CSCI-5722 Computer Vision Final Project Report

Ryo Suzuki
Department of Computer Science
ryo.suzuki@colorado.edu
http://ryosuzuki.org/

Project Category: Photo Sorter

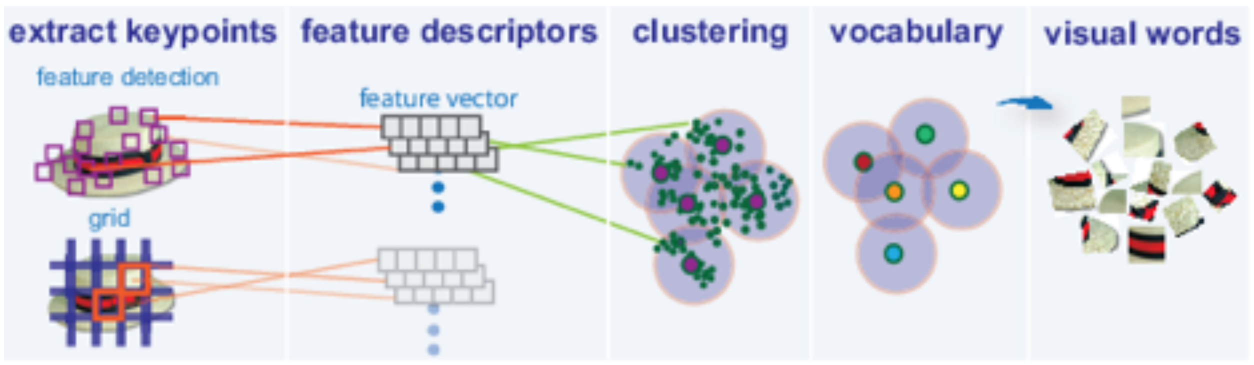
1. The initial project idea

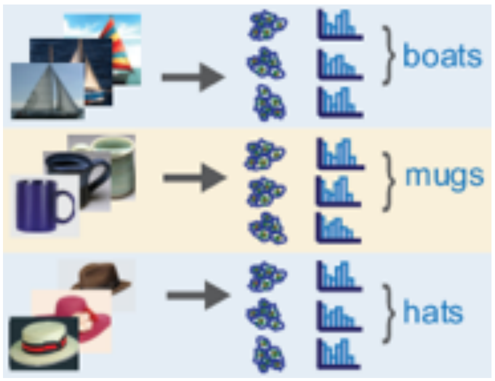
The initial project goal was basically the similar to the description in the provided instruction.
The simple web page that has

1. Detect the similar images from the pool of the images
2. Cluster the set of the images
3. Identify the category of the current image

2. Algorithm and approach

The task I tried to implement is well known as "image classification" in computer vision research. There are several approaches to achieving the image classification task, but I chose the "Bag of Visual Words" approach. I chose this approach because the idea is very straightforward and easy to understand for me (I took the natural language processing classes last year), but at the same time, it also seems to perform well.
The algorithm is very similar to the document classification in NLP. In the document classification, we identify the feature of words which represent the category (e.g. "football", "game", "score" for sports, and "president", "election", "white house" for political). After extracting these words from the document with the method like TF-IDF, we can identify the category of a new document by comparing the histogram of the frequency of the feature words.
The bag of visual words is basically the same, the only difference is that the bag of visual words uses "features" as "words". The algorithm basically consists of the following steps:





1.Extract the feature of images with the feature extraction like SIFT or SURF
2.Identify the “representative” of the feature vectors by categorizing with the method like K-means
3.Compute the frequency of the feature vector for each image
4.Create a histogram of the frequency and normalize with TF-IDF
5.Compare the histogram and identify the similar images or categories.

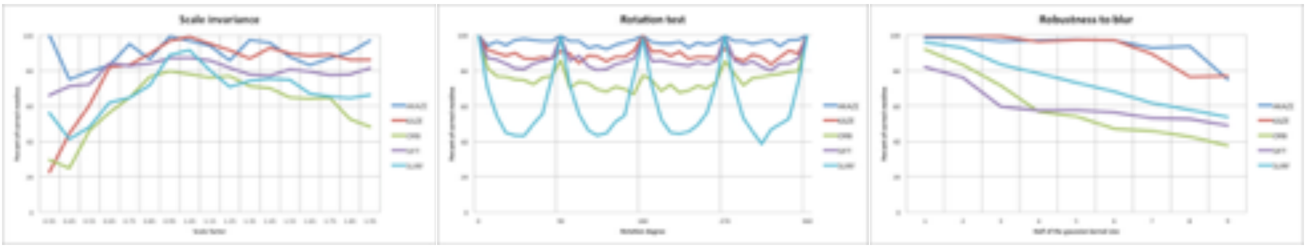
3. The method

The key to the success of this algorithm is the feature extraction method. I used the A-KAZE for the feature extraction since it is well known that A-KAZE has the better accuracy and performance. (resource: https://github.com/thorikawa/OpenCV-Features-Comparison). Since A-KAZE is available in OpenCV3 Core package, I use this library for the project.

Speed:

Algorithm	Average time per Frame (ms)	Average time per KeyPoint (ms)
AKAZE	30.4636	0.208972
KAZE	108.736	0.553694
ORB	5.78174	0.015048
SIFT	44.0598	0.161384
SURF	20.917	0.0444678

Accuracy:



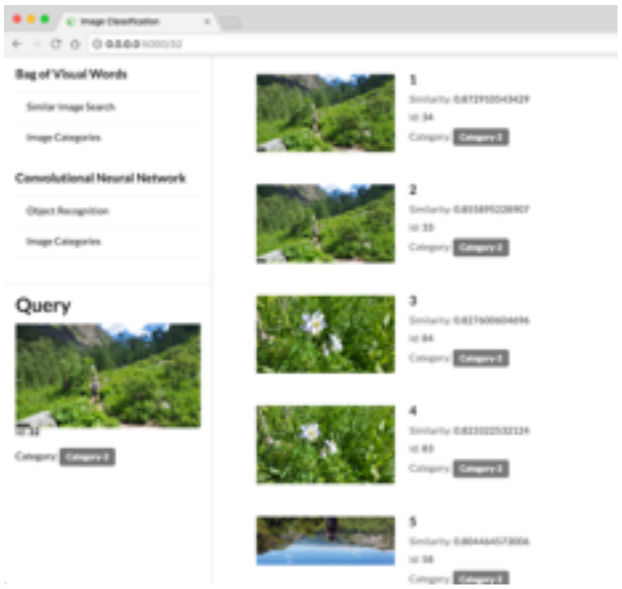
The next parameter is the dimension of the feature vector. I tested with the different parameters of the K-means in Step 2 (300 vs 3000), but I didn’t see any notable differences. I chose 300 for a project demo.

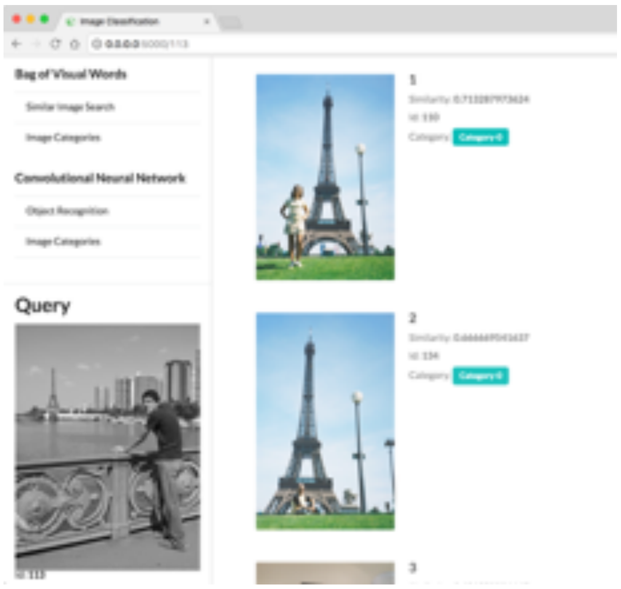
Finally, I used the simple K-means classification for the Step 5. The downside of K-means is that I need to choose the number of categories in advance, which may affect the accuracy. I tested with 5, 7, and 9 categories.

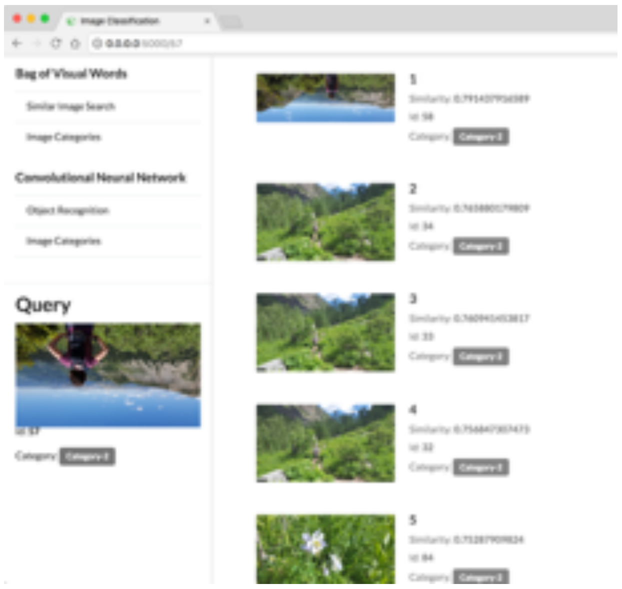
4. The data analysis and result

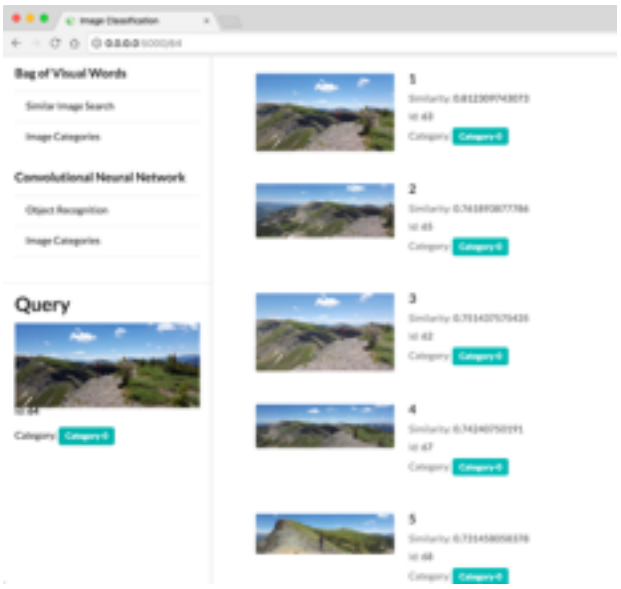
I used the set of images provided in Google Drive. Since the dataset is not supervised, so I just show the result with the screenshots of good and bad cases for the qualitative results.

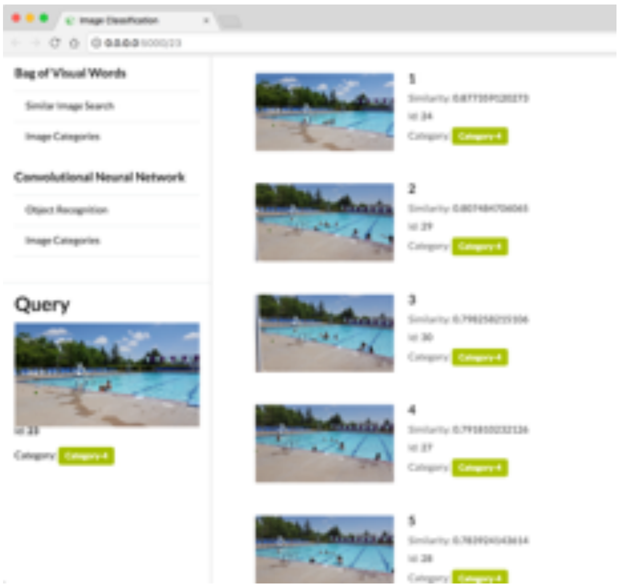
4.1 Successful results of similar image detection

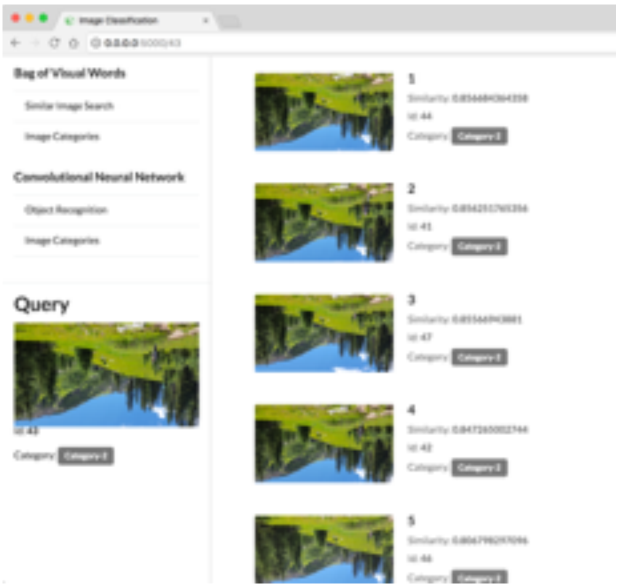




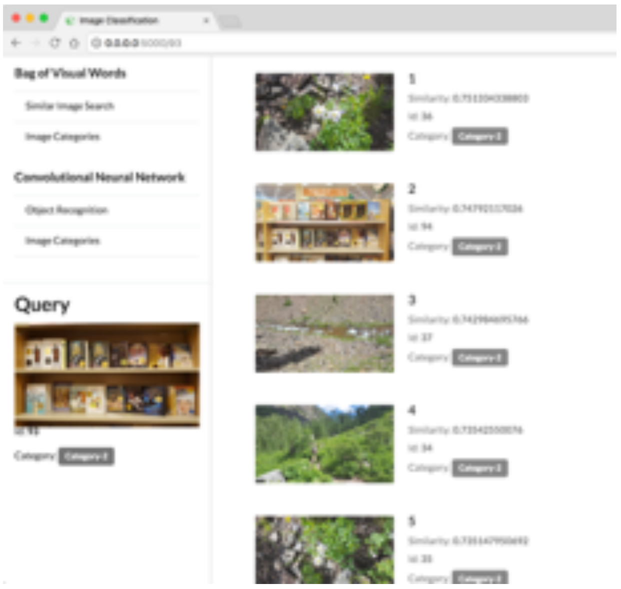


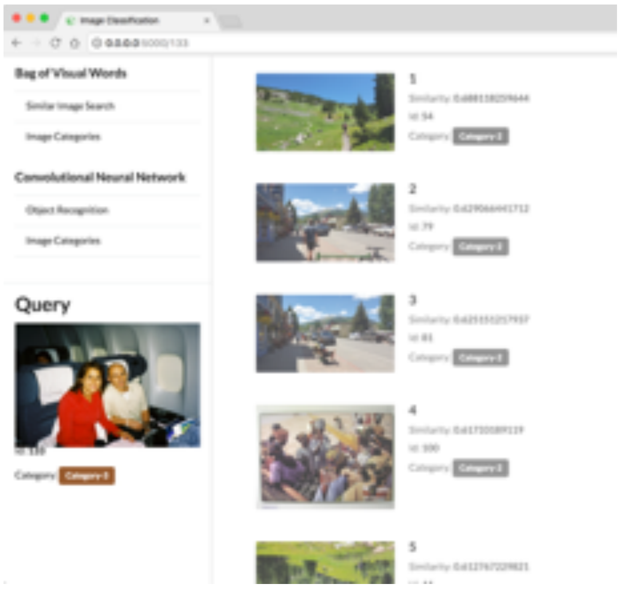




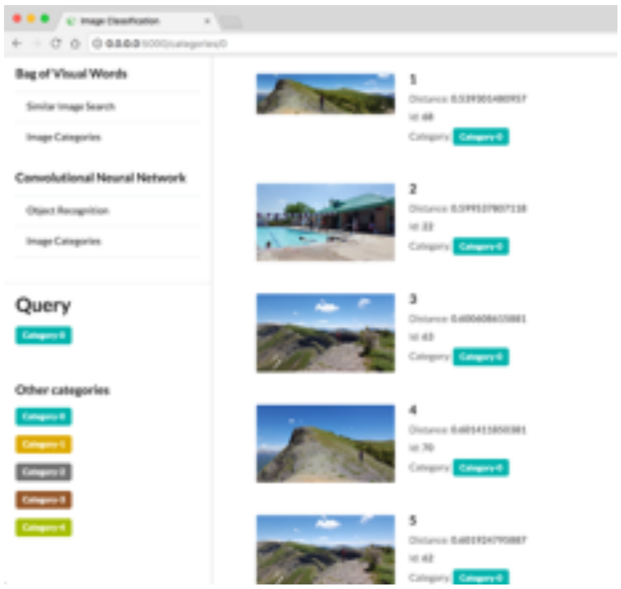


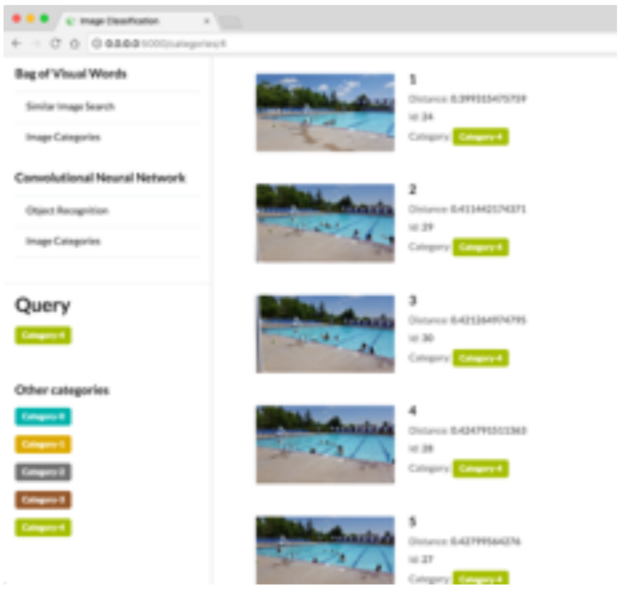
4.2 Unsuccessful results of similar image detection

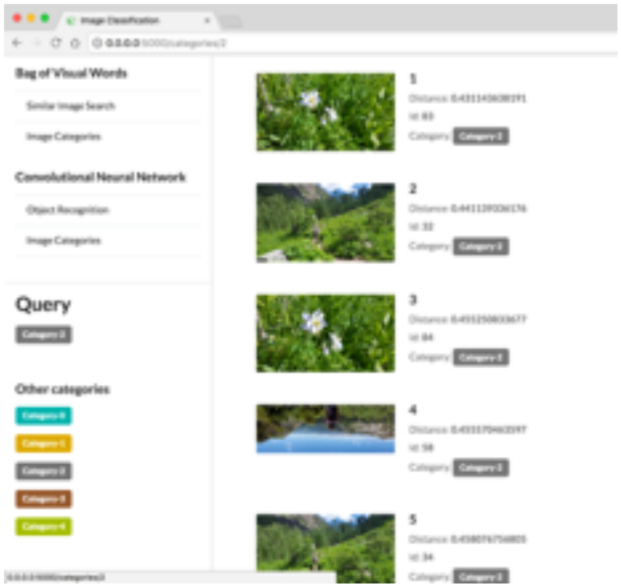


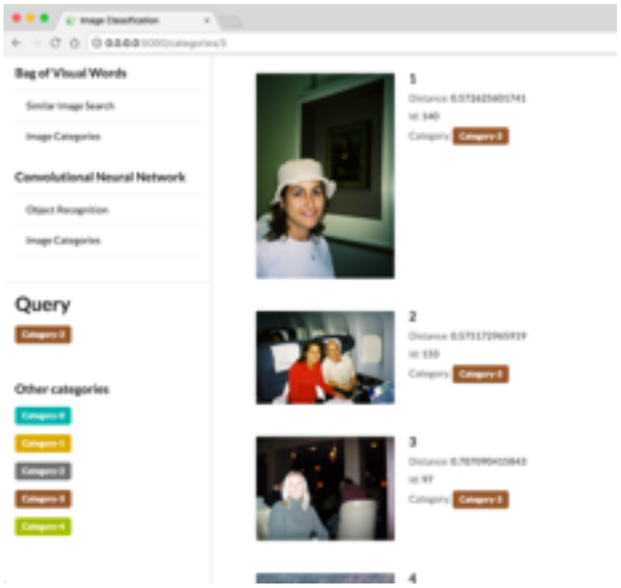


4.3 Results of image clustering









5. Known limitations

As you can see in the result section, identifying the similar or duplicated images perform fairly well, but the clustering task was not good enough. For example, I expected the algorithm can successfully classify the basic scene such as “forest”, “pool”, “architectures”, “human”, etc. But, it failed to do the meaningful classification particularly for the building and human.

The second limitation is that the user needs to manually label the classification. For example, the user can easily search the similar images by picking up the one image, but they cannot search like “beach” or “forest” without labeling by themselves. This could be the significant disadvantage comparing to the other services like Google Photos.

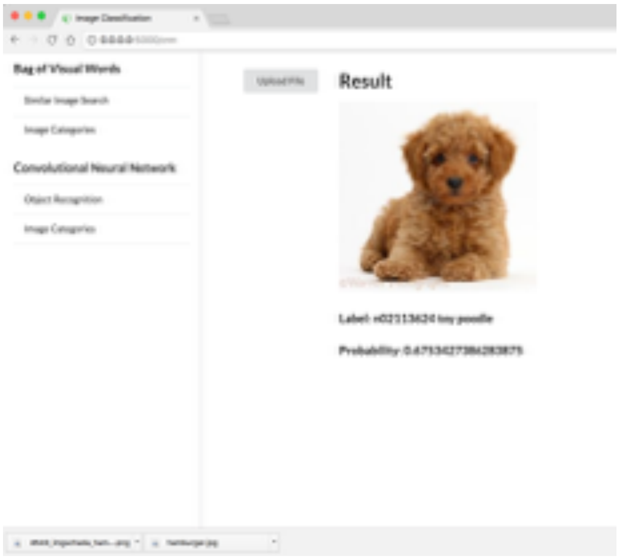
6. Deep learning: Beyond the traditional approach

After implementing the basic features of the bag of visual words, I wonder how the deep learning can differ from the traditional approaches such as the bag of visual words. So, I also implemented the object recognition and classification with the convolutional neural network (CNN) trained with the ImageNet dataset.

I basically followed the VGG-16, which won the second prize of ILSRC 2014 (ImageNet Large-scale Recognition Challenge) with 7.4% error rate. The paper can be seen in arXiv: https://arxiv.org/abs/1409.1556. VGG-16 has 16 weight layers

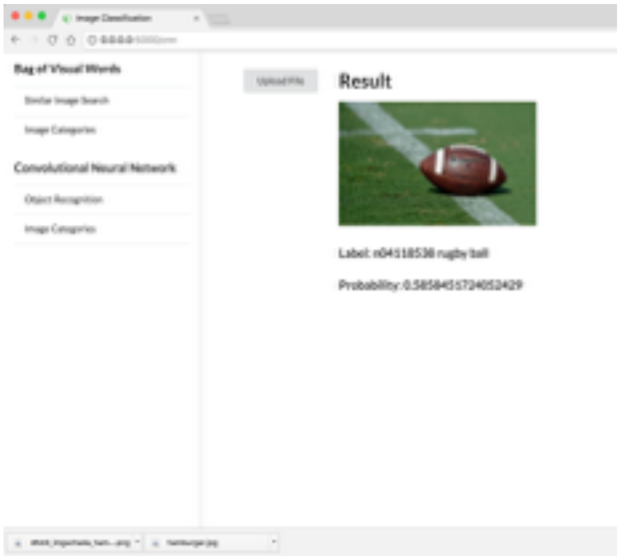
The trained model takes an image as an input and outputs the list of probability representing 1,000 ImageNet categories. (I also trained with CIFAR-10, but this limits the variety of the categories.)

By using this model, the program can automatically label the image (e.g. pig, elephant, cat, etc) and the user can retrieve the list of images by querying with the natural language, just like Google Photos. I will also show this in the demo.



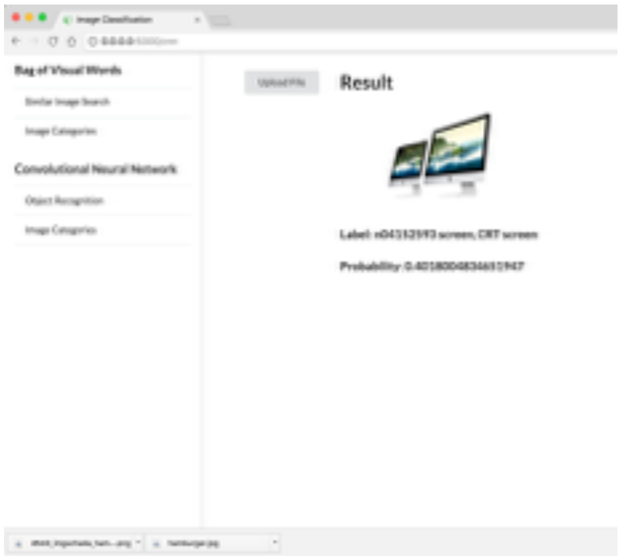
Label: n02113624 toy poodle

Probability: 0.6715427386283875



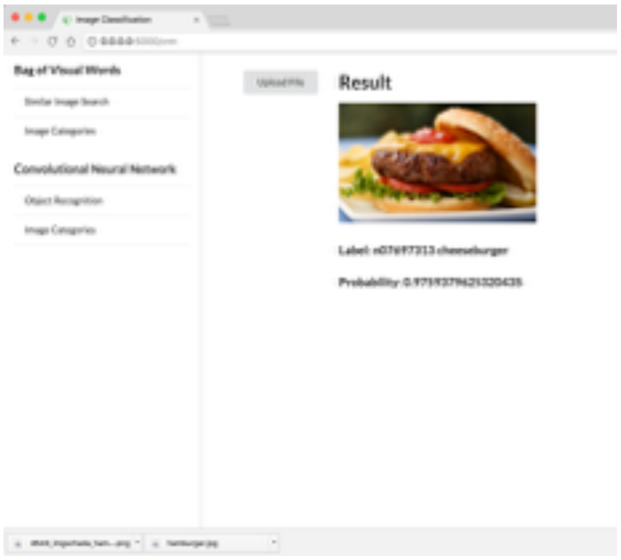
Label: n04118538 rugby ball

Probability: 0.5858455724052429



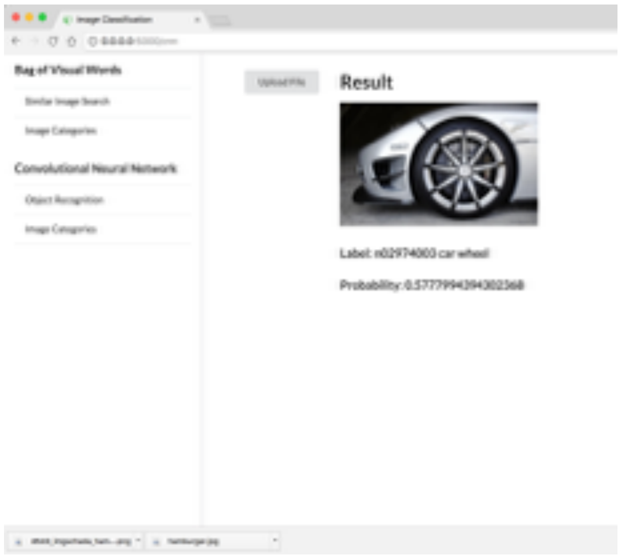
Label: n04152593 screen, CRT screen

Probability: 0.40180048346511947



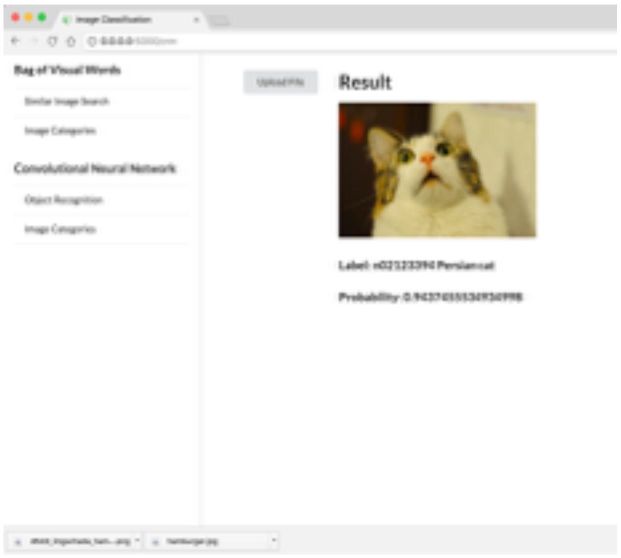
Label: n07697313 cheeseburger

Probability: 0.9719379621320435



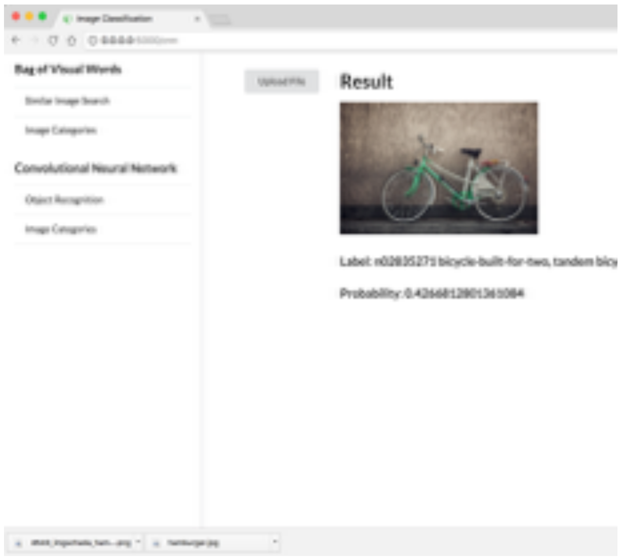
Label: n02974003 car wheel

Probability: 0.5777994394302368



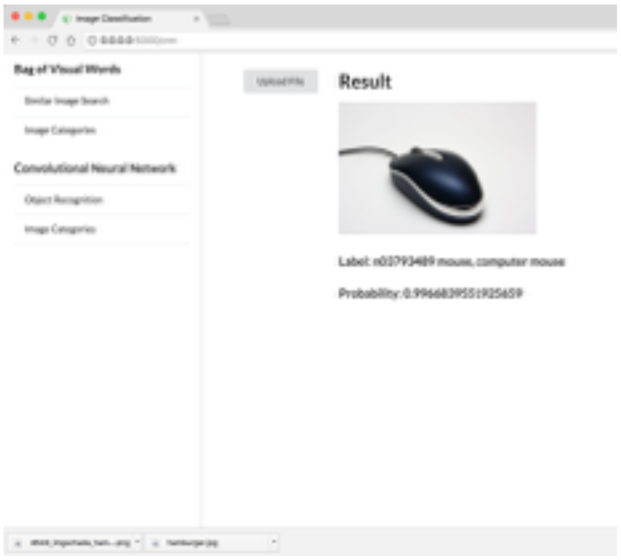
Label: n02123394 Persian cat

Probability: 0.9437455534934998



Label: n02835271 bicycle-built-for-two, tandem bicy

Probability: 0.4266812801361084



Label: n03793489 mouse, computer mouse

Probability: 0.9966839551925659

7. Advice for next year’s students

After implementing both the traditional approach (the bag of visual words) and the state-of-the-art approach (convolutional neural network), I learned the advantages and disadvantages of the both methods, particularly I learned the limitation of CNN techniques.

First, it requires the really “large” datasets to train the model. I understand why the computer vision researchers are so fascinated with ImageNet; ImageNet actually pushes the deep learning research forward. I also noticed that the technique for ImageNet is highly advanced (like VGG-16), but for the other domains, it seems really difficult to train and test the model.

Second, the CNN works pretty well for the object recognition of images provided by ImageNet, however, it cannot be transferred to the different task domain. In fact, I also tried to use the trained model for the scene classification (e.g. pool, forest, etc) for the original image sets, but the result was miserable. It was also reported that the bag of visual words may perform better than CNN when the dataset is small and different from ImageNet. (c.f. https://hal.archives-ouvertes.fr/hal-01056223/document)

Third, I realized that training CNN really takes time, not the scale of 1-2 hour, but 1-2 “days” without using GPU. But, GPU is expensive, and my MacBook Pro’s GPU was by AMD, so it cannot be used for the training. (Fortunately, I could use iMac with NVIDIA GPU + CUDA, but the set up was time-consuming.) This may be changed for the next 5 years, but it distracts me from rapid prototyping and test cycles.

While there are several drawbacks listed above, the deep learning may worth for learning. Particularly, learning both the traditional approach and the neural network approach may benefit to have a better understanding. I would recommend the next year’s student (and instructor) to learn (and teach) the basics of deep learning in addition to the traditional computer vision techniques.