Large scale object detection dataset retrieval

Team 4 - DB2

生醫電資所碩二	余銘仁	r09945024
地質系大五	周柏頤	b06204008
資工所碩一	徐浩宇	r10922195
電機系大四	王譽凱	b07901122

Outline

Motivation

Internet is popular. Images are everywhere. Added with extensive online annotation tools and services (like AWS), we can get images and their labels easily. Moreover, machine learning (ML) and deep learning (DL) have become trendy, what comes with it is the need for abundant training data. Therefore, whether we deal with those data fast enough becomes a critical challenge for ML/DL based application.

Let’s say we have an *object detection dataset* and our task is to *retrieve cropped images of a specific class*. Intuitively, we can load chunks of image data and bounding box data, then we perform cropping afterwards. However, it would cause memory issue by loading lots of data into client-side memory (RAM). Also, it is not time-efficient due to high memory transfer from DB (disk) to client (RAM).

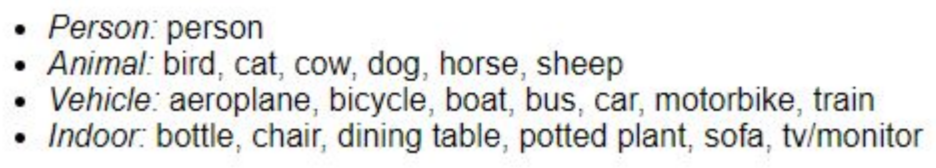
Due to above reasons, we aim to develop an extension function in MySQL to allow operations (compare, crop…) all running on the server side, return only the result (i.e. cropped images) to users. With this design, we can solve memory and time issues, and make it easier for users to get the data by simply calling a statement.

Objectives

1. Build an **object detection database** with targeted dataset
2. Develop an **extension function in MySQL** to allow operations all running on the server side, then return the result to users
   a. crop(*image*, *bbox*): crop an image *image* by its bounding box *bbox*
   b. similar(*embedA*, *embedB*): compute the similarity of embeddings from two images *embedA* and *embedB*
3. Retrieve cropped image data from database by **calling SQL commands on the client side** by **specifying either a target class or an target image**

Dataset & Model

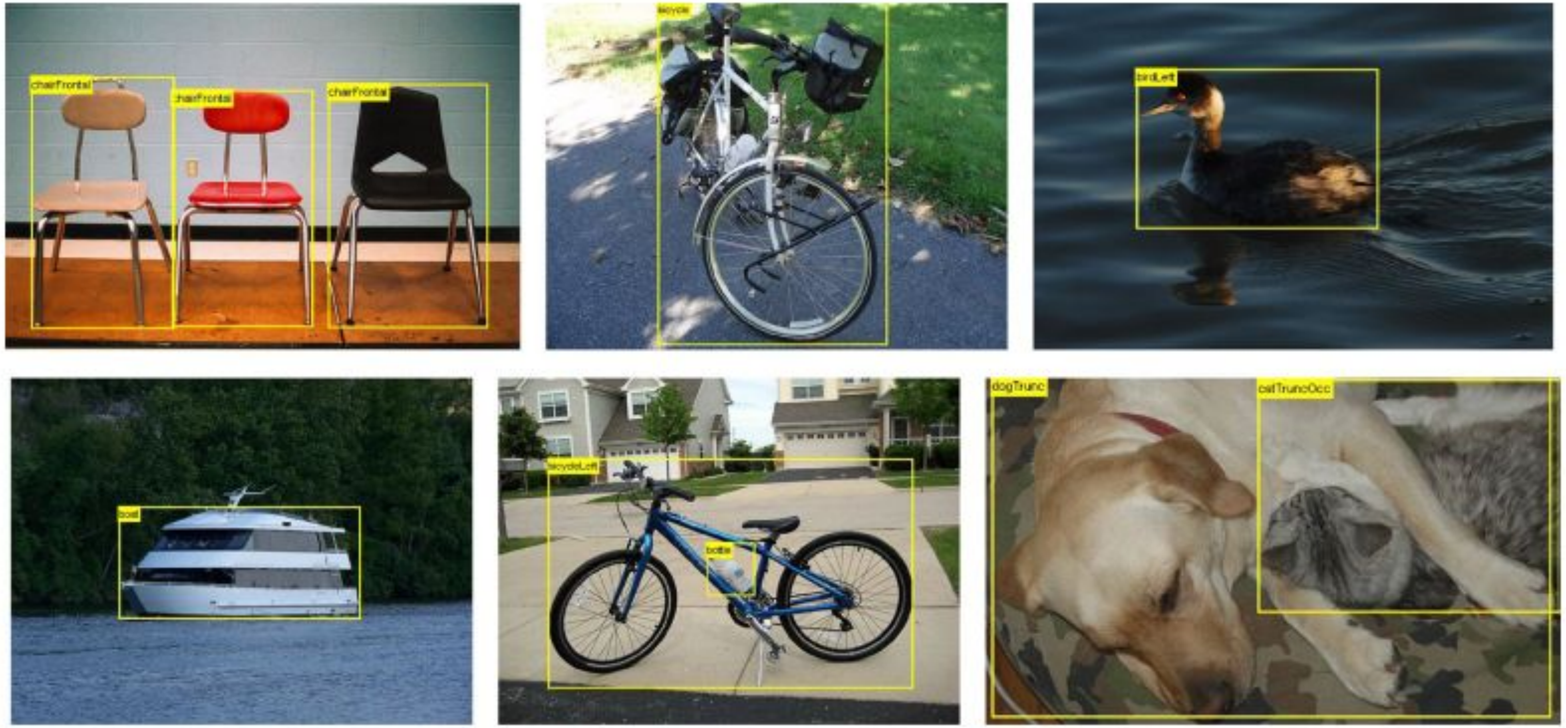
- Dataset used: **VOC 2012**
    - Dataset links: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/
    - Total 20 classes:



- *Person:* person
- *Animal:* bird, cat, cow, dog, horse, sheep
- *Vehicle:* aeroplane, bicycle, boat, bus, car, motorbike, train
- *Indoor:* bottle, chair, dining table, potted plant, sofa, tv/monitor

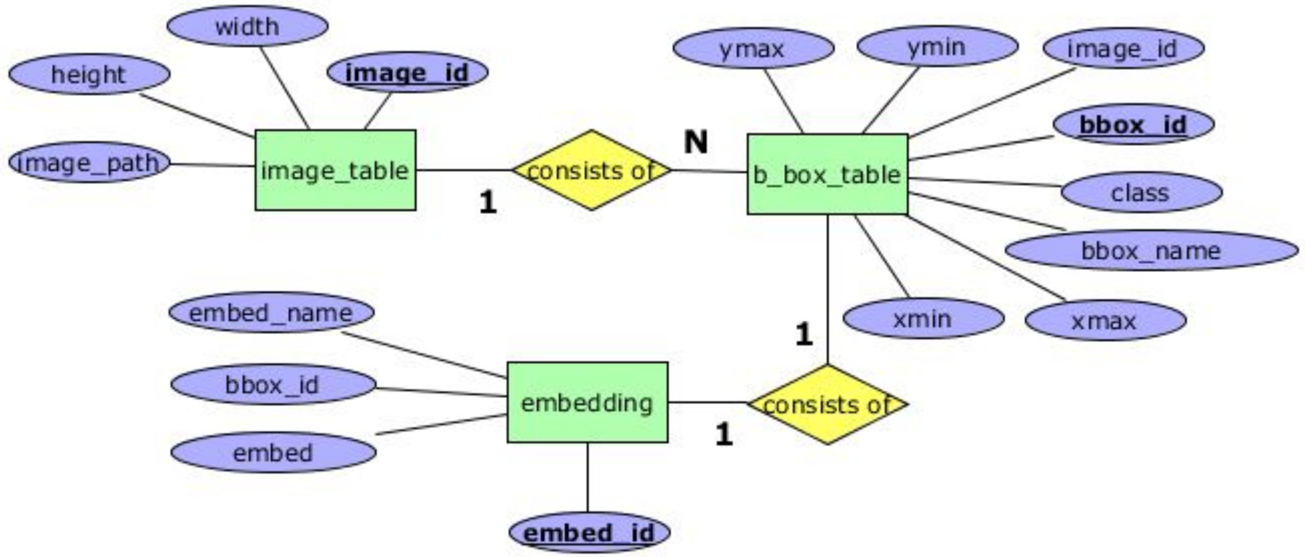
- Model used to generate image embedding: **SimCLR**
    - Github repo links: https://github.com/sthalles/SimCLR
    - We use Resnet18 model trained on CIFAR-10 dataset as our feature extraction model
    - Embedding size is 512 in our scenario
    - In short, SimCLR use contrastive learning technique to learn a more robust feature space for images.

Dataset & Model

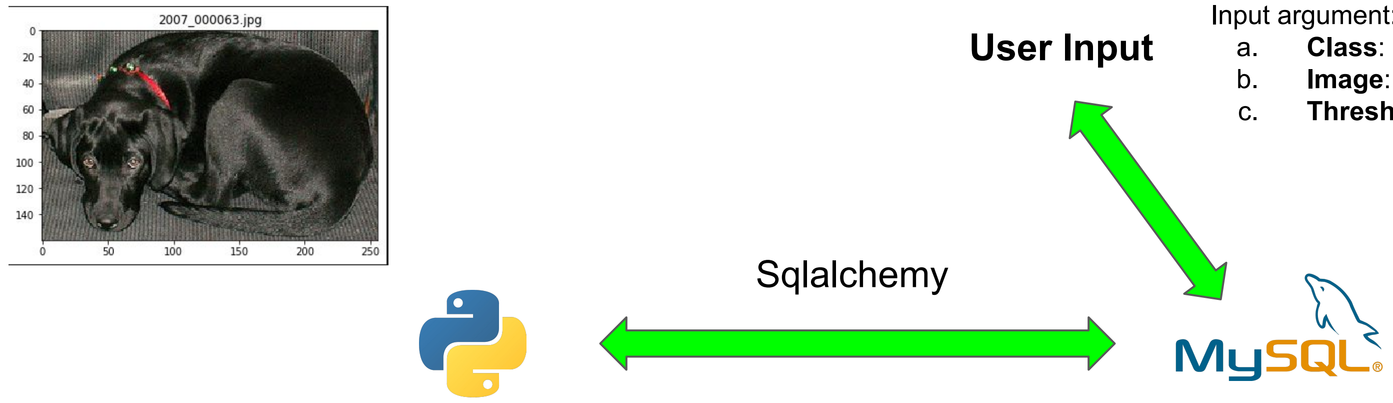


Some examples from VOC 2012 dataset

ER model of our object detection database



System Architecture



Input argument:

a. **Class**: dog, aeroplane……
b. **Image**: img_00.jpg……
c. **Threshold**: 0.3, 0.01……

<u>Preprocess</u>:

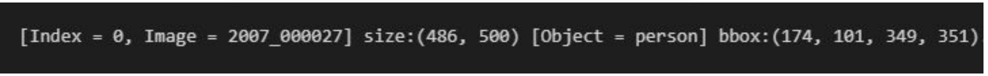
1. Insert VOC2012 dataset into MySQL DB
2. For each bounding box, generate its cropped image
3. Generate embedding for each cropped image via SimCLR model
4. Save embeddings output back to DB

<u>Main Task</u>:

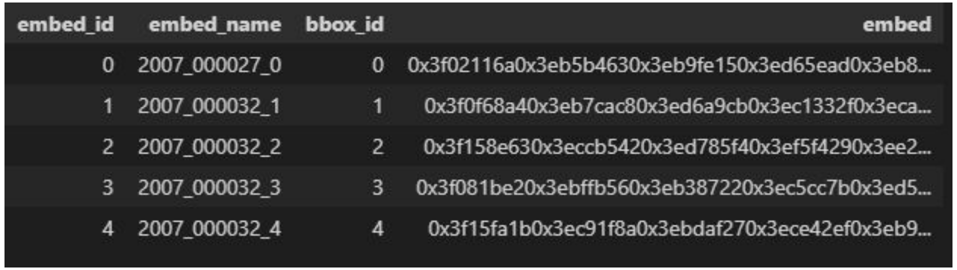
1. User input (i.e. class, image, threshold)
2. Prepare an SQL statement according to user input
3. Execute the statement in MySQL
4. Result save in a directory

Method

From the VOC2012 dataset, collect the image properties and corresponding bounding box properties from the annotation file



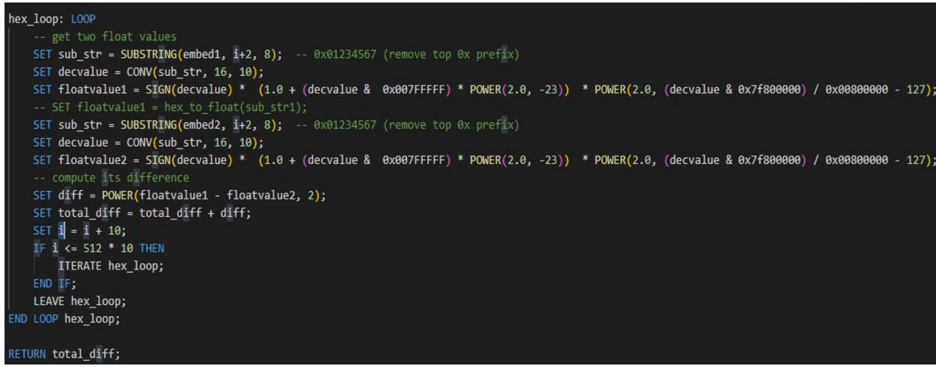
```
[Index = 0, Image = 2007_000027] size:(486, 500) [Object = person] bbox:(174, 101, 349, 351)
```



embed_id	embed_name	bbox_id	embed
0	2007_000027_0	0	0x3f02116a0x3eb5b4630x3eb9fe150x3ed65ead0x3eb8...
1	2007_000032_1	1	0x3f0f68a40x3eb7cac80x3ed6a9cb0x3ec1332f0x3eca...
2	2007_000032_2	2	0x3f158e630x3eccb5420x3ed785f40x3ef5f4290x3ee2...
3	2007_000032_3	3	0x3f081be20x3ebffb560x3eb387220x3ec5cc7b0x3ed5...
4	2007_000032_4	4	0x3f15fa1b0x3ec91f8a0x3ebdaf270x3ece42ef0x3eb9...

Main Task in details:

1. Convert embedding(HEX) to float32
2. Calculate mean-squared error of two embeddings between cropped images and image input by users
3. Find the best images with lowest error and return it as output



```
hex_loop: LOOP
    -- get two float values
    SET sub_str = SUBSTRING(embed1, i+2, 8);  -- 0x01234567 (remove top 0x prefix)
    SET decvalue = CONV(sub_str, 16, 10);
    SET floatvalue1 = SIGN(decvalue) *  (1.0 + (decvalue &  0x007FFFFF) * POWER(2.0, -23))  * POWER(2.0, (decvalue & 0x7f800000) / 0x00800000 - 127);
    -- SET floatvalue1 = hex_to_float(sub_str1);
    SET sub_str = SUBSTRING(embed2, i+2, 8);  -- 0x01234567 (remove top 0x prefix)
    SET decvalue = CONV(sub_str, 16, 10);
    SET floatvalue2 = SIGN(decvalue) *  (1.0 + (decvalue &  0x007FFFFF) * POWER(2.0, -23))  * POWER(2.0, (decvalue & 0x7f800000) / 0x00800000 - 127);
    -- compute its difference
    SET diff = POWER(floatvalue1 - floatvalue2, 2);
    SET total_diff = total_diff + diff;
    SET i = i + 10;
    IF i <= 512 * 10 THEN
        ITERATE hex_loop;
    END IF;
    LEAVE hex_loop;
END LOOP hex_loop;

RETURN total_diff;
```

Method

- 3 ways to store images in database
  - Image path: image stored in file system, not efficient when existing large amount of images (time complexity on searching in directory structure), also hard to maintain coherence
  - Image binary file: most space-efficient, since images are often compressed in those format
  - Image pixel values: not space-efficient, but good to manipulate on server side
- We can perform image cropping on server side when images are stored with pixel values (not space-efficient) ⇒ store image binary file instead

Problem Faced & Solution

- Problem:
  - Description: failed to implement crop() function in server side code
  - Reason:
    - Unable to perform cropping on server side when image stored as binary file
    - Exists a library (user-defined function) which is able to call Python script on SQL procedure to perform cropping ⇒ unluckily we fail
- Solution:
  - Fixed sql query for demo
  - Directly call crop() function in Python script

Demo Phase

Presentation Video Link:
https://drive.google.com/file/d/1r_x1lFeVS0h7tOzB8Jz0f19sTaekXGI_/view?usp=sharing