CSC 411
Machine Learning & Data Mining
ASSIGNMENT # 3
Out: Nov. 17
Due: Dec. 2, 4:00PM EST

Image Scene Prediction (100 points)

Imagine that you are hired by a large social media website, where the task you are given is to sort millions of images uploaded by your users into different categories depending on the subject of the photo. This would have taken you years. Fortunately, you are a student of CSC411, and now have a number of interesting machine learning tools at your disposal.

Instructions for Work

You are allowed to work in pairs and your solutions will be entered into a competition against your classmates. While you will be awarded marks for doing well in the competition, it is by no means the only way to achieve a high score. Rather, you will be graded on the creativity of your solutions, and the clarity with which you are able to explain them. If your solution does not live up to your expectations, then you should explain why and provide some ideas on how to improve it. You are free to use any third-party ideas or code that you wish as long as it is publicly available. You must provide references to any work that is not your own in the write-up.

We will be using the Kaggle platform (`http://inclass.Kaggle.com/`) in order to allow you to upload your solutions so that your results can be compared with those of your peers. First of all go to the Kaggle in Class website, and create a user account **with your university email address (ending in "mail.utoronto.ca")**. We will be posting a link to the actual competition shortly. However, we are providing the files for this competition so that you can start right away.

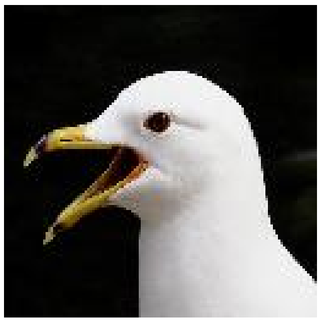
Description of the data

Our dataset is derived from FLICKR image tagging dataset. Each image is a 128x128 color JPG image. Each image is given a label, with the labels being {**1-structures, 2-indoor, 3-people, 4-animals, 5-plant life, 6-food, 7-car, 8-sea**}. These images have been shuffled and pre-cropped randomly from the original dataset. Some of the example input images are shown in Figure 1

(a) Structures

(b) Indoor

(c) People


(d) Animals

(e) Plant life

(f) Food

(g) Car

(h) Sea

Figure 1: Training examples from the Toronto Faces Dataset.

The data is divided into 3 parts. There are 7000 training images with labels, 970 public test images, and 2000 private test images. You will only receive corresponding labels for the 7000 training images. The Kaggle website will score your predictions on the public test images, and rank you submissions from today until the submission deadline. A few days before the deadline, we will also release a set of hidden test images (so called because it will be hidden from you until a few days before the deadline). You will be asked to submit your final predictions for both the public and the hidden test images to Kaggle. Your score on the hidden test set is not shown to you until after the deadline. This way, you cannot optimize your models by trying different submissions.

Baseline

Before deep learning, people often used handcrafted feature extractors (functions that are run over an image to extract vectors describing some region of interest). You can visit Wikipedia to get more information about this idea: `http://en.wikipedia.org/wiki/Feature_(computer_vision)`.

We used a variant of this concept called GIST `http://people.csail.mit.edu/torralba/code/spatialenvelope/` to extract a description vector for the entire image. Following this, we performed a K-Nearest Neighbours grouping on the resulting feature space. Using this, we achieve a 44.22% accuracy on the public test set, where accuracy is defined as:

$$a = \sum_i \frac{\text{I}[\text{pred}_i == \text{label}_i]}{N_{\text{sample}}} \tag{1}$$

where i indexes the sample, and $\text{I}[\text{pred}_i == \text{label}_i]$ is an indicator function that returns 1 if the condition holds, and 0 otherwise. Note that this accuracy is calculated based on all classes.

This is a very simple baseline, and is provided for your interests only. We expect that you should be able to create much better classifiers. For example, a convolutional neural network based classifier should be much more promising.

We expect a reasonable model to have somewhere above 50% accuracy.

Output Format and Kaggle Submission

The Kaggle competition accepts CSV files. In particular, the file should have two columns. The first column holds the image ID, while the second column holds the predicted class. Please see the attached sample submission for an example.

In particular, note that the first 970 IDs are part of the public test set, while the following 2000 IDs are part of the private test set. You can submit two sets of solutions to Kaggle per day. Before the private test set is released, you should set the predicted labels for all samples in the private test set to 0. After the private test set is released, you will submit your predictions for the full set of 2970 samples.

Deliverables

Please hand in:

1. The generated result file for the Kaggle competition on the public test set.

2. A write-up containing no more than 6 pages that is single-spaced with a font no smaller than 12-pt Times New Roman. The write-up should include:

   (a) An introduction describing at a high level some approaches that you considered, and why you considered them.

   (b) A description of your submitted solution, including any data processing, algorithms used, etc.

   (c) A section describing some empirical results from your solution. That is, some experiments that demonstrate that your solution is sensible. **This section must include a comparison of at least three approaches: our GIST+KNN baseline, your best approach, and at least one additional approach of your choosing**. Additionally, you may choose to include other experimental results such as an evaluation of a regularization technique to prevent overfitting, a comparison of different model parameters (such as tanh vs. sigmoid neurons/the number of neurons in a neural network, or different kernels/kernel parameter settings in an SVM). The idea is to justify your approach, and to demonstrate the different factors that you considered for your submitted solution.

   (d) A conclusion summarizing your work and findings. If your method performed poorly on the public or hidden test data, you may want to include an explanation as to why and suggest how your method may be improved.

   (e) References to any code, methods, or ideas that you used that are not your own. Remember, these must be publicly available and free to use.

If you are working for this part of the assignment as a team of two people, you and your partner should submit a single write-up. Please make sure to include both of your names and Kaggle username.

Your grade will be based on 4 criteria:

1. **Classification performance (10 pts):** You will get 10 points for beating the baseline. Note that your performance will be evaluated on hidden test set, therefore you should not overfit to the labeled and public test set.

2. **Design and analysis of experiments (30 pts):** You should follow the best practices of experimenting as discussed in the lectures and in your textbook. Using cross validation and performing statistical tests to prove the significance of your results will earn you points. Moreover you should follow a logical way to find and select the best hyperparameters.

3. **Report organization (30 pts):** Writing a good report is as important as programming a good classifier. You should present your results in a clear and concise fashion. You

should create plots and tables to show important trends and results. Your reports should not exceed 6 pages including figures. It can be less than 6 pages, so don't try to make it too long. Reports should be typed, not hand-written. **ALL information needs to be included in the 6 page write-up, except the result file for the public test set.** You should give attention to details such as naming figures correctly, using the right mathematical terms, labeling the axes of plots, etc.

4. **Comments on method and results (30 pts):** Your method should be motivated and justified from the structure of the dataset. You should explain for which cases your method worked well and for which cases it failed and why. You should comment on your overall performance and explain why is it high/low. You should also try different variations of your method (regularization, different kernels/neuron types/distance metrics etc.) and comment on what changes when you try different variants.

BONUS (15 pts)

As an add-on to the single label prediction, we can further assign multiple labels to an image. In this case, this is a binary classification problem for each possible label. You will notice that in many cases, the multiple labels assigned are not random. In general, there are two types of relationships. The first type is *co-occurrence*. For example, images of the ocean often have *sea* and *sky* at the same time. The second type is *hierarchical*. For example, an image tagged as *woman* should always have *people* as a tag as well.

In this case, the evaluation metric is a little different:

$$a = \sum_i \sum_j \frac{\mathrm{I}[\mathrm{pred}_{i,j} == \mathrm{label}_{i,j}]}{N_{\text{sample}} N_{\text{classes}}} \tag{2}$$

where all j indexes over all N_{classes} possible classes, and the other terms are as defined previously. Note that this accuracy takes the accuracy of all classes into account.

This task will not be part of the Kaggle competition. Instead, you will describe your procedure and results in your report.

Hints

As is the case when you tackle real machine learning problems, you should first examine the training data available to you. In particular, you should look through the data to see its formatting, and also calculate some statistics. For example, is the data evenly distributed among classes? If not, you may need to consider various ways to handle this imbalance.

You are free to try any methods or approaches that you wish for this assignment. There are many possible solutions; in fact this dataset is currently being used as a benchmark for active research. There are many approaches that you can try, including discriminative methods (k-nearest neighbors, SVMs, kernel SVMs, neural networks, etc.), generative methods (mixtures of Gaussians, etc.) and dimensionality reduction (PCA, etc.). Your solution does not have to be limited to simply applying machine learning models. In the real world, the best approaches can often be simple ones that rely on clever data processing.