# Dataset

Los datos que usaremos se obtienen de un dataset suministrado por Kaggle.com[[1]](#footnote-1) el cual cuenta con imágenes en formato .jpg con dimensiones de 512 píxeles x 512 píxeles. Dichas imágenes vienen clasificadas con diferentes etiquetas que identifican los diferentes elementos presentes en los dibujos.

## Origen del dataset

El dataset almacenado en la web Kaggle.com no es el dataset original del que provienen las imágenes. La fuente primaria de las imágenes y sus metadatos es la página web danbooru.donmai.us/ que es una web a la que los diferentes usuarios pueden subir ilustraciones y añadirle diferentes etiquetas para facilitar su búsqueda a otros usuarios.

Las imágenes almacenadas en la web fueron almacenadas creando el dataset que proporciona Kaggle.com junto a sus metadatos. El dataset resultante contiene aproximadamente 300.000 imágenes obtenidas de los dataset Danbooru2017 y Nagadomi’s moeimouto face calificadas como no sexuales.[[2]](#footnote-2)

Más adelante estudiaremos cómo estos datos han sido tratados y cómo están almacenados en el dataset.

## Estructura del dataset

### Contenido del dataset

El dataset contiene 3 carpetas:

* *danboou-images*: En esta carpeta están almacenadas las diferentes ilustraciones ordenadas por carpetas donde el nombre de la carpeta hace referencia a los últimos cuatro dígitos del nombre de las imágenes almacenadas en ella. De esta forma tenemos 151 carpetas que almacenan imágenes con id’s cuyos últimos cuatro dígitos van desde el *0000* hasta el *0150*.
* *danbooru*-*metadata*: En esta carpeta tenemos los archivos de texto (con extensión .txt) correspondientes a los metadatos del dataset original. Dichos metadatos tienen formato JSON pero su estructura no es adecuada y más adelante pasaremos a tratarlos para poder operar con ellos correctamente.
* *moeimouto-faces:* Esta carpeta contiene ilustraciones de caras ordenadas en carpetas correspondientes a el personaje al que pertenece dicha cara. Durante el desarrollo del trabajo no haremos uso de esta carpeta puesto que no nos proporciona imágenes de formato normalizado además de que las imágenes que guarda son primeros planos de caras que no usaremos puesto que no representan ilustraciones completas. Por todo esto no podemos añadir dichas imágenes a nuestro dataset.

### Tratamiento de las imágenes originales

Las ilustraciones originalmente guardadas en la página danbooru.donmai.us son tratadas para ser almacenadas en imágenes de 512 píxeles x 512 píxeles.

Para producir esta transformación las imágenes son contraídas hasta que una sus dos dimensiones ocupen menos de 512 píxeles, una vez son reducidas, se rellenan los píxeles restantes con píxeles en color negro.



En la figura 1 podemos observar cómo la imagen original (izquierda) es tratada de manera que al reducir su tamaño se generan dos bandas negras en las partes superior e inferior debido a que la imagen original tiene mayor tamaño de ancho que de alto. La imagen generada (derecha) pasa a formar parte del dataset.

Por otra parte, puede darse el caso en que una imagen originalmente no ocupe más de 512 píxeles en ninguna de sus dos dimensiones, en este caso la imagen no se extiende, sino que se rellenan los huecos sin imagen con píxeles negros.



En la figura 2 podemos observar que para completar los 512 píxeles x 512 píxeles se rellena el espacio restante dando lugar a un recuadro negro que rodea a la ilustración.

¿Meter aquí una reflexión de como esto puede afectar al resultado final o cuando obtengamos resultados volver a esta parte para justificarlo?

# Archivos de metadatos

## Tratamiento de metadatos para obtener datos JSON

Hacer un plan de tratamiento de metadatos como si todo estuviese planeado o explicarlo cronologicamente y luego hacer un esquema de lo que he hecho

Antes de poder analizar los archivos de metadatos de las ilustraciones, tenemos que pasarlos a un formato que nos resulte cómodo para empezar a analizar sus diferentes campos debido a que los archivos de metadatos están estructurados de manera que cada línea del fichero la ocupa un objeto JSON con los metadatos relativos a una ilustración. Nos encontramos que los diferentes objetos JSON están almacenados de manera individual, esto provoca que el fichero no tenga un formato válido para un archivo de datos JSON y por tanto no pueda ser tratado.

Para que los diferentes archivos de metadatos tengan un formato de JSON válido debemos crear un array de objetos JSON con todas las líneas de cada uno de los ficheros de metadatos. Para ello con un editor de texto sustituimos los saltos de línea por una coma y añadimos como primer carácter del texto el símbolo *[* y como carácter final *]*, de esta manera creamos para cada uno de los ficheros una estructura adecuada de datos JSON que podemos tratar con facilidad.

Una vez obtenidos los datos en formato JSON válido los almacenamos en la carpeta *MetadataRaw.*

Quizás no es el método más ortodoxo para hacer esto, deberia omitir que hice esto? Suena un poco raro omitirlo

## Análisis de metadatos

Antes de poder estudiar los diferentes campos que se almacenan en nuestros archivos de texto les damos un formato que nos resulte más sencillo de leer y analizar a través del script *formatJSON.py* obteniendo como resultados unos metadatos que pasamos a guardar en la carpeta *MetadataStructured*.

Ahora con los archivos bien almacenados podemos pasar a estudiar los diferentes campos de los datos con el fin de eliminar datos que nos resultan irrelevantes para el objetivo que perseguimos:

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre del ID | Contenido | Eliminar | Motivo |
| id | Número del nombre de la imagen que guarda sus datos. | ❌ |  |
| created\_at | Fecha de creación de la ilustración. | ✔️ | La fecha de creación de la imagen no identifica ningún elemento de la imagen |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

¿Debo justificar todos los campos?

1. <https://www.kaggle.com/mylesoneill/tagged-anime-illustrations/> [↑](#footnote-ref-1)
2. <https://www.gwern.net/Danbooru2019#kaggle> [↑](#footnote-ref-2)