuronal. Su función es calcular las probabilidades de clasificación dadas las imágenes tratadas. Cada neurona de esta capa estará conectada a cada una de las activaciones de la capa anterior, produciendo una salida por cada clase a clasificar.

Esta capa suele usarse al final de la arquitectura, cuando se han producido varios ciclos de capas convolucionales (CONV + RELU + POOL). Sin embargo, también es común ver arquitecturas con varias capas FC conectadas al final, ampliando la flexibilidad de clasificación de la red en base a características extraidas.

Chapter 3

Definición del problema

3.1 Kaggle

Kaggle es una plataforma donde se organizan competiciones de modelos predictivos o de aprendizaje automático sobre un problema real. Diferentes organizaciones plantean un problema y ofrecen un premio para el mejor modelo que lo resuelva, haciendo que miles de participantes prueben ideas diferentes y consigan puntos de vista que, en organizaciones tradicionales, un solo equipo no tendría.

En este trabajo se ha optado por la participación en uno de esos problemas, The Nature Conservancy Fisheries Monitoring.

3.2 The Nature Conservancy Fisheries Monitoring

En el océano pacífico, donde se captura más del 60% del atún del mundo, tienen lugar prácticas de pesca irregular que amenazan a los ecosistemas marinos y a la estabilidad de la pesca mundial. The Nature Conservancy es una asociación que trabaja con organizaciones locales y globales para preservar las especies marinas de cara al futuro.

La principal idea para controlar la pesca y explotación de recursos marinos es el uso de cámaras en barcos, que ayudan a monitorizar las actividades pesqueras de estos. Aunque funciona muy bien como sistema de control, la cantidad de datos e imágenes generadas hace que sea muy costoso de procesar manualmente.

La idea de este reto es desarrollar algoritmos que detecten y clasifiquen automáticamente especies de atunes, tiburones y otras especies que estos barcos pesqueros

cazan. El hecho de que se pueda analizar este tipo de imágenes de una manera rápida y automática permitirá asignar recursos de una manera mucho más efectiva para el control de este tipo de actividades. (Cita: fisheries monitoring)

3.3 Definición del problema

El problema consiste en clasificar cada una de las imágenes de un conjunto de imágenes de barcos en una de las ocho categorías disponibles. Las imágenes suelen mostrar la cubierta de un barco donde puede aparecer un pez. En base al pez que aparezca hay que clasificarlo en una de las siguientes seis categorías:

En caso de que no aparezca ningún pez en la imagen, esta tendrá la categoría NOF (No Fish). Y si aparece un pez en la imagen pero no perteneciente a ninguna de las categorías mencionadas, la categoría será OTHER.

$$categories = [ALB, BET, DOL, LAG, SHARK, YFT, OTHER, NOF]$$

3.3.1 Conjunto de datos

El conjunto de datos que se provee en el reto consta de tres elementos:

1. Conjunto de datos de entrenamiento: 3777 imágenes etiquetadas con una de las ocho categorías existentes.

Table 3.1: Ejemplo del archivo de envío

image	ALB	BET	DOL	LAG	NoF	OTHER	SHARK	YFT
img_00005.jpg	0.455	0.052	0.030	0.0173	0.123	0.079	0.046	0.194
img_00007.jpg	0.455	0.052	0.030	0.0173	0.123	0.079	0.046	0.194
img_00009.jpg	0.455	0.052	0.030	0.0173	0.123	0.079	0.046	0.194

2. Conjunto de test y evaluación: 1000 imágenes sin etiquetar.

3. Archivo de envío de prueba: Archivo CSV que muestra la estructura que debe tener el archivo con las soluciones

3.3.2 Envío de la solución y evaluación

Para el envío de la solución hace falta clasificar las 1000 imágenes del conjunto de evaluación, indicando la probabilidad de que caiga en cada una de las ocho categorías diferentes. En la tabla 3.1 se muestra un ejemplo de las primeras filas del archivo CSV a enviar.

El envío se evalúa usando multi-class logarithmic loss. Cada imagen ha sido etiquetada con una clase, pero es necesario enviar las probabilidades de cada clase para cada imagen. La formula final es:

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} \log\, p_{ij}$$

, siendo N el número de imágenes en el conjunto de test, M el número de clases, y_{ij} es 1 si la observación i pertenece a la clase j y 0 si no pertenece y p_{ij} la probabilidad predecida de que el elemento i pertenezca a la clase j.

Leaderboard y fases de la competición

Existe una tabla donde se publican los resultados de los envíos de cada participante. Estos resultados están calculados solo con un subconjunto del conjunto de test, de esta manera se evita que los participantes puedan aprovechar un sobreajuste para subir en la clasificación.

La puntuación final vendrá determinada por la puntuación de todo el conjunto de test.

Esta manera de evaluar es algo bastante común en kaggle, sin embargo los organizadores de este reto en particular han decidido hacer un pequeño cambio a la hora de evaluar la puntuación.

3.3.3 Doble fase de evaluación y envío

La evaluación en este reto contará de dos partes. La primera, donde se entrenará el modelo, requiere (como ya se ha dicho) evaluar el conjunto de test de mil imágenes. La fecha límite para la entrega de modelos de esta primera fase es siete días antes de

la finalización del reto. Será necesario enviar el CSV con las predicciones y el modelo usado.

Una vez finalice la primera fase, un segundo conjunto de test de muchas más imágenes será hecho público. Este conjunto de test deberá ser evaluado con el modelo generado en la fase anterior, siendo razón de descalficación modificar el modelo. Todos los participantes que no envíen la predicción de la segunda fase serán eliminados del reto.

Chapter 4

Soluciones presentadas

4.1 Idea

A la hora de afrontar este problema de clasificación de peces es lógico seguir una estrategia separada en dos pasos: primero buscar si existe un pez en la foto y luego intentar clasificarlo en una de las categorías existentes.

Para encontrar un pez en la foto es necesario encontrar una serie de características que puedan ser identificadas con algún pez. La idea de la solución parte de esta base. A la hora de clasificar una imagen primero es necesario encontrar el contenido relevante para ser usado en la clasificación.

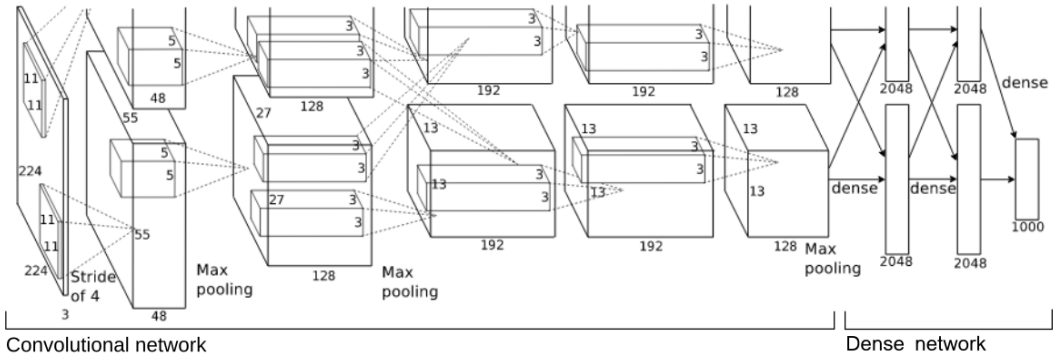
Las arquitecturas encontradas en problemas similares (Krizhevsky, Simonyan), permiten separar con claridad estas dos etapas, descubrimiento de caracteristicas y clasificación.

4.2 Arquitectura

La arquitectura general usada, que luego sufrirá pequeños cambios, es la descrita en la figura 4.1 (Krizhevsky et al.)

Esta arquitectura usa en su primera parte una red convolucional preentrenada sobre un conjunto de imágenes mucho más generalista, en este caso Imagenet. Al

FIGURE 4.1: Arquitectura de la red en dos partes



usar una red convolucional preentrenada sobre fotografías tendrá muchas más capacidad para distinguir características de diferentes objetos, además que existen varias categorías de peces en Imagenet, por lo que sabrá diferenciar este tipo de fotografías.

4.2.1 Red convolucional

Como explicamos en (cap redes convolucionales), cuando un modelo convolucional recibe una imagen va a devolver N matrices bidimensionales representando el resultado de las aplicación de los N conjuntos de filtros a la imagen inicial. Al ser N matrices bidimensionales también puede considerarse una matriz tridimensional.

Intuitivamente, y haciendo una simplificación, podemos pensar que cada uno de estos filtros representa un mapa de calor de la aparición en la imagen de diferentes características. Por ejemplo, ¿cúanto se parece cada parte de esta imagen a la piel de un pez?.

Hay que tener en cuenta que este modelo convolucional no ha sido entrenado con el conjunto de entrenamiento, si no con el conjunto de entrenamiento de Imagenet. Para lo único que vamos a usar esta red convolucional es para transformar estas imágenes a

4.2.2 Modelo preentrenado

Aqui detallo las implementaciones de los modelos basados en Imagenet (los disponibles en keras.io): VGG, ResNet e Inception. Tambien tengo que decir porque prefiero VGG a otros modelos, sobre todo a nivel didactico.

4.2.3 Fine tuning

Bibliography

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1097–1105.