Software and Data Engineering coding test

Caution

The data you have been provided is not for general use. It is to be used for this coding test only and must be deleted after you have submitted it. Do not share the data with anyone.

The data

You are been given a bunch of tar files consisting of imagery and the corresponding ground truth about the presence of specified attributes (eg roof, swimming). A good way to interpret these ground truth is to see them as pixel-level annotation for provided images for a specified class. These truths could potentially be used in semantic segmentation related AI/ML tasks.

Download your data from https://www.dropbox.com/s/dd4dhs9jxlnjsgm/AI_Data_Software_Engineering_Question.zip?dl=0.
It is ~2.36 GB. Password is ‘a1s2d3f4'

The dataset consists of:

1. A CSV file containing metadata about the data
2. Bunch of tar.gz files containing images (jpeg files), and masks (hdf5 files).

Each of these tar files represents a labeling job completed by a labeler on an image (with location id given by `loc_id`], and image capture date given by `date` for the specified attributes (aka `classes`). As the labelers work on capturing ground truth, they provide pixel-level information annotating each pixel with either:

- 0 for the attribute is absent
- 1 for the attribute is present
- -1 for unsure if the attribute is present or not.

CSV
This file contains information about which classes are present for location given by `loc_id` and `date` and `classes` as described above. The Label file is given as <job_index><loc_id><date>.tar.gz example `1_20681_2017-11-10.tar.gz`

The tar file

Images
The images are all aerial photographs. They may not all be the same size, but most of them will be 896x896.
Also, an image capture on a day at a specified location is considered unique.

labels/masks

Masks are stored in hdf5 files as key-value pairs with the key being the id of some class. In this data set key 2 correspondings to the roof and key 3 correspondings to solar panels.

The masks (hdf5 files) all match the dimension of their corresponding image and represent labels for pixels in the image. In other words, `mask[k][i,j]` is a label (for some class) for pixels [i,j] in the image for attribute k.

Masks have value 0 (not present) or -1 (unknown/no label), or >= 1 (present).
Note that many of the values in the mask are -1 this is expected.

Note that not all images have labels, and even if labels are available, some images have labels for the only roof or only solar (i.e. not both). Many images do have labels for both though.

Data quality
The data is derived and synthesized from real data, with everything that that implies: there may be bad (fuzzy) images, and there will be some mistakes/noise in the labels. That said, we are not out to trick you: the data is in general of reasonable quality, so do not expect pathologically bad data.

Your task

1. Your task is to combine these all jobs that were done on the same image and build a holistic view of that image for each unique attribute. If 10 jobs were completed on image 1 with 8 jobs on attributes [2, 3] and the remaining 2 jobs just on attribute [3] then generate a final dataset that consists of final mask for [2,3] that represents the majority.
2. Ensure that your solution is capable of handling more jobs being added to the same location and date pair with existing and new attributes. Because in principle, It is possible to have many jobs done on the same location for new or repeated attributes (`classes`). The jobs keep coming in and dataset volume continues to grow.
You are required to design your solution to cater to this growing scenario even though your dataset is static.

Please make note of the major choices you make (tools, techniques, designs & assumptions). Please include all your code and notes related to this solution in your response. You can choose to deliver your solution as an archive or private share of GitHub or alike solutions.
Please note that this coding test is confidential and you are required to not share either the data or problem statement with anyone else but Nearmap. You are also required to keep your solution confidential and not share it with anyone outside Nearmap.

Please feel free to say a few words about what you would do to improve, scale, optimize your solution if you had (lots) more time and resources (i.e. if this was your job rather than a take-home test).
We would like to see some code that shows your outputs and calculates some performance metrics.

Bonus point task
1. If you have used frameworks then write a brief note to describe how your solution will be deployed to the cloud.

Important Notes

Whilst we welcome your choices in terms of toolsets and frameworks, we prefer that language choice is either python or golang.

Please note that we do not expect a polished, production-ready solution. Having said, we expect to see correctness and your approach to problem-solving.

This test is not time-bound. You are free to take your time in drafting your solution. We have noted that about 4-6 hours is a reasonable estimate to complete the main task. The bonus point question is optional.