Creating bounding boxes for ImageNet data

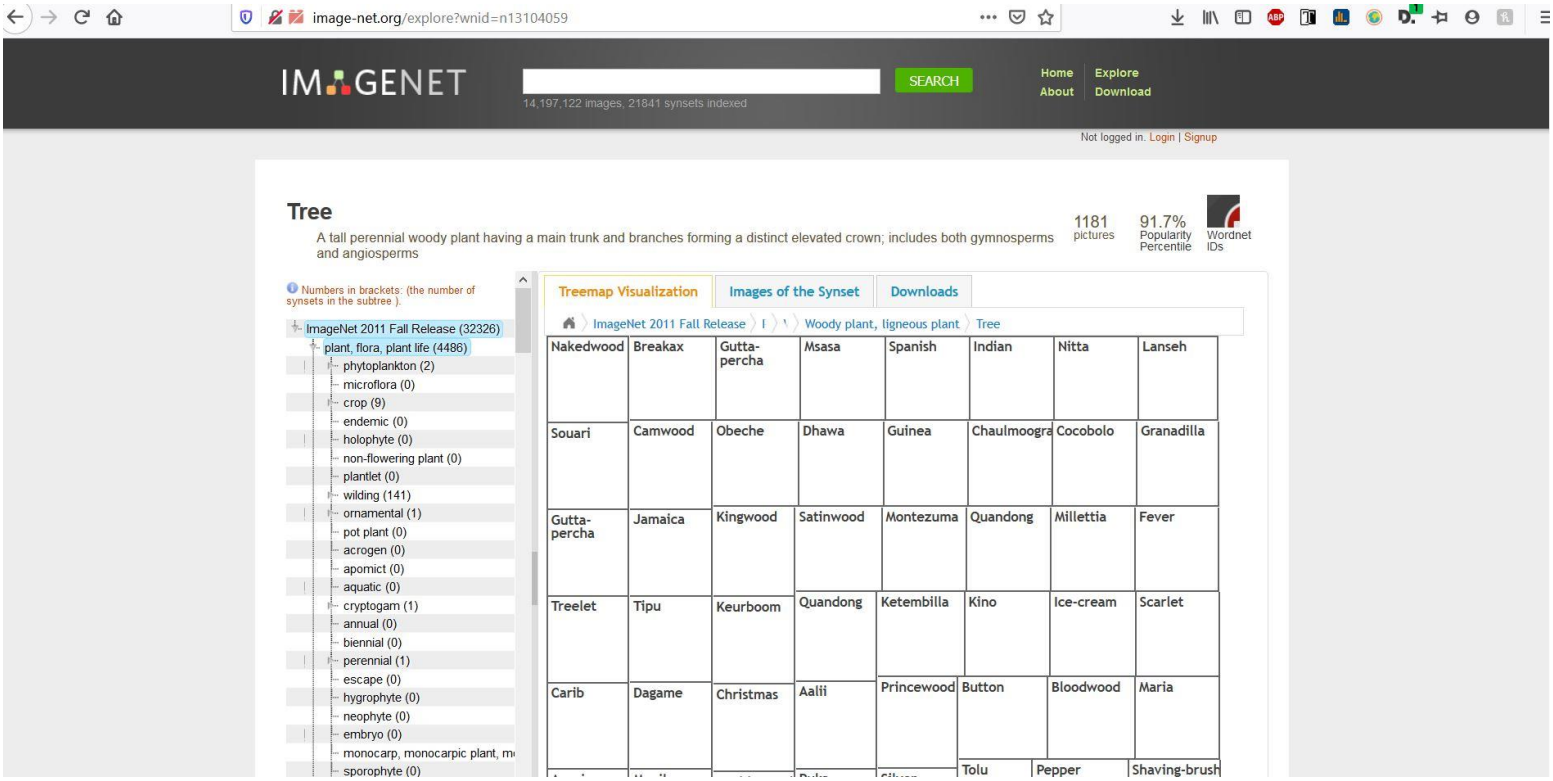
Access

We have access to annotated ImageNet data available on image-net.org. The credentials for that are:

Username: shubhomb

Access key: "a057c7178f6ba153a61e3e2ec546ea3b32a8d463"

Please do not share these with any external users.



Dataset Overview

The dataset consists of many images, most of which are not useful for Greenstand. However, you can explore at image-net.org to find images of useful things. Furthermore, to identify the hierarchical relationships between entities, (ex. “Golden Retriever”, “dog” “animal”), each of these resolutions is given a unique wnid. Visit image-net.org for more information.

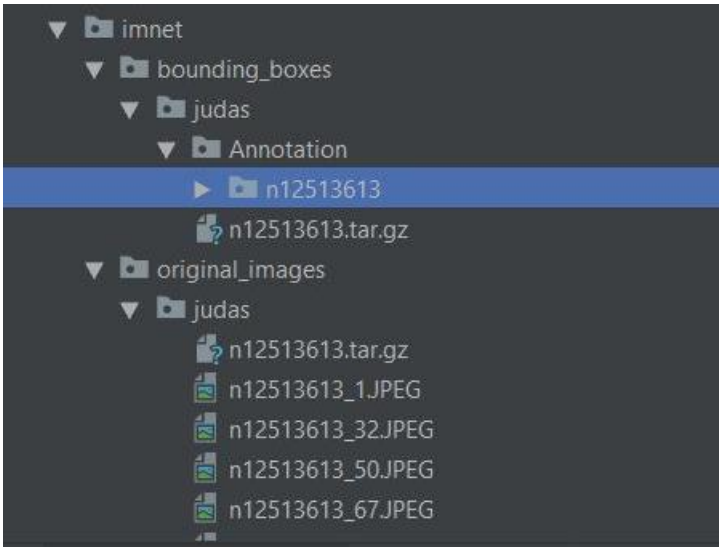
Current Data Extraction

The wnid field is the only thing needed to for data extraction. We have found that searching for images using the search bar at the top of the page yields the resolution of entity that works with the ImageNet API:

- For example, the wnid for tree yields unopenable files, but when searching tree in the search bar and download “judas” trees, it works fine

In the Github repo, see the imnet directory.

In download_images_and_boxes.py, a dictionary called synsets contains key-value pairs of names and wnids, and downloads images and annotations from ImageNet’s API. Change the `datadir` and `bbdir` arguments to specify the directory where raw images and annotation XML files respectively need to be downloaded. This should automatically untar the downloaded files and create a directory structure like:



The highlighted directory contains the XML annotations for n12513613, the Judas tree

The same `datadir` and `bbdir` are used in annotation_parser.py. The XML files defined by ImageNet provide metadata about images, including bounding boxes in many (but not all) cases. However, one image can contain images of multiple wnids, so we must specify which wnids are relevant for our use case, which is done in `names_wanted`.

If the directory structure is kept intact, the function `find_associated_annotation` will find the XML file that matches a given JPEG file, which is done entirely by naming convention. The function `extract_bounding_box` given an annotation XML path and the `names_wanted` parses the XML to find all boxes pertaining to wnids specified in

`names_wanted`. Then, the `draw_bounding_box_on_image` displays the bounding boxes outputted onto the image. Here are some examples from the Judas tree: