Images Preprocessing

In the following file, we will go over the image preprocessing steps and the steps we'll be taking before structuring the preprocessing method.

```
from datasets import load_dataset

train_dataset = load_dataset("Az-r-ow/chest_xray", split="train")

from utils.helpers import format_dataset

labels = train_dataset.features["label"].names

# Converting to pandas will encode the images to bytestrings
train_data = format_dataset(train_dataset, labels)
train_data.head()
```

	image	label	true_label
0	<PIL.JpegImagePlugin.JpegImageFile image mode=...	0	NORMAL
1	<PIL.JpegImagePlugin.JpegImageFile image mode=...	0	NORMAL
2	<PIL.JpegImagePlugin.JpegImageFile image mode=...	0	NORMAL
3	<PIL.JpegImagePlugin.JpegImageFile image mode=...	0	NORMAL
4	<PIL.JpegImagePlugin.JpegImageFile image mode=...	0	NORMAL

In a first step, we will be transforming the images to greyscale, resizing them to (100, 100) then normalize their values and then accumulate them on a canvas to form a pixel heatmap.

```
import numpy as np

img_size = (224, 224)


def get_heatmap(img, heatmap):
    # Converting the image to a numpy array
    resized_img = np.array(img)
    # Normalizing the image (to have values between 0 and 1)
    resized_img = resized_img / 255
    heatmap = np.add(heatmap, resized_img)

from utils.helpers import resize_grayscale

formatted_images = (
    train_data["image"]
    .map(lambda x: resize_grayscale(x, img_size))
    .reset_index(drop=True)
)
```

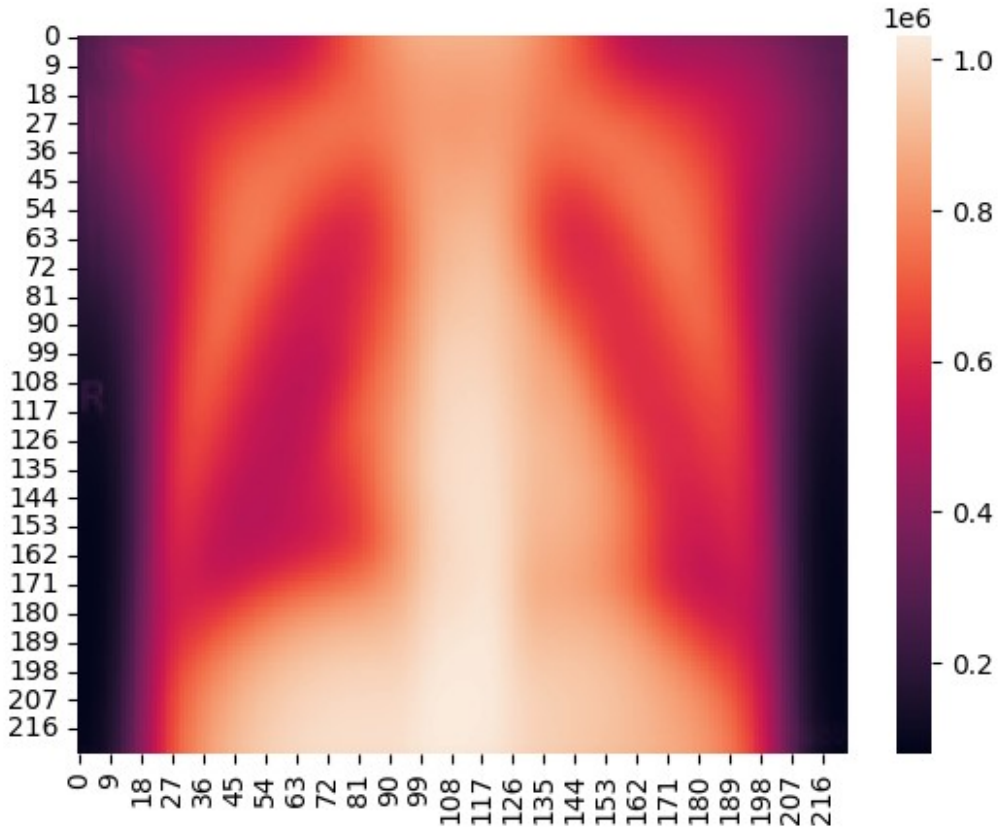
```
import numpy as np
import seaborn as sns

heatmap = np.zeros(img_size, dtype=np.float64)

for img in formatted_images:
    img = np.array(img)
    heatmap = np.add(heatmap, img)

sns.heatmap(heatmap)
```

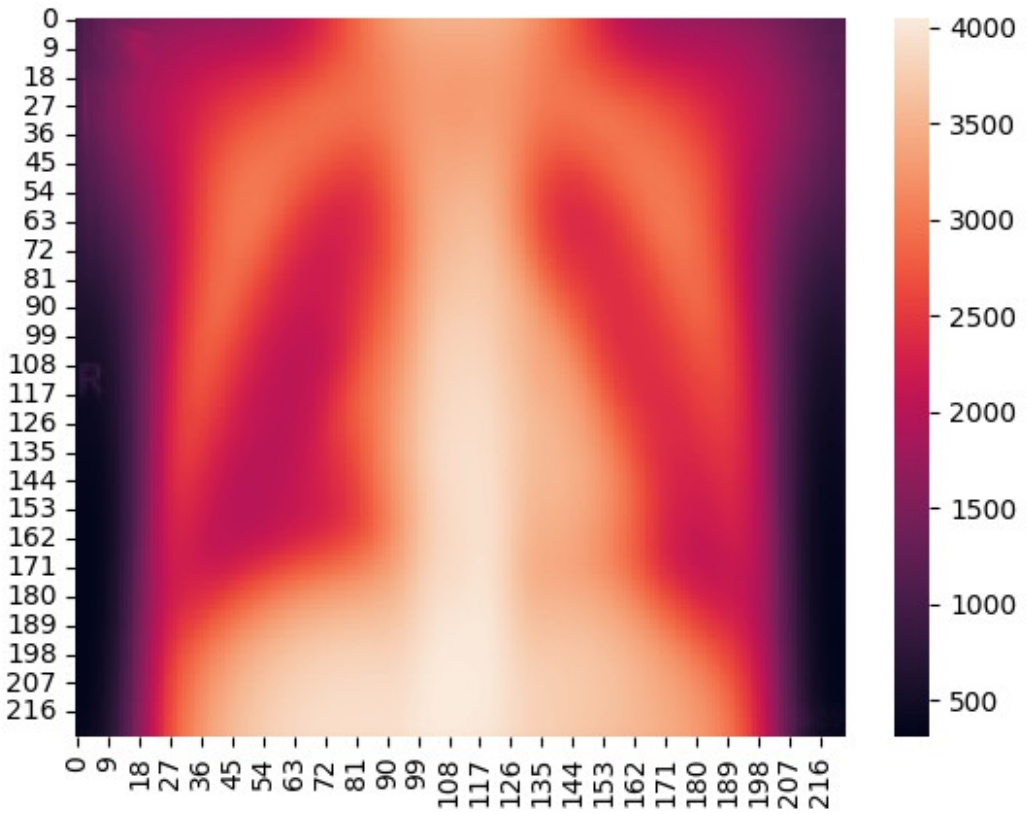
```
<Axes: >
```




```
filtered_heatmap = heatmap
filtered_heatmap = filtered_heatmap / 255

sns.heatmap(filtered_heatmap)
```

```
<Axes: >
```



We can notice that we have a very weak pixel density on the sides as well as on the shoulders region. Therefore, we can apply a mask on each image that will nullify these regions.

```
import cv2
from matplotlib import pyplot as plt

height, width = img_size[:2]

sideoffset = 7
topoffset = 6

mask = np.zeros(heatmap.shape[:2])
original_points = np.array(
    [
        [32, topoffset],
        [11, 28],
        [sideoffset, 100],
        [100 - sideoffset, 100],
        [100 - 11, 28],
        [67, topoffset],
    ],
    np.int32,
)

# Scale the points to the new image size
```

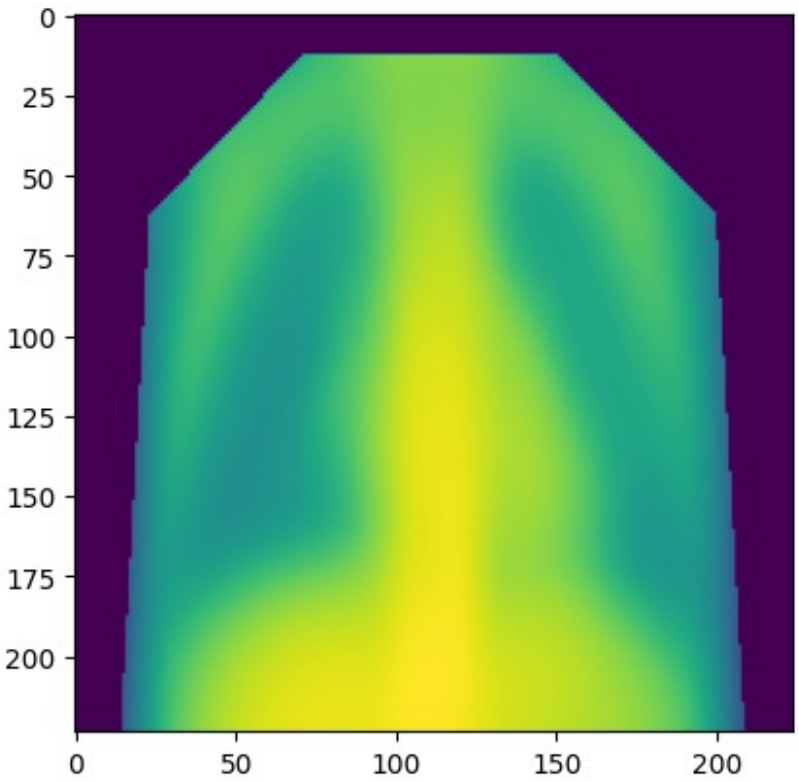
```
scale_x = width / 100
scale_y = height / 100
scaled_points = original_points * [scale_x, scale_y]

# Reshape the points for cv2.fillPoly
scaled_points = scaled_points.reshape((-1, 1, 2))

# Create a mask
mask = np.zeros((height, width), dtype=np.uint8)
cv2.fillPoly(mask, [scaled_points.astype(np.int32)], color=1)

# Apply the mask to the heatmap
masked = mask * heatmap

# Display the masked heatmap
plt.imshow(masked)
plt.show()
```




```
# Saving the mask in a npz file
np.save("../datasets/mask", mask)
```