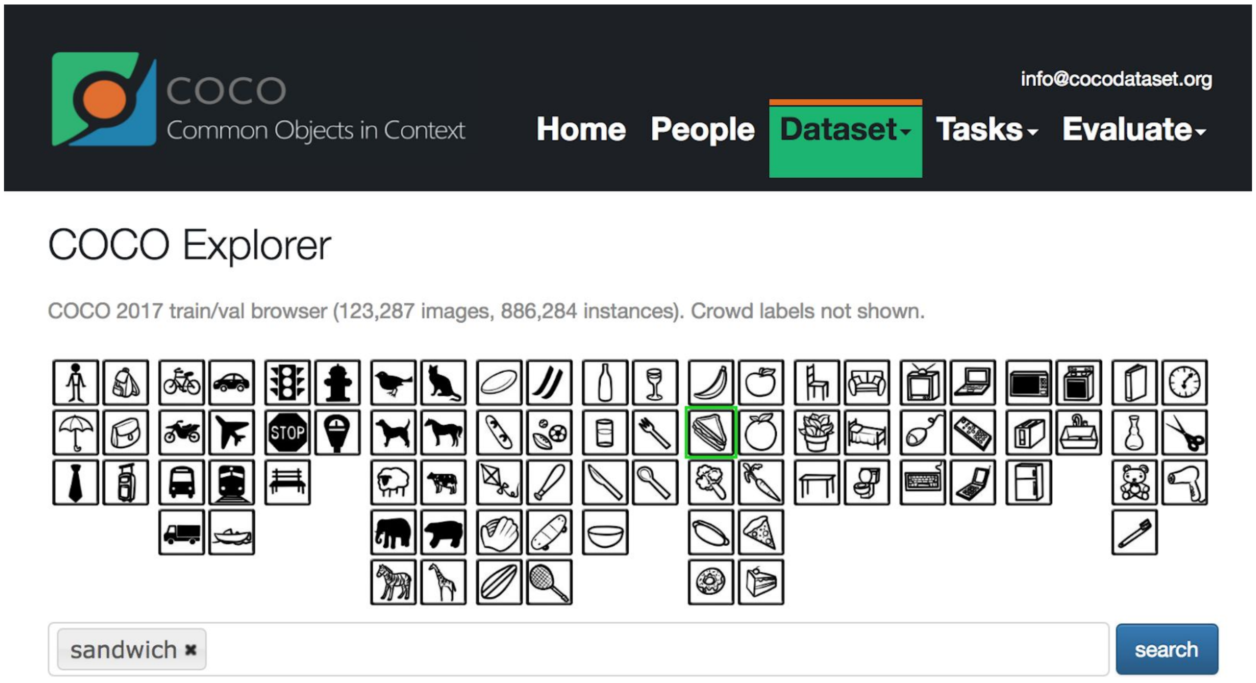
COCO Dataset

The COCO dataset is one of the largest, publicly available image datasets and it is meant to represent realistic scenes. What I mean by this is that COCO does not overly pre-process images, instead these images come in a variety of shapes with a variety of objects and environment/lighting conditions that closely represent what you might get if you compiled images from many different cameras around the world.

To explore the dataset, you can check out the dataset website.

Explore

Click on the explore tab and you should see a search bar that looks like the image below. Try selecting an object by it's icon and clicking search!



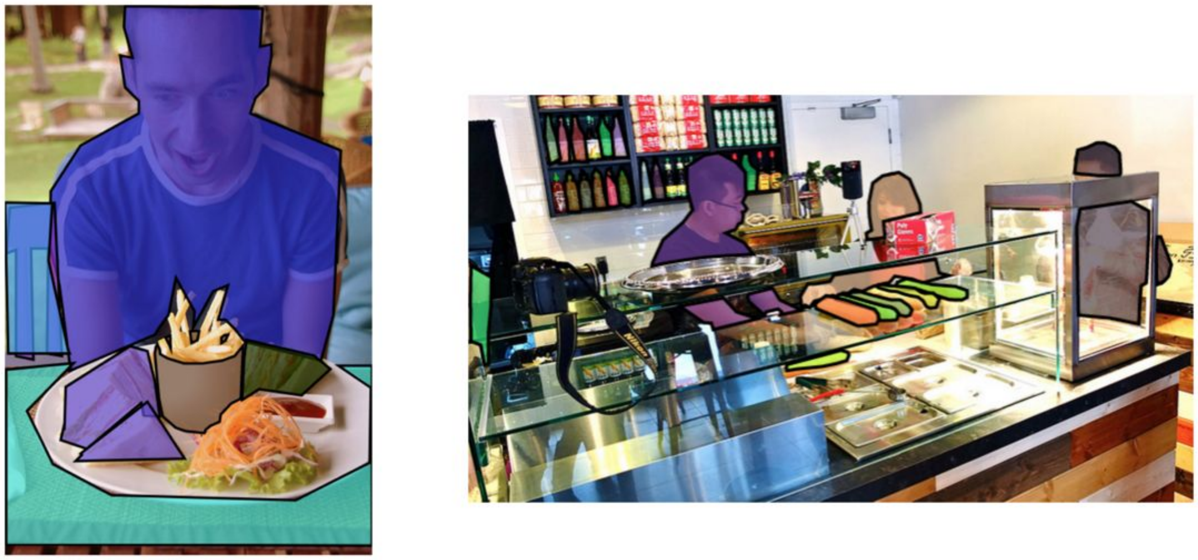
COCO Explorer

COCO 2017 train/val browser (123,287 images, 886,284 instances). Crowd labels not shown.

sandwich ✖

A sandwich is selected by icon.

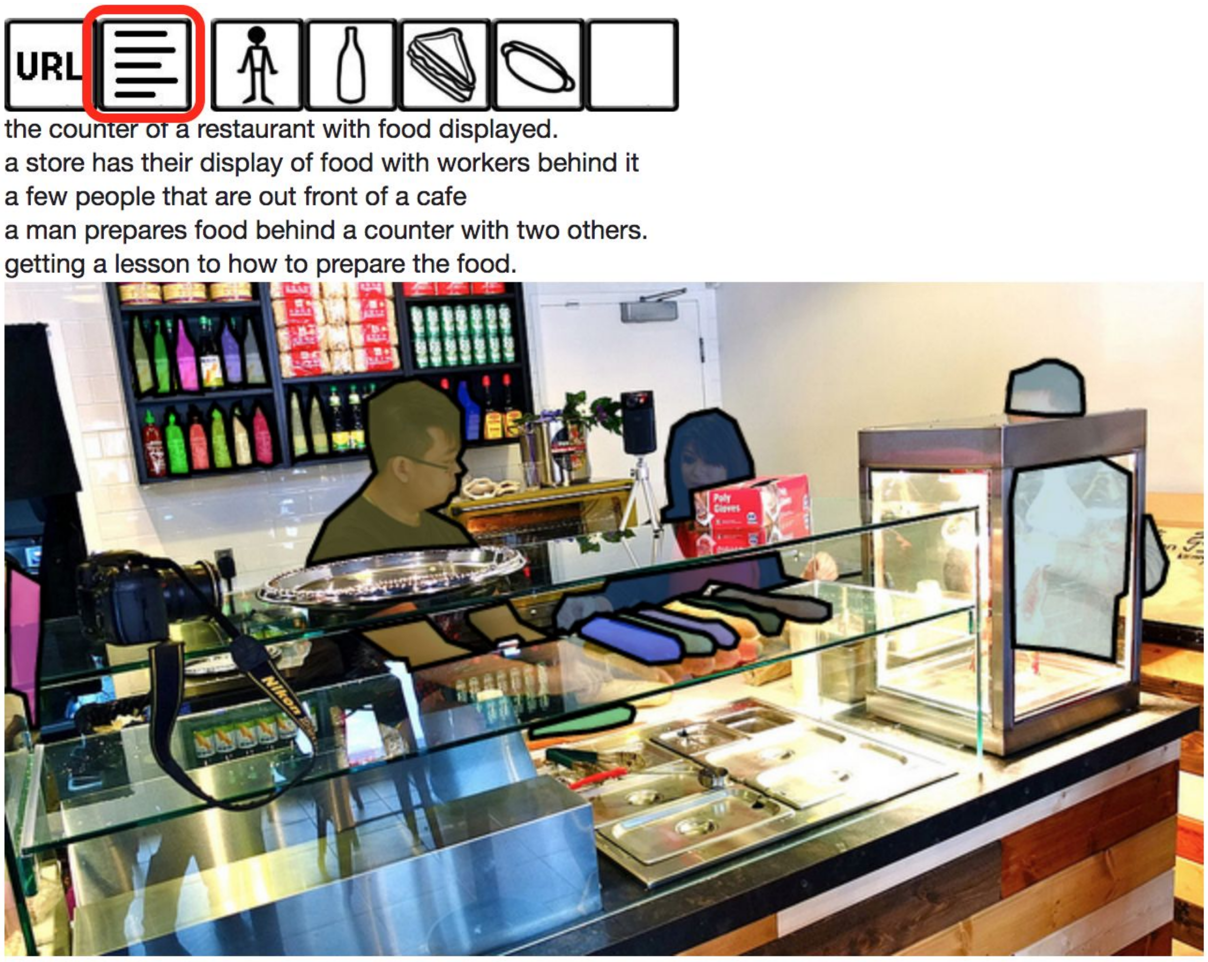
You can select or deselect multiple objects by clicking on their corresponding icon. Below are some examples for what a `sandwich` search turned up! You can see that the initial results show colored overlays over objects like sandwiches and people and the objects come in different sizes and orientations.



COCO sandwich detections

Captions

COCO is a richly labeled dataset; it comes with class labels, labels for segments of an image, *and* a set of captions for a given image. To see the captions for an image, select the text icon that is above the image in a toolbar. Click on the other options and see what the result is.


URL
the counter of a restaurant with food displayed.
a store has their display of food with workers behind it
a few people that are out front of a cafe
a man prepares food behind a counter with two others.
getting a lesson to how to prepare the food.

Example captions for an image of people at a sandwich counter.

When we actually train our model to generate captions, we'll be using these images as input and sampling *one* caption from a set of captions for each image to train on.

Supporting Materials

Creating COCO, paper

Words to Vectors

At this point, we know that you cannot directly feed words into an LSTM and expect it to be able to train or produce the correct output. These words first must be turned into a numerical representation so that a network can use normal loss functions and optimizers to calculate how "close" a predicted word and ground truth word (from a known, training caption) are. So, we typically turn a sequence of words into a sequence of numerical values; a vector of numbers where each number maps to a specific word in our vocabulary.

To process words and create a vocabulary, we'll be using the Python text processing toolkit: NLTK.
Reference:

- `nltk.tokenize` package: http://www.nltk.org/api/nltk.tokenize.html