Self-Paced Learning for Semisupervised Image Classification

Kevin Miller
Stanford University
You Know Where Stanford Is
kjmiller@stanford.edu

Abstract

Future Distribution Permission

1. Introduction

In the problem at hand, there is a set of images $x_1, ..., x_M$, each of which contains one of 20 objects (e.g. cars, sheep, humans, etc.). Let $y_1, ..., y_M$ denote the object that each image contains (e.g. $y_1 = 1$ means that image 1 contains a car). During learning, x and y are known for each image, and during classification, we want to predict y from x.
The simplest approach would be to use a completely supervised algorithm, such as Structural SVM. However, semisupervised algorithms tend to perform much better by using latent variables to reduce noise. In this project, Latent Structural SVM is used; for each image x_i, the latent variable h_i represents the location and dimensions of a bounding box around (what is hopefully) the object in the picture.

One problem with Latent Structural SVM is the fact that training an LSSVM involves optimizing a non-convex function, meaning that the algorithm can end get stuck in bad local optima. Daphne Koller, M. Pawan Kumar, and Ben Packer recently showed that LSSVM could be improved by ignoring "difficult" examples during each inner loop of the LSSVM optimization algorithm; specifically, examples with slack above some threshold would be ignored, and between each inner loop this threshold would be increased until all examples were included and the algorithm converged. This extension of LSSVM is known as Self-Paced Learning.

My project will involve three goals:
1. Applying Self-Paced Learning to new datasets (e.g. recent VOC datasets, SUN dataset, ImageNet dataset) with a more sophisticated feature set and bounding-box model than that which Koller, Kumar, and Packer originally used.
2. Working on an extension of Self-Paced Learning in which particular feature-sets can be ignored for each image.
3. Working on an additional extension in which particular between-class constraints can be ignored for each image and feature-set (e.g. penalize car-sheep misclassifications before penalizing car-bus misclassifications).

Each dataset will be split into a train set and a test set, and test error will be used to evaluate each algorithm. Multiple training and testing folds may be used if time permits. Objective values and bounding box accuracies may also be plotted over time to compare the progress of the different algorithms.

References:

http://ai.stanford.edu/~pawan/publications/KPK-NIPS2010.html