*Clasificación automática de arquitecturas*

Isabel Arrans Vega  
*dpto. Ciencias de la Computación e Inteligencia Artificial*  
*Universidad de Sevilla*Sevilla, España  
[isaarrveg@alum.us.es](mailto:isaarrveg@alum.us.es) | isabelarrans@gmail.com

Matthew Bwye Lera  
*dpto. Ciencias de la Computación e Inteligencia Artificial*  
*Universidad de Sevilla*Sevilla, España  
[matbwyler@alum.us.es](mailto:matbwyler@alum.us.es) | xamtehwt@gmail.com

*Resumen*— Con este trabajo, pretendemos diseñar una inteligencia artificial capaz de clasificar rápidamente edificios de entre 25 clases distintas de estilos arquitectónicos, haciendo uso de redes neuronales.

Nuestras redes neuronales tienen distintos puntos fuertes y débiles; algunas son capaces de entrenarse muy rápidamente, pero tienen sobreajuste, lo que lo hace menos confiable para cualquier tipo de imagen, mientras que otras tardan mucho en entrenarse, pero tienen menos problemas con el sobreajuste, lo que lo hace más fiable.

Palabras Clave—Inteligencia Artificial, Redes Neuronales, Convolución, Sobreajuste, Conjunto de entrenamiento, Aprendizaje, Época

# Introducción

Este proyecto se realiza en un ámbito académico, en la Universidad de Sevilla. En el grado de Informática - Ingeniería del Software, asignatura de Inteligencia Artificial, los alumnos debemos agruparnos por parejas, solicitar algún proyecto y realizarlo y documentarlo. En este caso, realizaremos un trabajo para la clasificación de imágenes mediante redes neuronales convolucionales.

Dado nuestro conocimiento en la materia, se espera que nos documentemos en internet acerca de la resolución de esta tarea y que se referencie de forma apropiada. Las expectativas son que aprendamos a desarrollar software que haga uso de inteligencias artificiales y que adquiramos las competencias necesarias para poder hacer más trabajos de este tipo en el futuro.

En este trabajo, haremos uso de redes neuronales; esto son una serie de algoritmos y operaciones que tratan de simular una red neuronal humana, utilizando la teoría de grafos para representarse. El objetivo de nuestro código es ser capaz de entrenarse con un conjunto de entrenamiento y luego clasificar de forma apropiada una serie de imágenes de edificios entre 25 categorías que representan los estilos arquitectónicos. Proporcionaremos una serie de redes neuronales con parecidos entre sí y distinciones entre sí, esperando que una buena variedad de formas de análisis.

Como nuestras redes neuronales trabajan sobre análisis de imágenes, deben hacer uso de operaciones de convolución, que facilitan el análisis y la comparativa entre imágenes. Por este motivo, las redes neuronales que diseñemos recibirán el nombre de redes neuronales convolucionales.

# Preliminares

En esta sección haremos una breve introducción de las técnicas que hemos empleado, y también mencionaremos una serie de trabajos relacionados que nos han ayudado a alcanzar las soluciones que hemos propuesto.

## Métodos empleados

Los métodos empleados son los siguientes:

* Redes neuronales convolucionales: Se trata de un método que recibe parámetros de entrada, normalmente en forma de imágenes en caso de ser una red neuronal convolucional, luego estos parámetros influyen sobre una serie de operaciones que se influyen las unas a las otras, que forman las capas intermedias, y, por último, recibimos una serie de parámetros de salida, que nos indicarán la respuesta del algoritmo. Este método se caracteriza por tener capacidad de aprendizaje dado un conjunto de entrenamiento, que le permite luego saber clasificar cualquier dato de entrada basándose en su aprendizaje previo.
* Convolución: Es un método de filtrado de imagen, que va a facilitar a nuestro algoritmo la clasificación de imágenes. Se aplicará este algoritmo varias veces para hacer distintos análisis de las imágenes.
* Aumento de datos: Un inconveniente típico de este tipo de proyectos es el sobreajuste, es decir, que el algoritmo clasifique muy bien lo que recibe del conjunto de entrenamiento, pero no consigue clasificar bien elementos externos a él. Para mejorar esta situación, decidimos aumentar el número de datos que compara. La forma más fácil que encontramos para esto fue replicar y rotar las imágenes que el algoritmo analiza, y que analice también estas versiones. El principal inconveniente de hacer esto es que el entrenamiento es más largo, ya que tiene que trabajar con más imágenes.
* Dropout: Similar al método explicado antes, este se utiliza para reducir el sobreajuste. Este método lo que hace es descartar algunas de las imágenes en cada época de entrenamiento, que ayuda a hacer un análisis menos basado en el conjunto de entrenamiento.
* Autotune:

## Trabajos Relacionados

Aquí recorreremos un poco los distintos trabajos que hemos encontrado que nos han sido de ayuda.

* Documentación de la asignatura de Inteligencia Artificial.
* Práctica 2 de la asignatura de Inteligencia Artificial.
* Tutorial de clasificación de imágenes de la página principal de Tensorflow.
* Tutorial de clasificación de imágenes de la página principal de Keras.
* Documentación de los métodos de Tensorflow y Keras.

# Metodología

Al comienzo de nuestro trabajo, partimos de la base que teníamos de haber realizado la Práctica 2 de la asignatura. Basándonos en los razonamientos que seguimos ahí, tratamos de construir una idea de la red que queríamos construir.

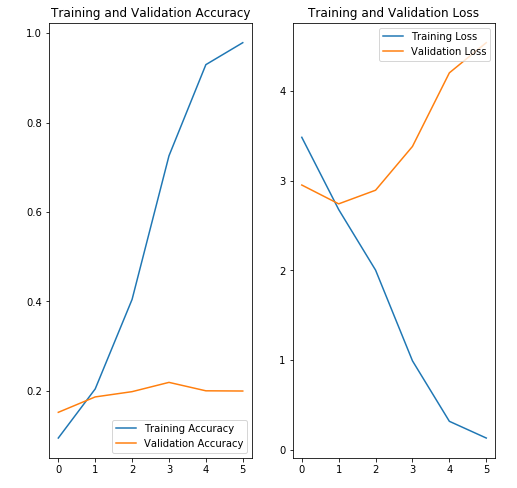
Por otro lado, leímos en profundidad lo que se esperaba que construyéramos. Juntando toda esta información, empezamos a pensar en qué redes debíamos construir.

En primera instancia decidimos que dos modelos claramente distintos podían ser uno que tuviera una salida (número decimal mayor o igual que 0 y menor o igual que 1) por cada clase, y la clase con mayor valor es la que el algoritmo determina que es la que le corresponde al edificio en cuestión, y el otro modelo en su lugar sólo tendría una salida (número decimal mayor o igual que 0 y menor o igual que 1), cuyo valor determinaría inmediatamente la clase del edificio (es decir, dividimos el intervalo [0, 1] en un segmento por cada clase, es decir, 25 segmentos, y el número de la salida estará en alguno de esos segmentos; la clase a la que pertenece el edificio será la clase asociada a ese segmento).

También debíamos construir dos modelos parecidos. Para ello, primero queríamos construir los dos modelos distintos y, una vez construidos, aplicar algún cambio que los hiciera distintos y nos permitiera analizar más datos. Sobre esta idea, vimos interesante construir más modelos si daban una variedad de resultados suficientemente interesantes, modificando más parámetros.

Una vez empezamos a trabajar, nos dimos cuenta de que era imposible de realizar el trabajo únicamente con el material aportado en las clases teóricas y prácticas, por lo que empezamos a documentarnos acerca de la librería de Keras y Tensorflow. Encontramos un par de tutoriales, y empezamos a replicar y analizar el código, además de leer las explicaciones que daban.

En poco tiempo ya teníamos unos resultados, simplemente adaptando el código que nos daban.



1. Gráfica que representa la efectividad del algoritmo para el conjunto de entrenamiento y para el conjunto de validación.

Como se ve en la Figura 1, nuestra primera versión no fue un éxito. Con sólo 6 épocas, alcanzaba una buena precisión para el conjunto de entrenamiento, pero no para el conjunto de validación. Esto quiere decir que cada vez clasificaba mejor los elementos del conjunto de entrenamiento, pero no pasaba igual con los del conjunto de validación, por lo cual, este algoritmo no funcionaría muy bien con imágenes de edificios que se encuentren fuera del conjunto de entrenamiento.

Para solucionar este problema, aplicamos los métodos de aumento de datos y de dropout. También modificamos la capa intermedia, que es la que realiza la convolución de las imágenes.

# Resultados

En esta sección se detallará tanto los experimentos realizados como los resultados conseguidos:

* Los experimentos realizados, indicando razonadamente la configuración empleada, qué se quiere determinar, y como se ha medido.
* Los resultados obtenidos en cada experimento, explicando en cada caso lo que se ha conseguido.
* Análisis de los resultados, haciendo comparativas y obteniendo conclusiones.

Se Puede hacer uso de tablas, como el ejemplo de la Table 1.

Table 1. Ejemplo de tabla

| Table Head | Table Column Head | | |
| --- | --- | --- | --- |
| Table column subhead | Subhead | Subhead |
| copy | More table copya |  |  |

# Conclusiones

Finalmente, se dedica la última sección para indicar las conclusiones obtenidas del trabajo. Se puede dedicar un párrafo para realizar un resumen sucinto del trabajo, con los experimentos y resultados. Seguidamente, uno o dos párrafos con conclusiones. Se suele dedicar un párrafo final con ideas de mejora y trabajo futuro.

##### Referencias

1. S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, 3rd ed, Pearson, 2010.
2. Y. LeCun, Y. Bengio, G. Hinton. “Deep Learning”, Nature, vol. 521, 2015, pp. 436-444.
3. Página web del curso IA de Ingeniería del Software. <https://www.cs.us.es/cursos/iais>. Consultada el 24/03/2018.