**Bag of Words (MATLAB)**

**Domingo Martinez**

**PROBLEM**

## Prediction

## For a given image set X, of considerable size and, one vector y of the class that each image belongs. Use bag of word method to find the features that identify each image, then use this reduce number of feature to train a logistic regression model and predict the class that belong any new images set not ever see before.

## Training

Learning the weights by minimizing the cross-entropy loss function.

1. Consider both the original loss and the one with a L2 regularization (of course this involves a weight of the regularizer).
2. Implement functions to evaluate the gradient and the loss using matrix operation.
3. Implement steepest descent (using gradient with a step size)
4. Optimize parameters by cross-validation. Explain details of the cross-validation.

## Evaluation Metrics

Evaluate by comparing different baselines, different parameter settings. The point is to see both the accuracy and the efficiency.

1. Accuracy: number of correctly classified test data over the whole dataset.
2. Convergence rate: plot the loss function as a function of the number of iterations. You should see a converging curve.
3. Measure the average time for each iteration, and for prediction.

## Data

Evaluate the methods on three different datasets.

1. Caltech 101 dataset.

**SOLUTION**

## Prediction

Dealing with images of median and generous size represent a complication. If we try to use all the features and given it directly to our classifier, in this case logistic regression, we will have more features and, thus, more data to train our model at the same time, more time to process all the data.

For this reason, we use bag of words. Whit this method we can pre-process each imagen and find a reduced number of features that can easily manager for our logistic model and train it with relative small number data training.

Bag of word, as their name point out, is based, like in a written document, where are words that appear more often than others and if we group up it we can identify the topic of such document, bag of words not only evaluates the words that are repeated they also evaluate which word is close to other. To do that this method uses a vector representation of words to predict the general meaning on a given context where the words appears.

We can apply this principle to images and get as result the meaning of the image, in our case, if the imagen is an airplane, a ferry or a laptop in the Caltech data set.

In the following line, we can see how implement this method using the tool box included whit Matlab.

bag = bagOfFeatures(imageSet(imds.Files), 'VocabularySize',words);

We specify the number of features that we want to work with, and then we use these features to codify all the images and reduce their entire number of features, also this process must be done to the new imagen vector to test, we can see how has been done this process:

for i = 1:length (imds.Files)

img = readimage(imds, i);

X(i,:) = encode(bag, img);

if (imds.Labels(i) == 'airplanes')

y(i)=1;

elseif (imds.Labels(i) == 'ferry')

y(i)=2;

elseif (imds.Labels(i) == 'laptop')

y(i)=3;

end

end

## Training

Having implemented bag of words method and reduced the number of features, the next step is train our logistic regression model, for that we will use a cross validation algorithm with a number of given folds, as follow:

tic

indices = crossvalind('Kfold',n,folds);

error\_training = zeros(folds, 1);

error\_test = zeros(folds, 1);

for i = 1:folds

fprintf('\nFold: %f', i);

test = (indices == i); train = ~test;

X\_train = X(train,:);

y\_train = y(train,:);

X\_test = X(test,:);

y\_test = y(test,:);

% run gradient descent

[all\_w, j\_h] = steepestGradientDescent(X\_train,y\_train,alpha,iterations,num\_labels,lambda);

fprintf('(Done %f)',toc);

pred = predict(all\_w, X\_train);

pred\_test = predict(all\_w, X\_test);

error\_training(i) = mean(double(pred == y\_train)) \* 100;

error\_test(i) = mean(double(pred\_test == y\_test)) \* 100;

end

fprintf('(Done %f)',toc);

% Plot the convergence graph

plotGradient(j\_h,'Steepest Gradient Descent');

fprintf('\nTraining Set Accuracy: %f', mean(double(error\_training)));

fprintf('\nTest Set Accuracy: %f\n', mean(double(error\_test)));

## Metrics Evaluation

There are different evaluation metrics applied in this project: Plots of the data to use, plots of the gradient descent and convergence rate in each iteration, plots of bias vs variance, number of words to use, time of training each method and a complete parametrizable part with variables, as long with a detail log and the times required to run each experiment.

The cross-validation method used was k-folds, this means 100% of the data were used to train and test the model, for this propose was left a parameter setup to configure the number of folds to use. Then was measured the mean accuracy with respect the number of correct predictions for training and test data.

**EXPERIMENTS**

## Caltech Dataset

For this experiment was used Caltech-101 dataset for three kinds of images airplanes, ferry and laptop for a total of 201 samples. The implementation for this dataset experiment is developed on the file *bagOfWords.m*, with parameters as following:

%% Setup of parameters

num\_labels = 3; % 10 labels, from 1 to 10

lambda = 0; % Regularization parameter

iterations = 1400; % Number of iterations gradient descent

alpha = 0.2; % Steep size for gradient

folds = 10; % Number of folds to use on cross validation

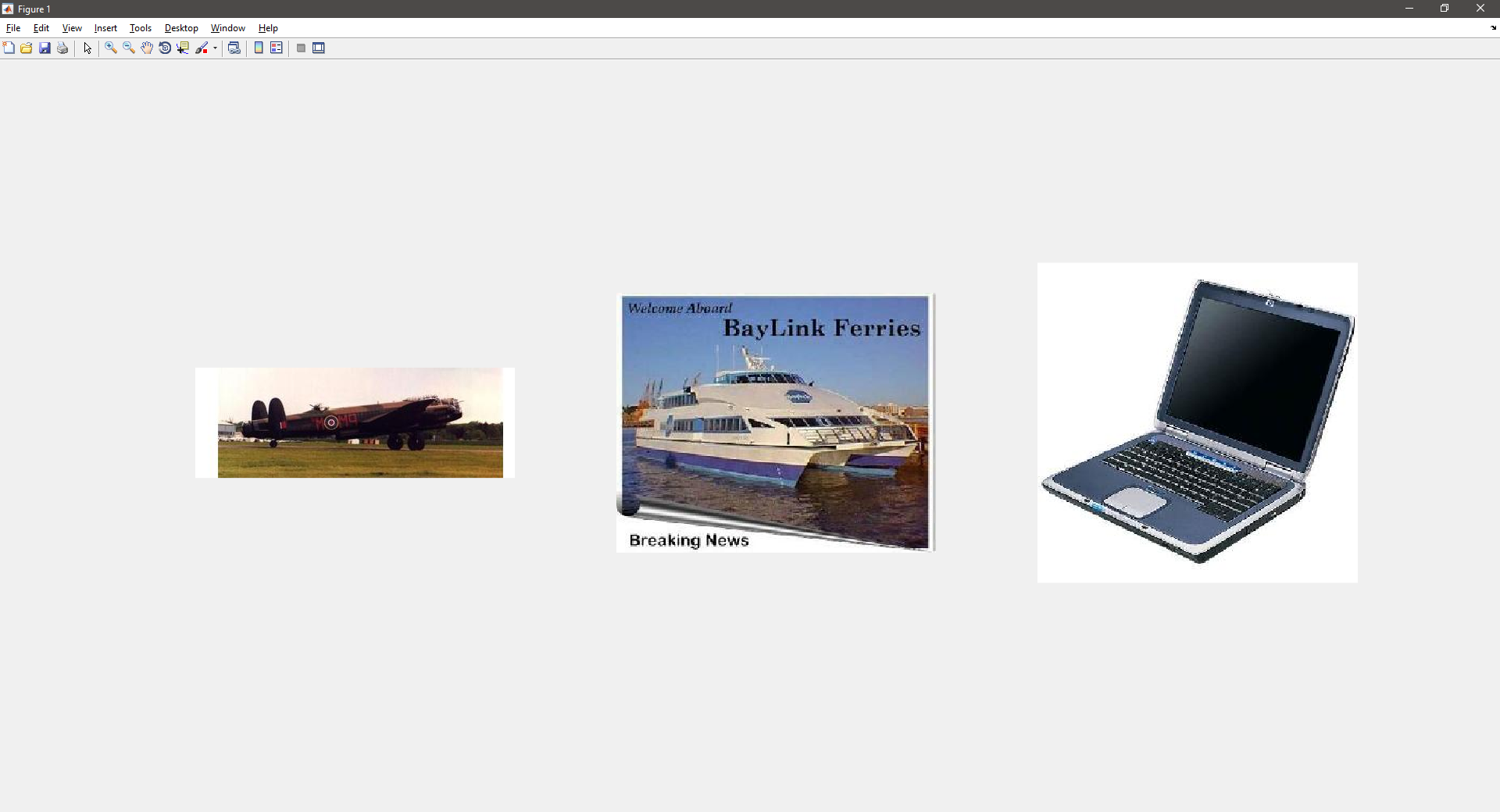
words = 200; % Number of words to use on bagOfWords

%% Optional parameters (y/n)

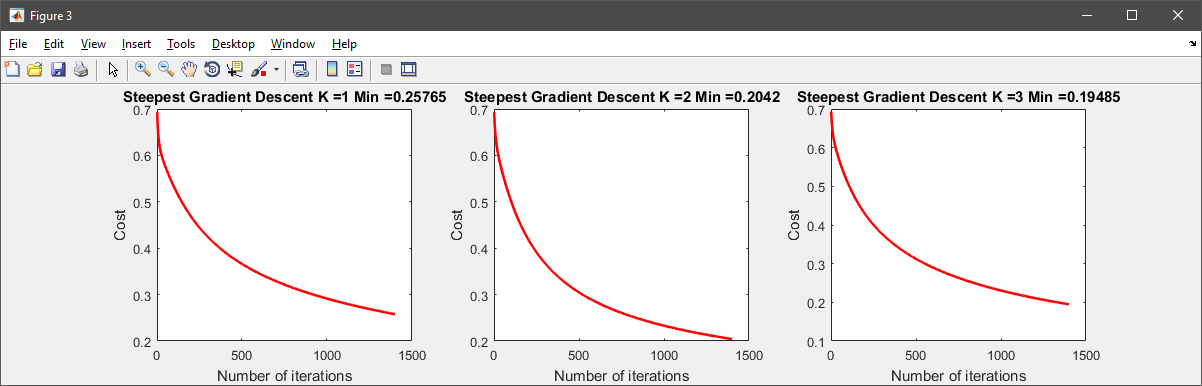
test\_steepst\_descent='y'; % Test gradian steepest descent

test\_optimal\_lambda='y'; % Test different lambda parameters (time consuming)

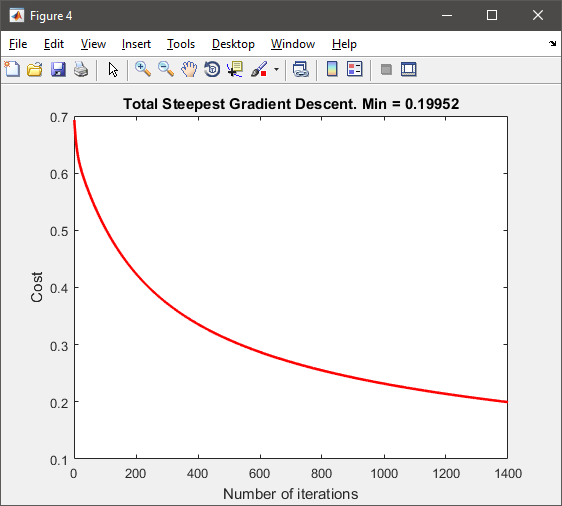
The first step was plot the data in order to visualize it and have a beter comprenhantion of the model to train. The model to use in all the experimients was a linear one, that means, the maximun polinomial degree is one.



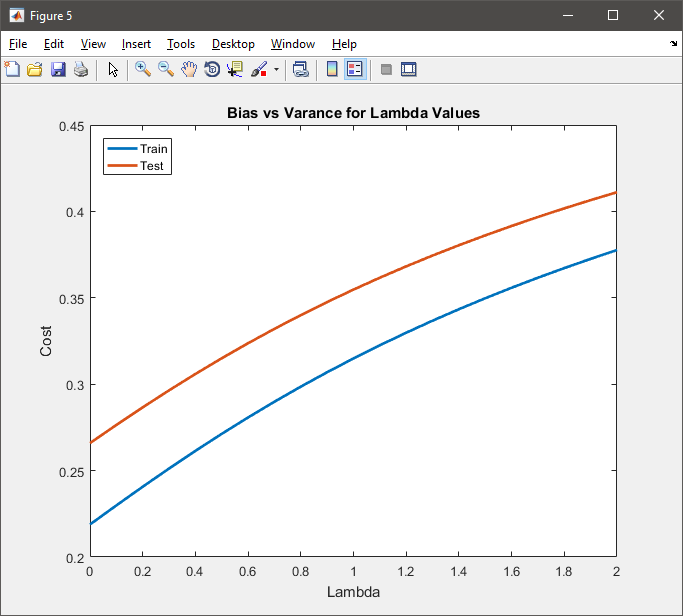
Then to choose the correct number of iteration was showed a plot for the gradient decent for each of the classes present in this experiment. The plot shows that, with 1400 iterations the gradient converges, and do not reach a significant better result if we increase the number of iterations.



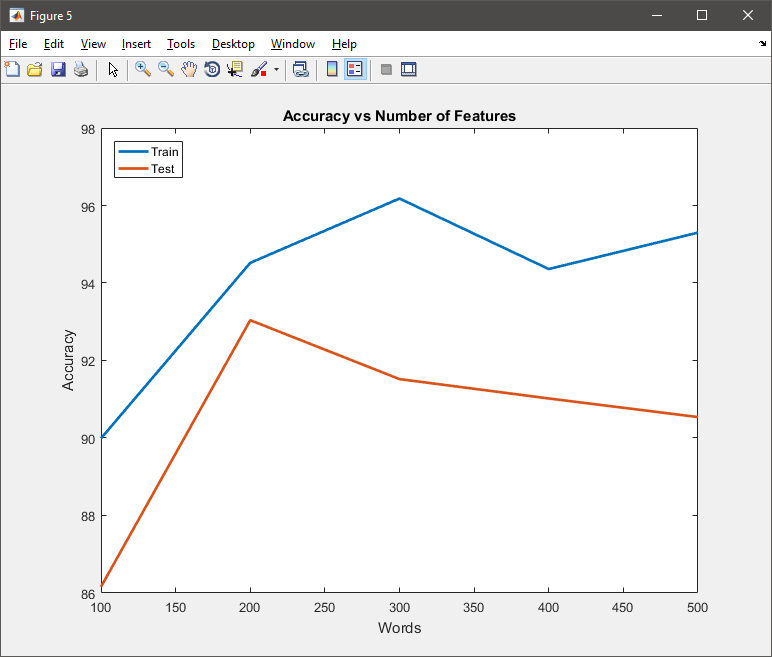
We can see a summary plot of the gradient decent for all three classes.



To evaluate the correct parameter lambda regularization to use, we calculated the cost function with different values of lambda, the following plot shows the result for lambda values from zero to two. This means that our model is too simple and the number of features is not enough to overfeed our model, choosing a lambda parameter of zero is the best option for this experiment.



To evaluate the correct number of words/features to use, we calculate the accuracy for the test and training data with different number of words, the following plot shows the result for words values from 0 to 500. The result shows that, a number of 200 words is adequate, less than that number is no enough features to describe what is the content of the image, and more that 200 we need more data to train our model.



**RESULTS**

## Caltech Dataset

Whit the parameter before mentioned was reached an 93.04% on the test accuracy, with a time for training steepest gradient descent of 36.64 seconds with a number of folds of 10. The biggest time was taken to calculate the words to use to encode each imagen, with a total of 696 seconds with is 12 min approx.

Part 1: Loading and Visualizing Data ...

Label Count

\_\_\_\_\_\_\_\_\_ \_\_\_\_\_

airplanes 67

ferry 67

laptop 67

(Done 1.079701)

Part 2: Extracting Features Bad of Words ...

Creating Bag-Of-Features from 1 image sets.

--------------------------------------------

\* Image set 1: .

\* Selecting feature point locations using the Grid method.

\* Extracting SURF features from the selected feature point locations.

\*\* The GridStep is [8 8] and the BlockWidth is [32 64 96 128].

\* Extracting features from 201 images in image set 1...done. Extracted 837424 features.

\* Keeping 80 percent of the strongest features from each image set.

\* Using K-Means clustering to create a 200 word visual vocabulary.

\* Number of features : 669939

\* Number of clusters (K) : 200

\* Initializing cluster centers...100.00%.

\* Clustering...completed 22/100 iterations (~24.66 seconds/iteration)...converged in 22 iterations.

\* Finished creating Bag-Of-Features

(Done 696.620450)

Number of features to use: 200.000000

Number of samples: 201.000000

Part 3: Encoding Images to Train and Test ...(Done 88.873059)

Part 4: Cross Validation Training Logistic Regression steepestGradientDescent...

Fold: 1.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 2.683715)

Fold: 2.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 5.908076)

Fold: 3.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 8.442178)

Fold: 4.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 11.511460)

Fold: 5.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 14.214850)

Fold: 6.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 18.316223)

Fold: 7.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 23.734119)

Fold: 8.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 28.268236)

Fold: 9.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 32.627425)

Fold: 10.000000

Training k: 1.000000

Training k: 2.000000

Training k: 3.000000(Done 36.640099)

Training Set Accuracy: 94.527624

Test Set Accuracy: 93.047619

With this result, we can conclude that the use of bag of words is a good method to pre-process images and low down the number of features without losing valuable information that is critical to a good classification, when it is used with logistic regression.