Hierarchical Multiclass Object Classification

Fereshte Khani
Massachusetts Institute of Technology
fereshte@mit.edu

Anurag Mukkara
Massachusetts Institute of Technology
anurag_m@mit.edu

Abstract

Human can use similarity between objects in order to recognize rare objects. They also make many abstract concepts when they see some objects very often. Interestingly, a large part of brain is associated with common classes like faces rather than rare objects like Ostrich. In our work we want to propose a model that has four mentioned characteristics. 1. Use more resources for categories that have many examples and less resources for categories that have few examples. 2. Has the ability recognize objects that it has never seen before, but can be made by combining previously seen objects. 3. Objects with few examples can borrow statistical strength from similar objects with many examples. 4. Models that have lot of sharing between model parameters of different classes should be favored.

1. Introduction

2. Previous work

3. Hierarchical Classification Model

3.1. Learning Independent Classification Models

Consider a classification problem where we observe a dataset D of labelled training examples. Each example belongs belongs to one K classes(eg. 40 object categories). The dataset D has n_k labeled examples for for each class $k \in 1, 2, ..., K$. We $x_i^{(k)} \in R^D$ be the input feature vector of length D for the i^{th} training example of class k and $y_i^{(k)}$ be the corresponding class label. $y_i^k = 1$ indicates that the i^{th} training example is a positive example for class K and $y_i^k = -1$ indicates that the i^{th} training example is a negative example for class K.

For a binary classification problem we can use a simple logistic regression model. In particular, for each class k, the probability of a positive instance can be modeled as:

$$p(y_i^{(k)} = 1|\beta^{(k)}) = \frac{\exp(\beta^{(k)^T} x_i^{(k)})}{1 + \exp(\beta^{(k)^T} x_i^{(k)})} \quad (1)$$

where k ranges over the K classes and $\beta^k \in R^D$ is the unknown model parameter of length D for class k. In addition, we place a zero-mean spherical Gaussian prior over each model parameter β^k:

$$p(\beta) = \prod_{k=1}^{K} p(\beta^{(k)}) = \prod_{k=1}^{K} N(0, \frac{1}{\lambda}) \quad (2)$$

where $N(\mu, \Sigma)$ indicates a Gaussian probability distribution with mean μ and covariance Σ. Here, λ is the prior parameter. With this modeling framework, the log posterior distribution of the unknown parameters can be expresses as

$$\log p(\beta|D) \propto \sum_{k=1}^{K} \Big[\sum_{i=1}^{n_k} \log p(y_i^{(k)}|x_i^{(k)}, \beta^{(k)}) + \log p(\beta^{(k)}) \Big] \quad (3)$$

Maximizing the log posterior(or finding the MAP estimate) over the unknown model parameters β is equivalent to minimizing the probabilistic loss function with quadratic regularization terms:

$$E = \sum_{k=1}^{K} \Big[\sum_{i=1}^{n_k} Loss(y_i^{(k)}, x_i^{(k)}, \beta^{(k)}) + \frac{\lambda}{2} \|\beta^{(k)})\|^2 \Big] \quad (4)$$

where the probabilistic loss function is given by:

$$Loss(y, x, \beta) = - \sum_{j \in -1,1} I\{y = j\} \log p(y = j|x, \beta) \quad (5)$$

One important property of the loss function is that it is convex and hence can be efficiently optimized using standard convex optimization solvers.

The main drawback of this formulation is that all the K classes are treated as unrelated entities and hence do not share any model parameters. In fact, it is observed that the objective function decomposes into K sub-problems, each of which is a binary classification problem that can be trained independently using the training data for that particular class.

3.2. Learning a Hierarchical Classification Model

In our framework, a hierarchical tree structure is used to model sharing of parameters between visually related

classes. Each node in the tree has a model parameter, a vector of length D associated with it. Let $\theta = \{\theta_1, \theta_2, ..., \theta_n\}$ denote the model parameters of all the nodes in the tree, where n is the number of nodes starting from the root node to the all the leaf nodes. The leaf nodes correspond to the K object categories or classes. The parameters for a particular class, say k, is the sum of the parameters of all the nodes, say $i_1, i_2, ..., i_l$, along the path from the leaf node corresponding to that class up until the root node of the tree:

$$\beta^{(k)} = \theta_{i_1} + \theta_{i_2} + ... + \theta_{i_l} \tag{6}$$