CS771 Introduction to Machine Learning
Assignment 2

Group No. 7
Avesh Kumar Agrawal(19111020)
Kamlesh Kumar Biloniya(160317)
Sanjeev Lal(19111077)
Shashank Kumar(19111084)
Shrikant Jhajhra(19111089)

1 Brief Description of FastXML

1.1 Objective

The main objective is to develop an Extreme-Multi-Label classifier that is faster and more accurate at prediction than the state-of-the-art Multi-Label Random Forest (MLRF) Algorithm [1] and the Label-Partitioning for Sub-Linear Ranking (LPSR) Algorithm [2].

1.2 Overview

Traditionally tree growing loss functions generally learn a tree based on Gini index or Entropy or Hamming Loss which are not rank sensitive.

FastXML achieves this by learning a rank sensitive function which orders the items based on user preference.

The intuition is similar to LPSR and MLRF's and comes from the observation that only a small number of labels are present, or active, in each region of feature space.

Like MLRF and unlike LPSR, FastXML defines the set of labels active in a region to be the union of the labels of all training points present in that region. This speeds up training as FastXML does not need to solve the label assignment integer program in each region. Furthermore, like MLRF, and unlike LPSR, FastXML learns an ensemble of trees and does not need to rely on base classiffiers.

Predictions are made by returning the ranked list of most frequently occurring active labels in all the leaf nodes in the ensemble containing the novel point.

1.3 Learning to partition a node [3]

Training of FastXML consisted of partitioning a parent's feature space recursively between its children. A node partition is ideally learnt by optimizing a global measure of performance like ranking predictions which are induced by the leaf nodes.

So, hierarchy is to learn by fastXML by directly optimizing a ranking loss function. nDCG is a measure which is sensitive to both ranking and relevance and therefore ensures that the relevant positive labels are predicted with ranks that are as high as possible.

nDCG, is then defined as

$$\mathscr{L}nDCG@k(r,y) = I_k(y) \sum_{l=1}^{k} \frac{y_{rl}}{log(1+l)}$$

$$where\ I_k(y) = \frac{1}{\sum_{l=1}^{min(k,1^T y)} \frac{1}{log(1+l)}}$$

where $I_k(y)$ is the inverse of the $nDCG@k$ of the ideal ranking for y obtained by predicting the ranks of all of $\mathbf{y}$'s positive labels to be higher than any of its negative ones. This normalizes $\mathscr{L}nDCG@k$ to lie between 0 and 1 with larger values being better and ensures that $DCG@k$ can be used to compare rankings across label vectors with different numbers of positive labels.

FastXML partitions the current node's feature space by learning a linear separator w such that

$$min\ \ \|w\|_1 + \sum_i C_\delta(\delta_i) log(1 + e^{-\delta_i w^T x_i})$$

$$-C_r \sum_i \frac{1}{2}(1+\delta_i)\mathscr{L}nDCG_{@L}(r^+, y^i)$$

$$-C_r \sum_i \frac{1}{2}(1+\delta_i)\mathscr{L}nDCG_{@L}(r^-, y^i)$$

$$w.r.t.\ \ w \in R^D, \delta \in \{-1,+1\}^L, r^+, r^- \in \Pi(1,L)$$

where i indexes all the training points present at the node being partitioned, $\delta_i \in \{-1,+1\}$ indicates whether point i was assigned to the negative or positive partition and r^+ and r^- represent the predicted label rankings for the positive and negative partition respectively. C_δ and C_r are user defined parameters which determine the relative importance of the three terms.

2 Advantages and Disadvantages of FastXML

2.1 Advantages [4]

1. Logarithmic time prediction in milliseconds.
   - Ensemble of balance tree classifiers.
2. Accuracy gains upto 25% over competing methods.
   - nodes partitioned using nDCG.
3. Upto 1000x faster training over the state-of-the-art.
   - Alternating minimization based optimization.
   - Proof of convergence towards stationary point.

2.2 Disadvantage

1. nDcg is hard to optimize.
   - nDCG is non-convex and non-Smooth.
   - Large input variation $\implies$ no chanfge in nDCG.
   - small input variations $\implies$ large changes in nDCG.

3 Statistics of chosen method

Number of trees grown: 7
Feature bias value: 1.5
SVM weight co-efficient: 2
Number of label-probability pairs retained in a leaf: 50
Maximum allowed instances in a leaf node: 5

	@1	@3	@5
prec	0.992	0.957	0.931
mprec	3.475e-03	2.988e-02	9.956e-02

Table 1: mprec and prec

4 Suggested Method

4.1 FastXML - chosen method without any modifications

1. While optimizing nDCG, millions of extra variables which were embedded in δ_i that were taking some extra time to optimize. So, we tried to eliminate these variables and directly tied to optimize nDCG but the time to optimize it further increased. So, we decided to use δ_i in nDCG and left our modification.
2. We tried a novel probabilistic hierarchical multi-label model for efficient training and prediction at a cost of only $1\% - 2\%$ loss in accuracy in many cases. But, it increases the model size significantly. So, we decided not to implement it.

5 Code for FastXML model [3]

`https://cse.iitk.ac.in/users/aavesh/cs771/submit.zip`

References

[1] R. Agrawal, A. Gupta, Y. Prabhu, and M. Varma, *Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages* In WWW, pages 13–24, 2013

[2] J. Weston, A. Makadia, and H. Yee. *Label partitioning for sublinear ranking.* In ICML, volume 28, pages 181–189, 2013

[3] Y. Prabhu, and M. Varma, *FastXML: A Fast, Accurate and Stable Tree-classifier for eXtreme Multi-label Learning* in KDD, 2014.

[4] Extreme Classification: A New Paradigm for Ranking Recommendation, *https://www.youtube.com/watch?v=1X71fTx1LKA&t=1439s* in Jun 22, 2016.