Evaluation of semantic labelling accuracy between multiple kernel functions using SVM classification

Tejas Vyas
Department of Computer Science
State University of New York at Buffalo
tejasvya@buffalo.edu

Pritesh Gupta
Department of Computer Science
State University of New York at Buffalo
priteshg@buffalo.edu

Abstract

The ability to classify objects in a given scene is a capability that many human beings take for granted. It's second nature to us to be able to classify an object. For computers, it's a lot harder.

In fact, a major problem in machine learning and computer vision is finding ways to help computers classify objects. This is a hard problem because a large variety of objects of the same type may exist, which makes it almost impossible to make a template. An object such as a phone may come in different shapes and sizes: maybe it's an old flip phone; maybe it's a brand new iPhone, or a landline phone etc.

One approach often used in machine learning is the idea of a support vector machine for proper classification. By using different types of kernel functions; an SVM basically assists a computer to determine if a given object exists in a particular scene. In this paper I will explain how linear kernels can be used on a support vector machine model, in order to provide a decent accuracy for classification. .

Introduction

Classifying image features is an extremely easy task for humans. Our brain is designed to process the images and segment them into labels and categorizes it automatically based on learning performed during the initials years based on the neighborhood we reside in. One of the examples may be right in front of us. Reading this paper, our eyes can comprehend the differences between various textures, colors and other features and label it accordingly, such as active and inactive buttons, page breaks, laptop screens, icons etc. Another example may be of a forest where our eyes are able to distinguish between different trees, sky, ground, animals and roads and label them accordingly which is then comprehended in our mind to show us a proper route.

Computers, on the other hand, do not have enough processing power to compute such labels at such a great rate and accuracy, and need various classifiers and training about features in order to classify different parts of an image taking copious amounts of time and with low accuracy. [1]

FIGURE 16.1: Material is not the same as object category (the three cars on the **top** are each made of different materials), and is not the same as texture (the three checkered objects on the **bottom** are made of different materials). Knowing the material that makes up an object gives us a useful description, somewhat distinct from its identity and its texture. *This figure was originally published as Figures 2 and 3 of "Exploring Features in a Bayesian Framework for Material Recognition," by C. Liu, L. Sharan, E. Adelson, and R. Rosenholtz Proc. CVPR 2010, 2010 © IEEE, 2010.*

In the given project, we were given the task to classify and analyze 720 benchmark image dataset based on the features by the 8 given labels. Each image in the dataset was already oversegmented into superpixels and the ground-truth labels for each superpixel have also been provided. We have eight (8) classes representing the semantic labels to which superpixels needed to be assigned. The classes included the following labels: sky, tree, road, grass, water, building, mountains, and foreground objects.

Our requirements for performance analysis required us to perform 5-fold cross-validation with the dataset, which was randomly split into different sets of 572 training images and 143 test images for each fold.

Related Works

The task of getting an accuracy comparable to human perception in the field of image feature representation is a very hard problem to tackle at the moment for the computers. The distinctions between the possible feature points vary tremendously. There are several categories where a feature point may not be based on the traditional mathematical curve. The number of computations to be performed in the task by a computer provide an exponential runtime with an extremely low accuracy in comparison to the feature segmentation performed by the humans.

Several researches have been conducted on the task which attempt to get an optimal solution to the segmentation problem. There are some researches which tackle the issue in a similar way by using a non-probabilistic classifier such as support vector machines.

2.1. Using Superpixels

Hoiem, Efros and Hebert in their research paper entitled *Recovering Surface Layout from an Image* tackle the issue using different set of superpixels as a support in order to perform multiple segmentation using different labelling queue in every attempt. Their agenda was to decompose the image scene using geometric class labelling. One possible issue with this implementation arises when an image has inconsistently divided feature points which make the geometric class labelling accuracy erratic. Being a geometric label, it is necessary to assume we have a partially smooth surface such that the spatial cues work correctly.

2.2. Using 3D reconstruction

Gould, Fulton and Koller in their research paper entitled *Decomposing a Scene into Geometric and Semantically Consistent Regions* tackle this issue from a slightly different point of view. Their model is defined in terms of a unified energy function over scene appearance and structure. Their primary objective was to recreate the scene by using 3d reconstruction and achieve state-of-the-art results for multi-class image segmentation and geometric reasoning.

2.3. Using MRF

Ren, Bo and Fox in their research paper entitled *RGB-(D) Scene Labeling: Features and Algorithms* use superpixels in order to perform spatial segmentation as well. Their attempt is slightly more adapted for the probabilistic classifiers as they use MRF i.e. Markov Random Fields in order to get an overview of the probabilistic distribution inconsistencies occurring in spatial orientation in a scene in order to decompose and create a more accurately segmented output.

Approach and Algorithm

We are given a set of image scenes and a default implementation SVM i.e. Support Vector Machine based non-probabilistic classifier which is fairly easy to implement. The implementation included a RBF kernel function i.e. a radial basis function which had a low accuracy.

The radial basis function may be defined as the following:

$$K(\mathbf{x}, \mathbf{x'}) = \exp\left(-\frac{||\mathbf{x} - \mathbf{x'}||^2}{2\sigma^2}\right)$$

By nature, the radial basis function can hold infinite dimensional mappings. However, the function by itself uses a Gaussian curve to rule our segmentation decision between multiple image features. It does not seem fairly intuitive for linearly classifiable data.

The algorithm consisted basic steps for segmentation with SVM model implemented in the end using a fitcsvm function. The analysis is done by running the function in 5 fold determined by the foldidx variable in the files.

Input:

- n_s: number of segments
- $\mathrm{p}_{ij} = \mathrm{P}(y_i = y_j|\mathrm{I})$: probability that adjacent superpixels r_i and r_j have same label, given the image

Initialize:

1. Assign n_s random superpixels to segments $1..n_s$
2. While any unassigned superpixels remain
    For each unassigned m_i:
        For each neighboring segment k:
            Assign $m_i = k$ with probability $\prod_j (\mathrm{p}_{ij}^{m_j=k}(1 - p_{ij})^{m_j \neq k})$

Fig: Algorithm description for train function

Our attempt to synthesize a more accurate output involved changing kernel functions in order to get better outputs. This was performed by changing the fitcsvm function in MATLAB which handles the SVM generation requests. We changed the training file trainSceneLabels.m line

SVMModel = fitcsvm(Ftrain, 2*(Ctrain==c)-1,'ClassNames',[1 -1], 'KernelFunction','*rbf*','Standardize',true);

In the above line, using multiple kernel functions gave us a set of different output confusion matrices which showed the accuracy.

Polynomial and quadratic kernel functions output a similar situation because they were unable to incorporate segmentation of superpixels for linearly separable points.

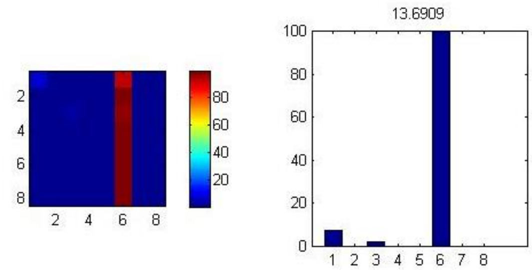
Our success came from using a linear kernel function defined as:

$$k(x, y) = x^T y + c$$

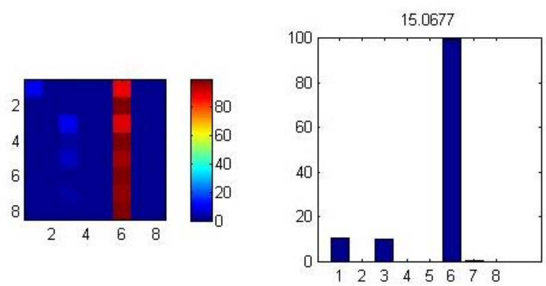
Our algorithm on addition of new kernals gave a good accuracy of 48% on 200000 super pixels on fold 2

4. Analysis and Results

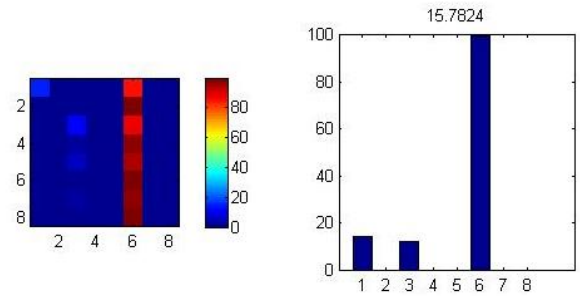
Our algorithm gave the following resulting confusion matrix and accuracy rates for the tests ran on foldidx = 4 using a Radial Basis Function for respective 10000,100000 and 200000 superpixels.



The above result is for super pixels 10000

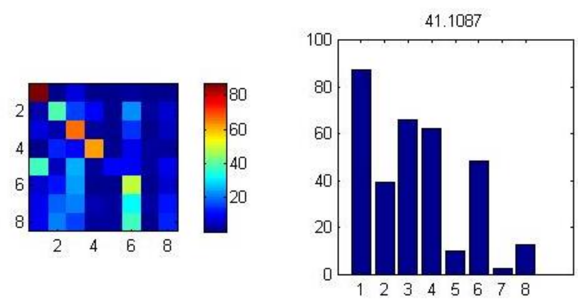


The accuracy gets a little better with the increase in super pixels. The above accuracy was for 100000.

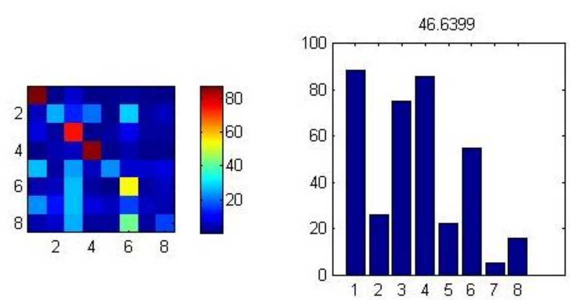


Finally, the best accuracy we got was for 200000 super pixels which was the maximum our workstation could handle.

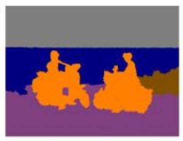
Below are the results after using the linear kernal function which show a 3 times increase for the 5 fold test where fold 2 generated the best results.



The output above is for super pixels 10000. The one below shows 200000.



The segmentation plot results also show a greatly detailed map.



5. Conclusion

So, in conclusion; we realize that the linear kernel gave me the highest accuracies overall. This leads me to think that the feature points were mostly linearly separable easily. This may not always be the case however. As discussed in lecture; there is no such thing as a perfect classifier/kernel function. The appropriate kernel truthfully depends on the layout of the data points. As I was unsure of what the layout of our feature points (superpixels), I tried a variety of kernel functions before arriving at the highest accuracy in linear kernel.

For humans it's easy to classify items in a scene and mentally label them accordingly. For computer systems, this classification process is extremely hard. Especially because so many different types and varieties of specific items can exist. This leads us to think that proper classification is a problem in machine learning and computer vision that will not be going anywhere anytime soon. Lots of improvements have been made in the past few years and we believe that in due time; computer vision experts will be able to develop methods to reach 100% accuracy for image classification.

References

[1] Class Notes Lecture 24

[2] X. Ren, L. Bo, and D. Fox, http://research.cs.washington.edu/istc/lfb/paper/cvpr12.pdf

[3] D. Hoiem, A. Efros, and M. Hebert, International Journal of Computer Vision 75(1),

[4] S. Gould, R. Fulton, D. Koller, Decomposing a Scene into Geometric and Semantically Consistent Regions