MINERÍA DE DATOs ASPECTOS AVANZADOS

MASTER EN CIENCIA DE DATOS E INGENIERIA DE COMPUTADORES

# Práctica final: Deep Learning


**Autores**

José Ángel Díaz García, Manuel Payan Cabrera y Gerardo Fernández Rodríguez

**Equipo**

Equipo8


**Puntuación y Posición**

0.97500 - 10



Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación


—

Granada, Abril de 2018

# Índice general

# Índice de figuras

# Índice de tablas

# Capítulo 1

# Introducción

Este documento presenta la memoria del trabajo realizado para la resolución de la práctica final de la asignatura 'Minería de datos, aspectos avanzados', enmarcada dentro del Máster en Ciencia de Datos de la universidad de Granada.

## 1.1. Motivación

La reciente incursión de las técnicas de minería de datos en actividades cotidianas o diarias constatando sus innumerables aplicaciones en diversos dominios o problemas, han propiciado que de en año en año sean más los profesionales de sectores muy dispares que deciden formarse en estas tareas que les lleven a ostentar en un futuro próximo el título de científicos de datos.

A caballo entre la minería de datos y la inteligencia artificial ha resurgido la idea de simular el comportamiento de la mente humana gracias al deep learning. El deep learning es un subconjunto dentro del match learning, que intenta llevar a la computación la idea de aprendizaje desde el ejemplo. La idea consiste en obtener un modelo para probar ejemplos y un conjunto de reglas para modificar el modelo en el caso que se produzca un error. Para llevar a cabo este proceso se utilizan redes neuronales.

Este creciente interés ha propiciado también un caldo de cultivo perfecto para diversas plataformas online que ayudan a estos profesionales en su

proceso de formación siendo una de las plataformas más conocidas Kaggle [1].

Esta plataforma, se basa en proponer problemas reales de minería de datos que cualquier persona interesada en la temática puede intentar resolver, en muchos casos sin grandes necesidades de cómputo o máquinas potentes. Esta plataforma evalúa los resultados aportados por cada participante y evalúa un ranking en función de diversas medidas de bondad, lo que la ha convertido también en una herramienta esencial entre los docentes del ámbito de la ciencia de datos que la usan para enfrentar a sus alumnos con problemas reales de ciencia de datos. Esta memoria, se centra por tanto en detallar el proceso llevado a cabo por los autores para resolver un problema propuesto por los profesores de la asignatura en la plataforma Kaggle. El problema en cuestión, será definido en la siguiente sección.

## 1.2. Problema y Dataset

El problema propuesto, es un problema de clasificación binaria de imágenes [2] en el cual deberemos crear un modelo basado en redes neuronales artificiales, capaz de clasificar las imágenes correctamente. Las clases a clasificar son **armas de fuego** y **smartphones**. Para entrenar y validar el modelo disponemos de los siguientes datos:

- **train**: Tenemos dos carpetas, una con las armas de fuego y otras de smartphones. Concretamente tendremos 392 referentes a armas de fuego y 351 referentes a smartphones.
- **test**: 800 ítems mezclados de ambas clases a clasificar

### 1.2.1. Evaluación

La evaluación de la práctica es mediante la medida del Accuracy estándar, por lo que no habrá penalización entre falsos negativos y falsos positivos, como ocurriría en un problema similar por ejemplo con datos médicos.

## 1.3. Herramientas y objetivos

En esta sección veremos una breve introducción a las herramientas usadas para el desarrollo de la práctica así como de los principales objetivos que se buscan conseguir con el desarrollo de la misma.

### 1.3.1. Hardware

Las herramientas hardware usadas han sido los pc personales de cada uno de los miembros del equipo, en los cuales alguno carece de GPU y en los demás no se ha podido configurar correctamente las mismas por lo que el 100 % del computo ha sido llevado a cabo en CPUs. Las máquinas usadas pueden verse en las siguientes tablas.

| Elemento | Características |
|---|---|
| Procesador | 2,6 GHz Intel Core i5 |
| GPU | - |
| Memoria Ram | 8 GB 1600 MHz DDR3 |
| Disco duro | SATA SSD de 120 GB |

Tabla 1.1: Especificaciones técnicas de la máquina 1.

| Elemento | Características |
|---|---|
| Procesador | Intel Core i7 6700HQ |
| Memoria Ram | 16 GB SDRAM |
| Disco duro | 128GB SSD |

Tabla 1.2: Especificaciones técnicas de la máquina 2.

| Elemento | Características |
|---|---|
| Procesador | Intel Core i7 6700HQ |
| Memoria Ram | 16 GB SDRAM |
| Disco duro | 128GB SSD |

Tabla 1.3: Especificaciones técnicas de la máquina 3.

### 1.3.2. Software

El software utilizado es en su práctica totalidad software libre, siendo el restante software propietario cuyas licencias vienen incluidas en el sistema operativo de las máquinas.

- **Tensorflow**: Entorno de deeplearning sobre Python.
- **Keras**: Capa de abstracción sobre Tensorflow.
- **Atom**: Editor de texto plano para la programación de los scripts.
- **TeXShop**: procesador de textos basado en *Latex* usado para elaborar la documentación del presente proyecto.

### 1.3.3. Objetivos

Los objetivos de este trabajo podrían resumirse en los siguientes:

- Obtener un modelo predictivo fiable que dado una nueva imagen pueda predecir si es un arma o un smartphone.
- Obtener un valor de *accuracy* aceptable para escalar posiciones en la competición de Kaggle.
- Comprender y estudiar las distintas técnicas de minería de datos avanzada vista en la asignatura.

## 1.4. Organización del trabajo

La organización del presente documento, se centra en detallar cada uno de los pasos seguidos durante el estudio y resolución del problema planteado en esta introducción, tras la cual en el capítulo 2.1 veremos la metodología y planificación temporal seguida por el equipo durante el transcurso de la práctica. En el capítulo 3 veremos las explicaciones asociadas al preprocesado de datos, más concretamente al data augmentation. El grueso de la memoria está en el capítulo 4 donde trataremos de ver las topologías y el proceso experimental seguido para alcanzar el resultado final. Finalizaremos la memoria con las conclusiones y vías futuras que quedan relegadas al capítulo 5.

# Capítulo 2

# Planificación

En este capítulo se detallan los enfoques de deep learning usados durante el transcurso de la competición así como la metodología de trabajo y desarrollo llevada a cabo por el equipo durante el transcurso de la práctica para favorecer mejores resultados y sobre todo optimizar tiempos.

## 2.1. Metodologías de Deep Learning

Dado que el equipo estaba formado por tres componentes se dividieron las directrices del mismo en las tres vertientes más estudiadas dentro de deep-learning:

1. **From Scratch**: Esta técnica consiste en crear y entrenar una red neuronal desde 0. En casos como el que nos encontramos, tendremos el problema de que hay una muestra muy pequeña de datos para el entrenamiento por lo que habrá que utilizar data augmentation para favorecer su mejora.

2. **Transfer Learning**: Esta técnica, se basa en utilizar redes ya entrenadas para clasificar un nuevo problema. Si el dominio del problema no es muy diferente a las imágenes usadas para entrenar la red inicial, puede que los resultados sean aceptables y tiempos de entrenamiento nulos si por otro lado, estamos en problemas muy especiales como casos

médicos esta opción no debe ni ser contemplada. En nuestro caso, al ser objetos comunes del mundo real si que la hemos contemplado.

3. **Fine Tunning**: Esta es la técnica que mejores resultados ha ofrecido en la competición permitiendo usar tipologías potentes y que son o han sido estado del arte en clasificación de imágenes. La idea de esta metodología reside en utilizar las primeras capas de redes ya entrenadas las cuales almacenan datos de curvas y aristas en gran medida y tras ello, se entrenan las ultimas capas con las imágenes de nuestro problema tras lo cual los resultados serán bastante buenos o aceptables.

## 2.2. Planificación temporal

El proyecto ha sido planificado temporalmente acorde a los siguientes puntos, tareas u objetivos:

- **Pruebas con modelos individuales**: En los primeros días cada uno de los miembros del equipo estudió una de las metodologías vistas en el punto 2.1. Se hicieron las primeras subidas a Kaggle.
- **Preprocesado**: Una vez se dio con la metodología a seguir, fine tunning, se estudiaron y variaron ciertas variantes del pre-procesado para obtener mejores resultados.
- **Estudio de mejores modelos**: Se estudio el estado del arte en la materia para adaptar al proceso de fine tunning modelos cuya potencia estuviera ya constatada.
- **Reuniones de grupo**: Reuniones de puesta en común de resultados y debate de técnicas y vías de estudio.
- **Redacción**: Tras la finalización de la competición se abre un período de redacción del presente documento.

En la figura 2.1, puede verse un croquis sobre el calendario del mes de marzo, en el que se detallan las distintas tareas que se han detallado en puntos anteriores.


Figura 2.1: Croquis de la planificación seguida.

## 2.3. Desarrollo

Para el desarrollo del código se ha usado un repositorio para el control de versiones en GitHub que puede verse en [3]. En este hemos creado ramas para cada uno de los miembros del equipo que eran fusionadas al realizar algún avance.

Dado que el proceso de entrenamiento con Keras permite guardar pesos, se ha llevado a cabo un desarrollo basado en **soluciones intermedias**. Con este método, se guardan los pesos de ciertas redes por lo que otros miembros del equipo por medio del método **read model** pueden leer estos pesos y utilizarlos para sus experimentos sin necesidad de volver a entrenar modelos ya entrenados.

# Capítulo 3

# Preprocesado

Dentro del trabajo como analista de datos el apartado del preprocesamiento de los datos es una de las partes más importantes que se han de abordar. En este capítulo enunciaremos el proceso de preprocesado llevado a cabo durante la realización de la práctica.

## 3.1. Resize data

Uno de los problemas que más nos hemos encontrado en la realización de la práctica es la dimensión de los datos, ya que las imágenes cambian de forma y dimensiones por lo que en función del modelo de red neuronal usado aplicaremos uno u otro tamaño de entrada.

En este problema en concreto, el problema no es muy elevado, pero en problemas de tipo médico con imágenes de gran resolución contar con un dataset fácilmente manejable en memoria es determinante. En nuestro caso, para las primeras aproximaciones hemos redimensionado los datos a **54x54** pixeles.

## 3.2. Data Augmentation

Uno de los principales problemas en *deep learning* es la falta de datos. Para ello, pueden usarse técnicas de *data augmentation* que consisten en aplicar

ligeras transformaciones a las imágenes para conseguir un conjunto de entrenamiento mayor, pudiendo obtener de una sola imagen 5 o 6 variaciones que enriquecen enormemente el modelo. Para ello, usando las funciones propias de Keras hemos aplicado las siguientes transformaciones a las imágenes:

- **rotation_range**: Se generan imágenes aleatorias que se rotan una cantidad de grados definidos.
- **rheight_shift_range**: Cambio aleatorio en la altura.
- **rwidth_shift_range**: Cambio aleatorio en el ancho.
- **rshear_range**: Rango de corte.
- **rzoom_range**: Zoom aleatorio.
- **rhorizontal_flip**: Flip aleatorio de forma horizontal.
- **rfill_mode**: Relleno de los puntos de la frontera.

Con todos estos cambios generamos un conjunto de imágenes más rico que nos proporciona variedad a la hora de entrenar el modelo, evitando o suavizando así el problema de no contar con muchos datos de entrada para la fase de entrenamiento, el cual sigue siendo uno de los grandes problemas a los que el deep learning se enfrenta.

## 3.3. Filtros de ruido

Además de las técnicas anteriormente descritas se han probado distintos filtros de ruido para mejorar el resultado de las imágenes. La mejor solución dentro de la aplicación de estas técnicas la hemos encontrado usando SMOOTH que se fundamenta en el suavizado de fronteras para que sea más sencillo la detección de objetos dentro de las imágenes.

# Capítulo 4

# Clasificación con Redes Neuronales

En este capítulo veremos el proceso seguido y las distintas vertientes de entrenamiento usadas a lo largo de la realización de la práctica. Concretamente veremos los resultados en función de la metodología y topologías usadas.

## 4.1. From Scratch

## 4.2. Transfer Learning

Para probar este punto hemos seguido la idea que puede verse en el siguiente código de github [4]. En el se usa la red Inception_BN, que es el estado del arte en clasificación de la batería de imágenes ***imagenet***, para clasificar las imágenes de nuestro problema. Este problema, tiene cientos de clases correspondientes a diferentes objetos u animales y clasifica una foto en función a estas, dado que nosotros tenemos entre manos objetos del mundo real y no son muy técnicos esta solución nos pareció al menos digna de tener en cuenta.

Los resultados obtenidos por este modelo de procesado son aceptables en imágenes sencillas donde las clasifican claramente como armas de fuego, pistolas, rifles de asalto o smartphones pero hay otros casos mas complejos y

especiales del dominio del problema cuya clasificación falla, debido a esto esta opción queda descartada ya que para obtener buenos resultados en la competición no podemos quedarnos con clasificar bien la mayoría de imágenes sencillas sino que deberíamos aprender también las complejas.

A continuación y a modo de ejemplo podemos ver algunas imágenes que se han clasificado y la salida dada por el modelo:


Figura 4.1: Imagen de pistola, clasificada como Assault Rifle.

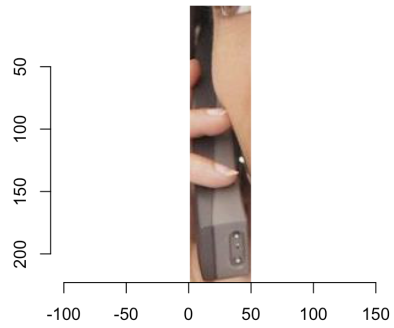


Figura 4.2: Imagen de teléfono, clasificada como Window Screen.

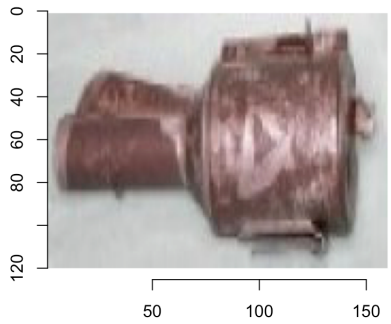


Figura 4.3: Imagen de arma, clasificada como Barril.

## 4.3. Fine Tunning

# Capítulo 5

# Conclusiones y vías futuras

En este capítulo final se estudian los resultados obtenidos a lo largo del trabajo y así como las vías futuras para aumentar aún más el accuracy. También se complementan las conclusiones que se han ido obteniendo a lo largo del trabajo.

## 5.1. Vías futuras

Respecto a las vías futuras, pese a que algunas ya han sido comentadas a lo largo de la memoria, cabe reunirlas en este último capítulo.

Principalmente, destacar la necesidad de disponer de máquinas más potentes o al menos con capacidad de procesado con GPU a la hora de procesar esta gran cantidad de imágenes con redes neuronales para poder ademas de mejorar los tiempos de ejecución, poder tener las imágenes con tamaños completos sin perdida de calidad.

Respecto a las mejoras en la implementación, ensembles de diversas topologías de redes neuronales con *fine tunning* y prevención del *overfitting* serían muy probablemente la mejor opción para atacar el problema. Estas soluciones han sido estudiadas en la literatura [6] pero no hemos conseguido implementarlas ya que son complejas y aún residen casi en su totalidad en el ámbito de investigación de las mismas.

## 5.2. Conclusiones finales

Como conclusión que aúna todas las demás podemos concluir que el problema era ciertamente complejo, y aunque permitía buenos resultados rápidamente mejorarlos iba subiendo en dificultad. Las redes neuronales y el deep learning pone a nuestra disposición una gran cantidad de técnicas (fine tunning, transfer learning y from scratch) que han sido descritas a lo largo de la memoria y las cuales se han intentado probar a modo experimental. Si sumamos esto a la necesidad de conseguir buenos resultados en la competición, creemos que el tiempo asignado para dicha práctica debería de haber sido algo más extenso lo que habría facilitado aún más la asimilación de los conceptos de la asignatura.

Respecto a las características del problema, muchas de las soluciones vistas no ofrecían buenos resultados llegando incluso algunas a no poder ejecutarse por falta de memoria. Por tanto, un problema que necesita de poder procesar imágenes, donde las herramientas disponibles para su procesado son tan limitadas (ordenadores personales), difícilmente podrá obtener buenos resultados, o al menos no todos los buenos resultados que podrían alcanzarse con equipos más potentes por lo que el sesgo entre soluciones que hayan podido hacer uso de GPU y otras que no, es elevado.

Por otro lado, cabe destacar la potencia de las librerías de deep learning, para este tipo de problemas. Con muy poco código y con conocimientos de programación básicos, se pueden lograr grandes resultados, aunque llegar a utilizar estas herramientas nivel experto puede llegar a ser una ardua labor.

Para finalizar las conclusiones, queremos profundizar un poco en lo trascendental del problema. Estamos trabajando con datos reales de identificación de armas y su comparación con otros objetos similares como puede ser el caso de smartphones. La potencia y uso de problemas reales del ámbito de la seguridad ha quedado constatada como es el caso del estudio de Olmos, Tabik y Herrera [7] que ha recibido prestigiosos premios internacionales y constata la necesidad de estas soluciones en la sociedad actual.

# Bibliografía


[1] Website de Kaggle https://www.kaggle.com

[2] Competicion en Kaggle. https://www.kaggle.com/c/pistolas-vs-smartphones-con-deep-learning

[3] Repositorio del proyecto. https://github.com/joseangeldiazg/guns-smartphones-classification

[4] Tutorial uso red pre-entrenada https://github.com/dmlc/mxnet/blob/master/R-package/vignettes/classifyRealImageWithPretrainedModel.Rmd

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proc. IEEE, November 1998.

[6] Jianjun Liu Shengping Xia Weidong Hu Wenxian Yu. Weights Updated Voting for Ensemble of Neural Networks Based Incremental Learning. *International Symposium on Neural Networks. Advances in Neural Networks* ISNN 2009 pp 661-669

[7] R Olmos, S Tabik, F Herrera. Automatic Handgun Detection Alarm in Videos Using Deep Learning *Neurocomputing*, 275, 66-72