**Introduction to image segmentation**

**Image segmentation**

Image segmentation is a computer vision task that involves partitioning the image into multiple segments, or areas withing the image, on the pixel level. This means that each pixel in an image is assigned to a particular segment. There exist three types of segmentation: semantic, instance, and panoptic and each of them requires a different model architecture. Let's discuss them one by one.

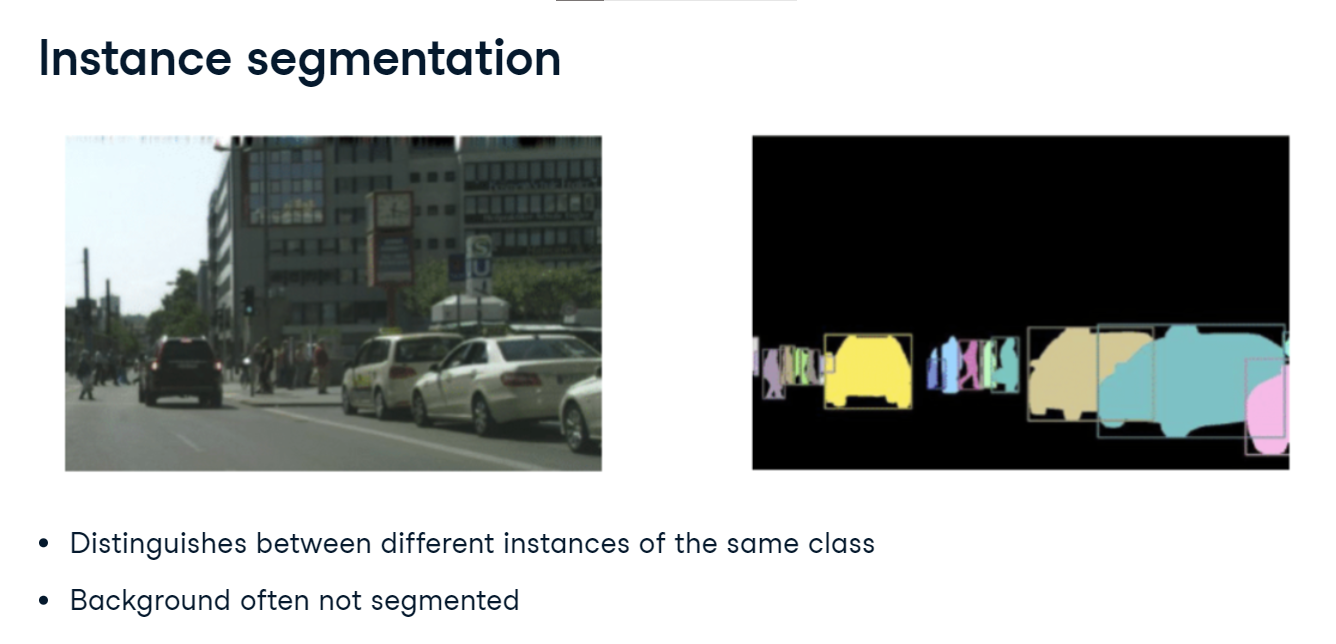
**Semantic segmentation**

In semantic segmentation, each pixel in the image is classified into a predefined class or category. All pixels belonging to the same class are treated equally, and there is no distinction between different instances of the same class. In a street scene, all pixels belonging to cars are marked as dark blue, all pixels belonging to roads are marked as purple and so on, without distinguishing between individual cars or road sections.



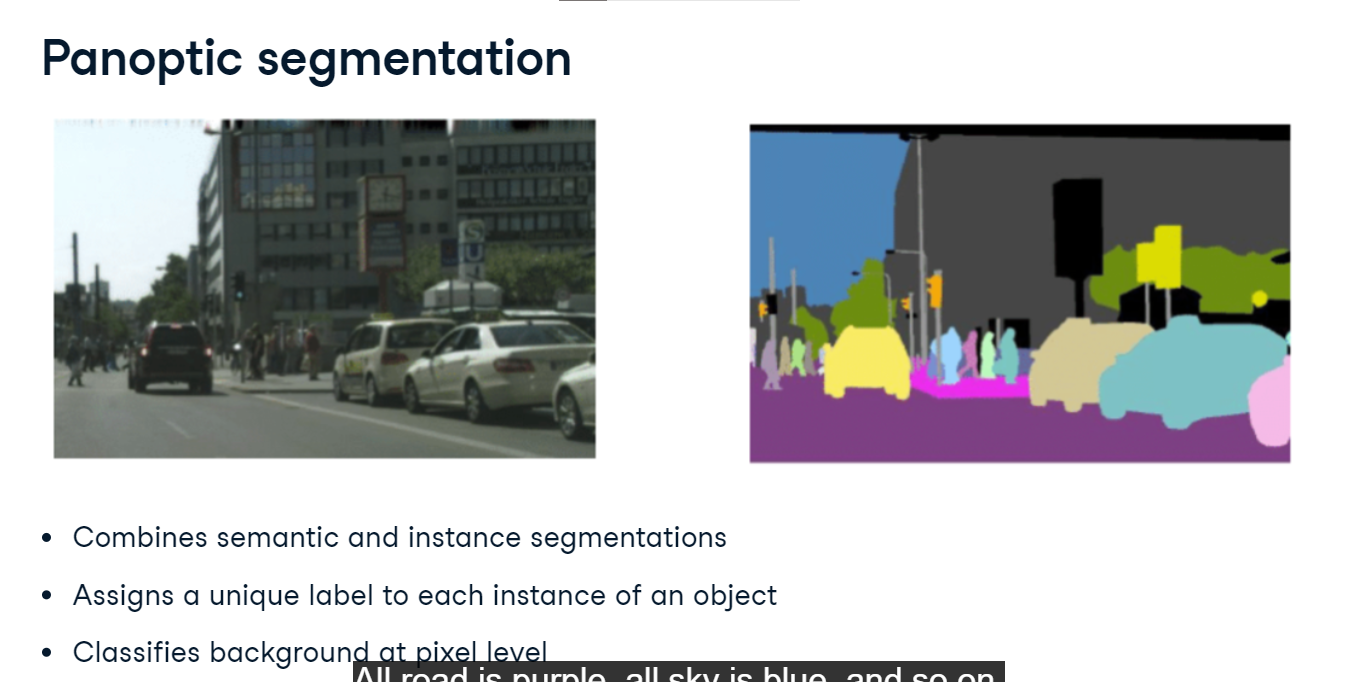
**Instance segmentation**

Instance segmentation goes a step further than semantic segmentation by not only classifying each pixel but also distinguishing between different instances of the same class. In the same street scene, each car would be assigned a unique label shown with a unique color, so that different cars can be differentiated from each other, even though they belong to the same class "car". Since the primary focus of instance segmentation is on identifying and segmenting individual object instances in the image, the background such as road or sky is typically not segmented.



**Panoptic segmentation**

Panoptic segmentation combines the concepts of semantic segmentation and instance segmentation. It assigns a unique label to each instance of an object while also classifying background regions (such as sky, road, or grass) at the pixel level. In the street scene, each car would get a unique label like in instance segmentation (each car is shown in different color). At the same time, areas like the road, sky, and trees are labeled at the pixel level without instance differentiation like in semantic segmentation. All road is purple, all sky is blue, and so on.



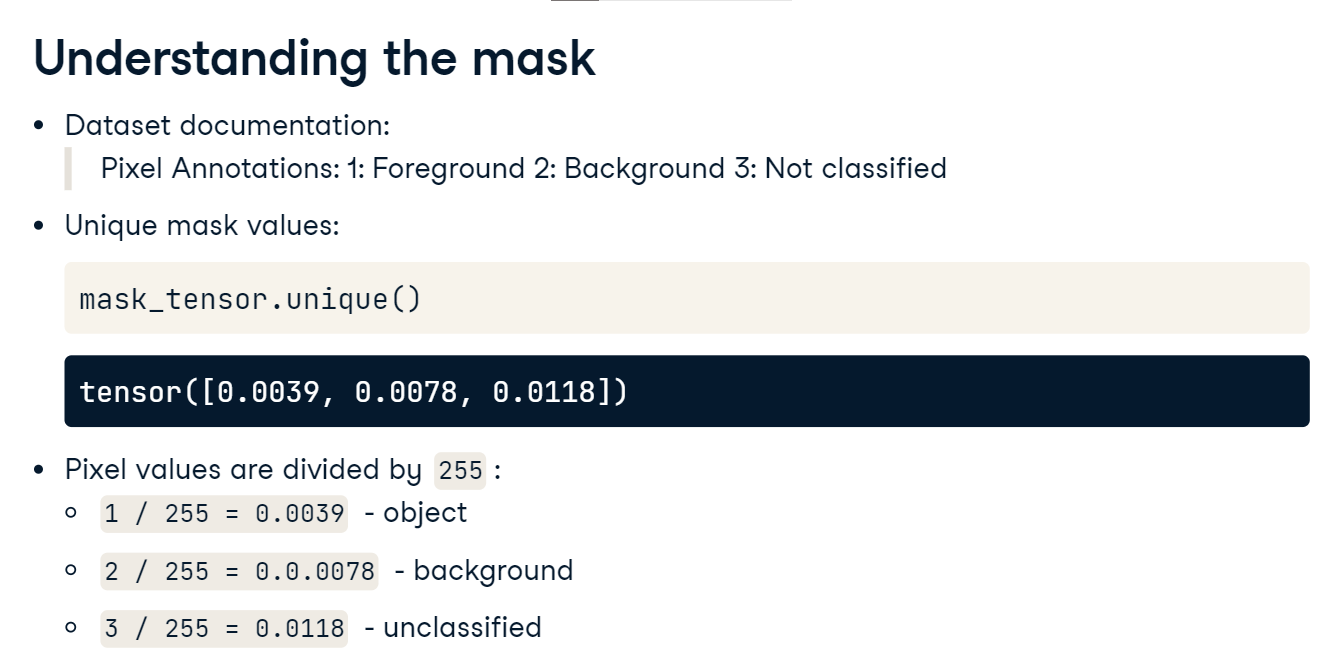
**Data annotations**

Let's take a look at data annotations for segmentation tasks. We load two image files. image is the picture of this British Shorthair cat sitting on a sofa. mask is the corresponding data annotation. The mask tells us which pixels are part of the cat, and which are not. Let's convert both PIL images to PyTorch tensors and print their shapes. The image is 333 by 500 pixels and has three color channels. The corresponding annotation is of the same height and width, but has only value for each pixel, describing its segment.



**Understanding the mask**

In the dataset documentation, we read that the annotations can only take three values: 1 for the object, 2 for the background, and 3 for unclassified. But when we print the unique mask values, we see three different numbers! This is because the ToTensor transform has divided the pixel values by 255. In our case, 1 over 255 which equals 0.0039 denotes the foreground, or the object of interest. A similar calculation is done for background and unclassified.



**Creating a binary mask**

Let's create a binary mask, where 1 corresponds to the object and 0 to everything else. We will use the torch-dot-where function to do so. It takes three arguments. First, the condition to check: whether the pixel value represents the object. Then, the value to use when the condition is met, here 1, followed by the value to use otherwise, here 0. Let's take a look at our binary mask. We convert the mask tensor back to a PIL image and display it. The cat's shape is clearly visible!



**Segmenting the object**

Now, let's segment our cat out of the picture. To create the object tensor, we multiply the image with the binary mask. Next, we proceed just like we did with the mask: we transform the object to a PIL image and display it. The cat has been segmented out and the sofa in the background is gone!

