Multi-Class Image Classification

Due Wednesday, May 1, 2013 via Blackboard

Out April 17, 2013

1 Important Note

As usual, you are expected to use Java for this assignment. It could take hours to run your experiments. Start early.

Bin Zhao (binzhao@cs.cmu.edu) is the contact TA for this assignment. Please post clarification questions to the Google Group:
`machine-learning-with-large-datasets-10-605-in-spring-2013`

2 Background: Bag-of-Words Feature Representation for Images

A simple approach to classifying images is to treat them as a collection of regions, describing only their appearance and ignoring their spatial structure. Similar models have been successfully used in the text community for analyzing documents and are known as "bag-of-words" models, since each document is represented by a distribution over fixed vocabulary(s). In this assignment, we will carry out multi-class image classification, based on "bag-of-words" feature representation for images.

Creating a Visual Vocabulary Methods for classifying text documents are mature, and effective enough to operate with millions or billions of documents at once. Documents of text contain some distribution of words, and thus can be compactly summarized by their word counts (known as a bag-of-words). What cues, then, can one take from text processing to aid visual search? An image is a sort of document, and it contains a set of local feature descriptors. However, at first glance, the analogy would stop there: text words are discrete tokens, whereas local image descriptors are high-dimensional, real-valued feature vectors. How could one obtain discrete "visual words"?

To do so, we must impose a quantization on the feature space of local image descriptors. That way, any novel descriptor vector can be coded in terms of the (discretized) region of feature space to which it belongs. The standard pipeline to form a so-called "visual vocabulary" consists of (1) collecting a large sample of features from a representative corpus of images, and (2) quantizing the feature space according to their statistics. Often simple k-means clustering is used for the quantization; the size of the vocabulary k is a user-supplied parameter. In that case, the visual dictionary is the k cluster centers.

Bag-of-Words Feature Representation Once a visual vocabulary is established, the corpus of sampled features can be discarded. Then a novel image's features can be translated into words by determining which visual word they are nearest to in the feature space (i.e., based on the Euclidean distance between the cluster centers and the input descriptor). Finally, each image is represented as a discrete vector, where each term in the vector is a "word".

3 Data

In this assignment, we will use a subset of ImageNet. The data appears at /afs/cs.cmu.edu/project/bigML/imagemc/.

The data set contains images from 100 classes. Therefore, we will be doing a 100-class image classification. The data has been divided into 3 parts: training data (image_train.txt), validation data (image_val.txt) and test data (image_test.txt). Specifically, the training data set contains 200 images for each class, making the total number of training images to be 20000. Each image is represented as a vector, whose elements are indices in the visual vocabulary, i.e., each element in the vector is a discrete number. The visual dictionary contains 1000 visual words in total. Therefore, each element in the vector is an integer between 0 and 999.

For training data, there is one instance per line. The first token of each line is the class label, then a tab, then the data. The training data are sorted from class 1 to class 100, meaning the first 200 images are from class 1, the next 200 images are from class 2, etc. For validation data, the format is the same as training data, with the exception that validation data are not sorted according to class label. For test data, we only give the feature part, without the corresponding label.

In this assignment, we will carry out training using the training data, and tune parameters using the validation data, and finally test your classifier on the test data.

4 One-vs-Rest Multi-Class Classification

One of the simplest, yet very powerful way of constructing multi-class classifier is to combine multiple binary classifiers. In this part, you will be using one-vs-rest strategy to combine multiple binary classifiers to get the multi-class classifier. Specifically, with K classes, K binary classifiers are learned, with each discriminating class k from all other classes. For example, for k-th binary classifier, the positive examples will be those with class label k, and the negative examples are those with class label other than k.

Consequently, for the data given in this assignment, you will be learning 100 binary classifiers. Please use the efficient regularized SGD for logistic regression in Assignment 5 to train the binary classifiers. Represent each image as a 1000 dimensional vector, with the i-th element being the count of $i-1$ in the image data. Since each image has different number of "words", we further divide the 1000 dimensional vector by its L_2 norm, such that the newly obtained feature vector has norm 1.

For each binary classifier, it will have 200 positive data points, and 19800 negative data points, making the training data imbalanced. To solve this problem, we will up-sample the positive data. Specifically, assume we are training the k-th binary classifier, replicate the 200 class k data points by 99 times, such that we have 19800 positive data points. Randomly sort those data and feed into SGD. Train 5 epochs for SGD.

After SGD training, we will end up with 100 parameter vectors $\{\beta_1, \dots, \beta_{100}\}$. Given a test data point, the label could be decided as $y^* = \arg\max_k \beta_k^\top \mathbf{x}$, where $\mathbf{x}$ is the normalized word count feature.

4.1 Parameter Tuning on the Validation Set

As in many machine learning algorithms, selecting the optimal parameter is crucial for the success of the algorithm. In this assignment, set $\eta = 0.5$, but try different values of μ. Apply the learned multi-class classifier on the validation set. Pick the optimal value of μ as the one giving highest accuracy on the validation set.

5 Deliverables

Submit a compressed archive (zip, tar, etc) of your code. Also, submit a txt file named *test.txt*, containing the classification result on the test data. Specifically, each row of the txt file contains one single label. The label file should strictly follow the order of the test data, otherwise I cannot compute accuracy of your classification.

Also respond to the following questions.

1. Provide the predicted label file (named *val_balanced.txt*) for validation set, using your selected optimal parameter μ. In this label file, each row only contains one label. Also provide the classification accuracy computed on the validation set.

2. Using your selected optimal μ, re-train the multi-class classifier. However, this time, do not up-sample the positive examples, i.e., for each binary classifier, there will be 200 positive data points, and 19800 negative data points. Provide the predicted label file (named *val_imbalanced.txt*). Also provide the classification accuracy computed on the validation set. Is the new classification accuracy different from previous one? Discuss the reasons.

3. Besides one-vs-rest, there are many other ways of constructing multi-class classifier. One example is the hashing technique discussed in class. For multi-class classification, hashing technique hashes the feature vector $\mathbf{x}$ together with label y to a new feature representation $\phi(\mathbf{x}, y)$. Then we will learn a single parameter vector β, where a test data point $\mathbf{x}$ is labeled as $y^* = \arg\max_k \beta^\top \phi(\mathbf{x}, k)$. Using L_2 regularization on β (as in Assignment 5), write out an optimization formulation to learn the vector β. Namely, write out the following formulation

$$\begin{aligned} \min_{\beta} \quad & L(\beta) + \mu\|\beta\|_2^2 && (1) \\ s.t. \quad & constraint(\beta) && (2) \end{aligned}$$

where $L(\beta)$ is a function of β, and $constraint(\beta)$ are constraints. Give your formulation of $L(\beta)$ and $constraint(\beta)$, and explain the reason for your choice. For this question, you might want to refer to the following work on multi-class classification: {K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-basd vector machines. Journal of Machine Learning Research, 2:265-292, 2001.}

6 Marking breakdown

- Code correctness and test label file: [**70 points**].
- Question 1, 2, 3 [**10+10+10 points**]
- BONUS credit: [**10 points**] We will compute classification accuracy using your submitted test label file and rank your accuracy. 10 points will be given to the best performing classifier, 9 points to the runner-up, 8 points to the third best, etc.