Supplementary - Parabel: Partitioned Label Trees for Extreme Classification with Application to Dynamic Search Advertising

Yashoteja Prabhu*
yashoteja.prabhu@gmail.com

Anil Kag†
anilkagak2@gmail.com

Shrutendra Harsola‡
shharsol@microsoft.com

Rahul Agrawal‡
Rahul.agrawal@microsoft.com

Manik Varma*†
manik@microsoft.com

[ ]

1 ALGORITHMS

Algorithm 1 presents the pseudocode for Parabel training by assuming 1-vs-All classifiers for both the child traversal distributions in the internal nodes and the label sampling distributions in the leaf nodes. Note, however, that in general any probabilistic classifiers can be used as alternatives to 1-vs-All classifiers. Algorithm 3 presents the pseudocode for Parabel prediction algorithm. Algorithm 2 describes the label clustering algorithm used for learning the balanced binary label trees.

Algorithm 1 Parabel Training

Input: (a) Training data $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$; (b) number of trees T; (c) maximum labels in a leaf M; (c) cost co-efficient of linear classifiers C
Output: Trained trees $\mathcal{T}_1, .., \mathcal{T}_T$

for $t \in \{1, .., T\}$ **do**
  $\mathcal{T}_t \leftarrow$ TRAIN TREE$(\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N, M, C)$
end for
return $\{\mathcal{T}_1, .., \mathcal{T}_t\}$

procedure TRAIN TREE$(\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N, M, C)$
  $\mathbf{v}'_l \leftarrow \sum_{l=1}^L y_{il}\mathbf{x}_i \ \forall l \in \{1, .., L\}$ # L is number of labels
  $\mathbf{v}_l = \frac{\mathbf{v}'_l}{\|\mathbf{v}'_l\|_2} \ \forall l \in \{1, .., L\}$ # Label representations are formed
  $\mathcal{T} \leftarrow$ HIERARCHICAL SPHERICAL BALANCED k = 2-MEANS$(\{\mathbf{v}_l\}_{l=1}^L, M)$
  $\mathcal{I} \leftarrow \mathcal{T}.\mathcal{I}$; $\mathcal{L} \leftarrow \mathcal{T}.\mathcal{L}$; $\mathcal{Y} \leftarrow \mathcal{T}.\mathcal{Y}$; $\mathcal{C} \leftarrow \mathcal{T}.\mathcal{C}$ # $\mathcal{I}$, $\mathcal{L}$, Y_n, C_n are the internal nodes, leaf nodes, labels in node n and children of node n in tree $\mathcal{T}$
  for $n \in \mathcal{I}$ **do** # Iterate over internal nodes
    $\mathcal{X}_n \leftarrow \{i : \sum_{l \in \mathcal{Y}_n} y_{il} > 0\}$ # $\mathcal{X}_n$ is set of data points active in node n, y_{il} is the lth element of $\mathbf{y}_i$
    for $\hat{n} \in \mathcal{C}_n$ **do** # Iterate over children of node n
      $\mathcal{X}_{\hat{n}} \leftarrow \{i : \sum_{l \in \mathcal{Y}_{\hat{n}}} y_{il} > 0\}$
      for $i \in \mathcal{X}_n$ **do**
        $z_i \leftarrow \mathbb{1}[i \in \mathcal{X}_{\hat{n}}]$ # $\mathbb{1}$ is indicator function which takes values in $\{0.1\}$
      end for
      $\mathcal{T}.\mathbf{w}_{n\hat{n}} \leftarrow \arg\min_{\mathbf{w}} \|\mathbf{w}\|^2 + C \sum_{i \in \mathcal{X}_n} \mathcal{F}((2z_i - 1)\mathbf{w}^\top \mathbf{x}_i)$ # $\mathcal{F}$ can be log loss or squared hinge loss *etc.*
    end for
  end for
  for $n \in \mathcal{L}$ **do** # Iterate over leaf nodes
    $\mathcal{X}_n \leftarrow \{i : \sum_{l \in \mathcal{Y}_n} y_{il} > 0\}$
    for $l \in \mathcal{Y}_n$ **do**
      $\mathcal{T}.\mathbf{w}_{nl} \leftarrow \arg\min_{\mathbf{w}} \|\mathbf{w}\|^2 + C \sum_{i \in \mathcal{X}_n} \mathcal{F}((2y_{il} - 1)\mathbf{w}^\top \mathbf{x}_i)$ # Learn 1-vs-All for labels in leaf node n
    end for
  end for
  $\mathcal{T}.\text{depth} \leftarrow \lceil \log_2(\frac{L}{M}) \rceil$
return $\mathcal{T}$
end procedure

*Indian Institute of Technology Delhi
†Microsoft Research India
‡Microsoft Bing Ads

2 THEOREMS AND PROOFS

2.1 Spherical Balanced $k = 2$-Means Clustering

Label partitioning in Parabel involves solving the following Spherical Balanced $k = 2$-Means Clustering problem

$$\max_{\boldsymbol{\mu}_\pm \in \mathbb{R}^D, \boldsymbol{\alpha} \in \{-1,+1\}^L} \frac{1}{L} \sum_{l=1}^{L} \left(\frac{1+\alpha_l}{2} \boldsymbol{\mu}_+^\top \mathbf{v}_l + \frac{1-\alpha_l}{2} \boldsymbol{\mu}_-^\top \mathbf{v}_l \right) \tag{1}$$

$$\text{s. t.} \quad \|\boldsymbol{\mu}_\pm\|_2 = 1, \quad -1 \leq \sum_{l=1}^{L} \alpha_l \leq 1$$

where $\mathbf{v}_l, \alpha_l$ are respectively the feature vector and the cluster assignment variable for the l^{th} label; $\boldsymbol{\mu}_\pm$ are the means for the positive (left) and the negative (right) clusters.

The optimization problem in (1) is NP-hard [1]. Parabel therefore employs the following alternating minimization algorithm which converges to a local optimum. The algorithm is initialized by sampling $\boldsymbol{\mu}_\pm$ from $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_L\}$ uniformly at random without replacement. The following two steps are then repeated in each iteration of the algorithm until convergence. In the first step of each iteration, (1) is maximized while keeping $\boldsymbol{\mu}_\pm$ fixed. It is straightforward to show that the optimal solution is given by $\alpha_l^* = \text{sign}\Big(\text{rank}\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) - \frac{L+1}{2}\Big)$ with sign(0) being resolved to $+1$ or -1 depending on whether the label is closer to the positive or negative cluster respectively. In the second step, each α_l is fixed and (1) is maximized with respect to $\boldsymbol{\mu}_\pm$ to get $\boldsymbol{\mu}_\pm^* = \boldsymbol{\mu}_\pm' / \|\boldsymbol{\mu}_\pm'\|_2$ where $\boldsymbol{\mu}_\pm' = \sum_{l:\alpha_l = \pm 1} \mathbf{v}_l$. The following theorems show that the closed form solutions obtained for these two steps are in fact the global maxima for their respective subproblems.

Theorem 2.1. *In (1), if $\boldsymbol{\mu}_\pm$ are kept fixed and the objective is optimized with respect to $\boldsymbol{\alpha}$, then the solution $\alpha_l^* = sign\Big(rank\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) - \frac{L+1}{2}\Big)$ when $rank\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) \neq \frac{L+1}{2}$, and $\alpha_l^* = sign\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big)$ when $rank\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) = \frac{L+1}{2}$ is a global maximum.*

Proof. Let us prove by contradiction. First, we can see that the purported solution

$$\alpha_l^* = \text{sign}\Big(\text{rank}\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) - \frac{L+1}{2}\Big) \ \textbf{if} \ \text{rank}\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) \neq \frac{L+1}{2} \tag{2}$$

$$\alpha_l^* = \text{sign}\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) \ \textbf{if} \ \text{rank}\Big((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l\Big) = \frac{L+1}{2} \tag{3}$$

obeys the balance constraint on $\boldsymbol{\alpha}$ and hence is a feasible solution to

$$\max_{\boldsymbol{\alpha} \in \{-1,+1\}^L} F(\boldsymbol{\alpha}) = \frac{1}{L} \sum_{l=1}^{L} \left(\frac{1+\alpha_l}{2} \boldsymbol{\mu}_+^\top \mathbf{v}_l + \frac{1-\alpha_l}{2} \boldsymbol{\mu}_-^\top \mathbf{v}_l \right) \tag{4}$$

$$\text{s. t.} \quad -1 \leq \sum_{l=1}^{L} \alpha_l \leq 1$$

Now, if the solution (2) is not a global maximum of the objective (4), then there exists a true global maximum $\overline{\boldsymbol{\alpha}} \in \{-1,+1\}^L$ such that $\overline{\boldsymbol{\alpha}} \neq \boldsymbol{\alpha}^*$, $F(\overline{\boldsymbol{\alpha}}) > F(\boldsymbol{\alpha}^*)$ and $\overline{\boldsymbol{\alpha}}$ has the minimum hamming distance $\frac{\|\overline{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_1}{2}$ among all the globally maximal solutions.

Case 1: $\frac{\|\overline{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_1}{2} = 1$

In this case, L is an odd number and $\overline{\boldsymbol{\alpha}}$ and $\boldsymbol{\alpha}^*$ differ in exactly one place, say $l \in \{1, \cdots, L\}$. Then l is definitely allotted to the odd and bigger sized cluster in $\overline{\boldsymbol{\alpha}}$ because otherwise the difference between the cluster sizes would be more than 1 thus disobeying the balance constraint. Moreover, l is closer to its own cluster (in terms of cosine similarity) than the other one, otherwise we could shift l to other cluster to get $\boldsymbol{\alpha}^*$ without reducing the objective, which leads to contradiction. Without loss of generality, let's assume that l is allotted to the positive cluster. Then l is ranked strictly after $\frac{L+1}{2}$ in $\boldsymbol{\alpha}^*$ in terms of $(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_l$. In this case, exchanging the label at $\frac{L+1}{2}$ position (which belongs to negative cluster in $\overline{\boldsymbol{\alpha}}$) and the label l would give a higher valued feasible solution leading to contradiction. A similar argument holds if l were to belong to negative cluster in $\overline{\boldsymbol{\alpha}}$. Therefore no such global maximum exists whose value is greater than $F(\boldsymbol{\alpha}^*)$ and whose hamming distance is 1.

Case 2: $\frac{\|\overline{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_1}{2} > 1$

In this case, there will be atleast one pair of labels $l1$ and $l2$ which are in different clusters in $\overline{\boldsymbol{\alpha}}$ and whose assignments are exchanged in $\boldsymbol{\alpha}^*$ as compared to $\overline{\boldsymbol{\alpha}}$. W.l.o.g let $l1$ be allotted to negative cluster and $l2$ to positive cluster in $\overline{\boldsymbol{\alpha}}$. Then $l1$ will be allotted to positive cluster and $l2$ to negative cluster in $\boldsymbol{\alpha}^*$. Consequently,

$$(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_{l1} \geq (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^\top \mathbf{v}_{l2} \tag{5}$$

$$\implies \boldsymbol{\mu}_+^\top \mathbf{v}_{l1} + \boldsymbol{\mu}_-^\top \mathbf{v}_{l2} \geq \boldsymbol{\mu}_+^\top \mathbf{v}_{l2} + \boldsymbol{\mu}_-^\top \mathbf{v}_{l1} \tag{6}$$

Algorithm 2 Hierarchical Spherical Balanced $k = 2$-Means

Input: (a) Label feature vectors $\{\mathbf{v}_l\}_{l=1}^{L}$; (b) maximum labels in a leaf M
Output: Label tree $\mathcal{T}$

$\mathcal{T} \leftarrow$ new tree
$n_0 \leftarrow$ new node # n_0 is the root node
$\mathcal{Y}_{n_0} \leftarrow \{1, \cdots, L\}$ # Root node contains all labels
$\mathcal{I} \leftarrow \{n_0\}$ # Set of internal nodes
$\mathcal{L} \leftarrow \phi$ # Set of leaf nodes
$\mathcal{C}_{n_0} \leftarrow \phi$ # Children of node n_0
$\mathcal{P}_{n_0} \leftarrow \phi$ # Parent of node n_0
GROW-NODE-RECURSIVE$(\mathcal{I}, \mathcal{L}, \mathcal{Y}, \mathcal{C}, \mathcal{P}, n_0, \{\mathbf{v}_l\}_{l=1}^{L}, M)$
$\mathcal{T}.\mathcal{I} \leftarrow \mathcal{I}$; $\mathcal{T}.\mathcal{L} \leftarrow \mathcal{L}$; $\mathcal{T}.\mathcal{C} \leftarrow \mathcal{C}$; $\mathcal{T}.\mathcal{P} \leftarrow \mathcal{P}$
return $\mathcal{T}$

procedure GROW-NODE-RECURSIVE$(\mathcal{I}, \mathcal{L}, \mathcal{Y}, \mathcal{C}, \mathcal{P}, n, \{\mathbf{v}_l\}_{l=1}^{L}, M)$
  if $|\mathcal{Y}_n| \leq M$ **then** # n contains less than M labels, make n a leaf
    $\mathcal{C}_n \leftarrow \phi$ # n has no children nodes
    $\mathcal{L} \leftarrow \mathcal{L} \cup \{n\}$
  else # Split node and grow child nodes recursively
    $n_+ \leftarrow$ new node # n_+ is left child node
    $n_- \leftarrow$ new node # n_- is right child node
    $\mathcal{Y}_{n_+}, \mathcal{Y}_{n_-} \leftarrow$ SPHERICAL BALANCED $k = 2$-MEANS$(\mathcal{Y}_n, \{\mathbf{v}_l\}_{l=1}^{L})$
    $\mathcal{C}_n \leftarrow \{n_+, n_-\}$ # $\mathcal{C}_n$ are children of n
    $\mathcal{P}_{n_+} \leftarrow n$ # $\mathcal{P}_{n_+}$ is parent of n_+
    $\mathcal{P}_{n_-} \leftarrow n$
    $\mathcal{I} \leftarrow \mathcal{I} \cup \{n_+, n_-\}$
    GROW-NODE-RECURSIVE$(\mathcal{I}, \mathcal{L}, \mathcal{Y}, \mathcal{C}, \mathcal{P}, n_+, \{\mathbf{v}_l\}_{l=1}^{L}, M)$
    GROW-NODE-RECURSIVE$(\mathcal{I}, \mathcal{L}, \mathcal{Y}, \mathcal{C}, \mathcal{P}, n_-, \{\mathbf{v}_l\}_{l=1}^{L}, M)$
  end if
end procedure

procedure SPHERICAL BALANCED $k = 2$-MEANS$(\mathcal{Y}_n, \{\mathbf{v}_l\}_{l=1}^{L})$
  $\boldsymbol{\mu}_+, \boldsymbol{\mu}_- \sim \text{Unif}(\{\mathbf{v}_l \; \forall l \in \mathcal{Y}_n\})$ # Initialize means by uniform sampling from label features without replacement
  $\mathcal{Y}_{n_+} \leftarrow \phi$; $\mathcal{Y}_{n_-} \leftarrow \phi$
  do
    $s_l \leftarrow \boldsymbol{\mu}_+^\top \mathbf{v}_l - \boldsymbol{\mu}_-^\top \mathbf{v}_l \; \forall l \in \mathcal{Y}_n$
    $\overline{\mathcal{Y}}_{n_+} \leftarrow \mathcal{Y}_{n_+}$; $\overline{\mathcal{Y}}_{n_-} \leftarrow \mathcal{Y}_{n_-}$
    $\mathcal{Y}_{n_+} \leftarrow \phi$; $\mathcal{Y}_{n_-} \leftarrow \phi$
    $\{r_l\}_{l \in \mathcal{Y}_n} \leftarrow \text{argsort}(\{-s_l\}_{l \in \mathcal{Y}_n})$ # Sorting s_l in decreasing order. r_l indicates how many labels are scoring higher than l
    for $l \in \mathcal{Y}_n$ **do**
      if $r_l < \frac{|\mathcal{Y}_n|}{2}$ **then**
        $\mathcal{Y}_{n_+} \leftarrow \mathcal{Y}_{n_+} \cup \{l\}$
      else if $r_l > \frac{|\mathcal{Y}_n|}{2}$ **then**
        $\mathcal{Y}_{n_-} \leftarrow \mathcal{Y}_{n_-} \cup \{l\}$
      else
        $\mathcal{Y}_{n_{\text{sign}(s_l)}} \leftarrow \mathcal{Y}_{n_{\text{sign}(s_l)}} \cup \{l\}$
      end if
    end for
    $\boldsymbol{\mu}_+ \leftarrow \frac{\sum_{l \in \mathcal{Y}_{n_+}} \mathbf{v}_l}{\| \sum_{l \in \mathcal{Y}_{n_+}} \mathbf{v}_l \|_2}$
    $\boldsymbol{\mu}_- \leftarrow \frac{\sum_{l \in \mathcal{Y}_{n_-}} \mathbf{v}_l}{\| \sum_{l \in \mathcal{Y}_{n_-}} \mathbf{v}_l \|}$
  while $\mathcal{Y}_{n_+} \neq \overline{\mathcal{Y}}_{n_+} || \mathcal{Y}_{n_-} \neq \overline{\mathcal{Y}}_{n_-}$
return $\mathcal{Y}_{n_+}, \mathcal{Y}_{n_-}$
end procedure

Algorithm 3 Parabel Prediction

Input: (a) Test data point $\mathbf{x}$; (b) trained trees $\mathcal{T}_1, .., \mathcal{T}_T$; (c) beam search width P
Output: Ranking over labels R

for $t \in \{1, .., T\}$ **do**
  depth $\leftarrow \mathcal{T}_t$.depth
  $n_0 \leftarrow \mathcal{T}_t$.root
  $\mathcal{B} \leftarrow \{n_0\}$ # Set of boundary nodes which are maintained for beam search
  $LL_{n_0} \leftarrow 0$ # LL_{n_0} is log-likelihood of visiting the node n_0
  for $d = 0;\ d <$ depth $- 1;\ d++$ **do**
    $\overline{\mathcal{B}} \leftarrow \mathcal{B}$
    $\mathcal{B} \leftarrow \phi$
    for $n \in \overline{\mathcal{B}}$ **do**
      for $n_c \in C_n$ **do** # Iterate over children of node n
        $LL_{n_c} \leftarrow -\mathcal{L}(\mathbf{w}_{nn_c}^\top \mathbf{x}) + LL_n$
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{n_c\}$
      end for
    end for
    $\mathcal{B} \leftarrow$RETAIN TOP$(\mathcal{B}, LL, P)$
  end for
  for $n \in \mathcal{B}$ **do** # Iterate over visited leaf nodes
    for $l \in \mathcal{Y}_n$ **do** # Iterate over labels in leaf node n
      $\mathbb{P}_l \leftarrow \mathbb{P}_l + \exp(-\mathcal{L}(\mathbf{w}_{nl}^\top \mathbf{x}) + LL_n)$ # Sum up the marginal probabilities of a label l being relevant to data point $\mathbf{x}$ estimated by all trees
    end for
  end for
end for
$R \leftarrow$ rank$(\{\mathbb{P}_l\})$ # Rank the active labels in decreasing order of their aggregate marginal probabilities
return R

procedure RETAIN TOP$(\mathcal{B}, LL, P)$
  $R \leftarrow$ argsort$(\mathcal{B}, \text{comparator} \leftarrow LL_{n_1} > LL_{n_2})$ # Sort the boundary nodes in decreasing order of their log-likelihoods
  $\mathcal{B} \leftarrow \{R[1], .., R[P]\}$
return $\mathcal{B}$
end procedure

Exchanging $l1$ and $l2$ in $\overline{\boldsymbol{\alpha}}$ would reduce the hamming distance $\frac{\|\overline{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^*\|_1}{2}$ without increasing the objective, thus leading to contradiction. Therefore no such global maximum exists whose value is greater than $F(\boldsymbol{\alpha}^*)$ and whose hamming distance is greater than 1.

Since the above 2 cases cover all alternative possibilities and show that each alternative results in a contradiction, we have proved that $\boldsymbol{\alpha}^*$ is in fact a global maximum. □

The optimality for the case of optimizing (1) w.r.t $\boldsymbol{\mu}_\pm$ while keeping $\boldsymbol{\alpha}$ fixed is easily seen by observing that the problem reduces to two simple, independent quadratic equations one for each of $\boldsymbol{\mu}_+$ and $\boldsymbol{\mu}_-$.

THEOREM 2.2. *The Spherical Balanced $k = 2$-means clustering algorithm terminates in a finite number of iterations at a cluster assignment that is locally optimal.*

PROOF. Since in each iteration, both the steps of alternating maximization increase the objective in (1), no configuration of $\boldsymbol{\alpha}$ is going to repeat at the end of 2 distinct iterations. Since there are finite number of total configurations of $\boldsymbol{\alpha}$, equal to $\binom{L}{\lfloor \frac{L}{2} \rfloor}$, the algorithm will terminate in finite number of iterations. The algorithm terminates when a round of iteration, consisting of two optimization steps, fail to make any progress in maximizing the objective. In such a case, changing no single parameter amongst $\boldsymbol{\mu}_\pm, \boldsymbol{\alpha}$ can increase the objective. Hence, a local maximum is reached. □

2.2 A Hierarchical Probabilistic Model

The Parabel's probabilistic model is given by

$$\mathbb{P}(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{z}} \mathbb{P}(\mathbf{y}|\mathbf{z},\mathbf{x})\mathbb{P}(\mathbf{z}|\mathbf{x}) \tag{7}$$

$$= \sum_{\mathbf{z}\in\mathcal{Z}_\mathbf{y}} \prod_{n\in\mathcal{L}:z_n=1} \mathbb{P}(\mathbf{y}_n|z_n=1,\mathbf{x}) \prod_{n\in\mathcal{I}:z_n=1} \mathbb{P}(\mathbf{z}_{C_n}|z_n=1,\mathbf{x}) \tag{8}$$

where $\mathcal{Z}_\mathbf{y}$ denotes the set of all the configurations of $\mathbf{z}$ which could have led to $\mathbf{y}$ being sampled. The model is based on the following assumptions and theorem.

Unvisited node assumption: This assumption formalizes the observation that the children of an unvisited internal node will never be traversed and that the labels in an unvisited leaf node will never be sampled. This implies that

$$\mathbb{P}(\mathbf{z}_{C_n} = \mathbf{0}|z_n = 0,\mathbf{x}) = 1 \quad \forall n \in \mathcal{I} \tag{9}$$

$$\mathbb{P}(\mathbf{y}_n = \mathbf{0}|z_n = 0,\mathbf{x}) = 1 \quad \forall n \in \mathcal{L} \tag{10}$$

The set of $\mathbf{z}$ values which obeys this assumption is denoted by $\mathcal{Z}_\mathbf{y}$.

Subtree independence assumptions: Parabel assumes that the probability distribution at a visited node n, whether internal or leaf, is sampled independently of all the nodes that are outside the subtree rooted at node n such that

$$\mathbb{P}(\mathbf{z}_{C_n}|z_n=1,\mathbf{x}) = \mathbb{P}(\mathbf{z}_{C_n}|z_n=1,\mathbf{x},\bar{\mathbf{y}}_{\mathcal{S}_n},\bar{\mathbf{z}}_{\mathcal{S}_n}) \quad \forall n \in \mathcal{I} \tag{11}$$

$$\mathbb{P}(\mathbf{y}_n|z_n=1,\mathbf{x}) = \mathbb{P}(\mathbf{y}_n|z_n=1,\mathbf{x},\bar{\mathbf{y}}_{\mathcal{S}_n},\bar{\mathbf{z}}_{\mathcal{S}_n}) \quad \forall n \in \mathcal{L} \tag{12}$$

where $\bar{\mathbf{y}}_{\mathcal{S}_n}$ and $\bar{\mathbf{z}}_{\mathcal{S}_n}$ denote the sets of all the labels and all the z variables that lie outside the subtree rooted at node n.

Theorem 2.3. **Tree factorization**: *Given a label tree, if the subtree independence and unvisited node assumptions hold at all the tree nodes, then for a label vector* $\mathbf{y}$ *and an indicator vector* $\mathbf{z} \in \mathcal{Z}_\mathbf{y}$

$$\mathbb{P}(\mathbf{y}|\mathbf{z},\mathbf{x}) = \prod_{n\in\mathcal{L}:z_n=1} \mathbb{P}(\mathbf{y}_n|z_n=1,\mathbf{x}) \tag{13}$$

$$\mathbb{P}(\mathbf{z}|\mathbf{x}) = \prod_{n\in\mathcal{I}:z_n=1} \mathbb{P}(\mathbf{z}_{C_n}|z_n=1,\mathbf{x}) \tag{14}$$

Proof. **Claim 1**:

$$\mathbb{P}(\mathbf{y}|\mathbf{z},\mathbf{x}) = \mathbb{P}(\bigcup_{n\in\mathcal{L}} \mathbf{y}_n|\mathbf{z},\mathbf{x}) \tag{15}$$

$$= \prod_{n\in\mathcal{L}} \mathbb{P}(\mathbf{y}_n|\mathbf{y}_1,\cdots,\mathbf{y}_{n-1},\mathbf{z},\mathbf{x}) \quad \text{–by chain rule of probability} \tag{16}$$

$$\text{Since from the unvisited node assumption } z_n = 0 \implies \mathbf{y}_n = \mathbf{0} \text{ with probability 1} \tag{17}$$

$$= \prod_{n\in\mathcal{L}:z_n=1} \mathbb{P}(\mathbf{y}_n|\mathbf{y}_1,\cdots,\mathbf{y}_{n-1},\mathbf{z},\mathbf{x}) \tag{18}$$

$$= \prod_{n\in\mathcal{L}:z_n=1} \mathbb{P}(\mathbf{y}_n|\mathbf{y}_1,\cdots,\mathbf{y}_{n-1},z_n=1,\bar{\mathbf{z}}_{\mathcal{S}_n},\mathbf{x}) \quad \text{–because } \mathbf{z}_{\mathcal{S}_n} = z_n \text{ for a leaf node} \tag{19}$$

$$= \prod_{n\in\mathcal{L}:z_n=1} \mathbb{P}(\mathbf{y}_n|z_n=1,\mathbf{x}) \quad \text{–by applying (11)} \tag{20}$$

Claim 2:

$$\mathbb{P}(\mathbf{z}|\mathbf{x}) = \prod_{n\in\mathcal{I}:z_n=1} \mathbb{P}(\mathbf{z}_{C_n}|z_n=1,\mathbf{x}) \tag{21}$$

Let's prove the above claim by using the principle of induction. Consider a subtree $\mathcal{T}$ of the given label tree such that $\mathcal{T}$ shares the same root as the given label tree. Let $\mathbf{z}(\mathcal{T}), N(\mathcal{T})$ and $\mathcal{I}(\mathcal{T})$ be the set of indicator variables, number of nodes and set of internal nodes of the subtree $\mathcal{T}$. We will use induction over $N(\mathcal{T})$.

Hypothesis: For any subtree $\mathcal{T}$

$$\mathbb{P}(\mathbf{z}(\mathcal{T})|\mathbf{x}) = \prod_{n\in\mathcal{I}(\mathcal{T}):z_n=1} \mathbb{P}(\mathbf{z}_{C_n}|z_n=1,\mathbf{x}) \tag{22}$$

Let us prove the hypothesis through induction on the number of nodes $N(\mathcal{T})$ of the subtree.

Base case: The simplest subtree $\mathcal{T}$ with $N(\mathcal{T}) = 1$ contains just the root node. For this case, $\mathbf{z}(\mathcal{T}) = \{z_{\text{root}}\}$ and $\mathcal{I}(\mathcal{T}) = \phi$. We assume $z_{\text{root}} = 1 \forall \mathbf{z} \in \mathcal{Z}_{\mathbf{y}}$ to be always true, since tree traversal always begins at the root node. Consequently,

$$\mathbb{P}(\mathbf{z}(\mathcal{T})|\mathbf{x}) = \mathbb{P}(z_{\text{root}} = 1|\mathbf{x}) \tag{23}$$

$$= 1 \tag{24}$$

$$= 1 * \prod_{n \in \phi : z_n = 1} \mathbb{P}(\mathbf{z}_{C_n}|z_n = 1, \mathbf{x}) \tag{25}$$

Thus the base case holds true.

Induction: Consider a subtree $\mathcal{T}$ with the same root as the given label tree and $N(\mathcal{T}) > 1$. Let us do a pre-order traversal of $\mathcal{T}$ and let $\hat{n}$ be the last **internal** node that is visited during the traversal. Furthermore, let $\overline{\mathcal{T}}$ be another subtree that is got by removing all the children of node $\hat{n}$, which naturally occur after $\hat{n}$ during the traversal.

Let's assume that the hypothesis holds true for all the subtrees with number of nodes less than $N(\mathcal{T})$. Then it also holds true for the subtree $\overline{\mathcal{T}}$. Now

$$\mathbb{P}(\mathbf{z}(\mathcal{T})|\mathbf{x}) = \mathbb{P}(\mathbf{z}_{C_{\hat{n}}}|z_{\hat{n}}, \mathbf{z} \setminus \{z_{\hat{n}}, \mathbf{z}_{C_{\hat{n}}}\}, \mathbf{x})\ \mathbb{P}(\mathbf{z} \setminus \mathbf{z}_{C_{\hat{n}}}|\mathbf{x}) \quad \text{–by chain rule of probability} \tag{26}$$

$$\tag{27}$$

By applying hypothesis to $\overline{\mathcal{T}}$,

$$\mathbb{P}(\mathbf{z} \setminus \mathbf{z}_{C_{\hat{n}}}|\mathbf{x}) = \prod_{n \in \mathcal{I}(\overline{\mathcal{T}}) : z_n = 1} \mathbb{P}(\mathbf{z}_{C_n}|z_n = 1, \mathbf{x}) \tag{28}$$

If $z_{\hat{n}} = 0$ then by (9), $\mathbf{z}_{C_{\hat{n}}} = \mathbf{0}$, leading to

$$\mathbb{P}(\mathbf{z}(\mathcal{T})|\mathbf{x}) = \mathbb{P}(\mathbf{z}_{C_{\hat{n}}}|z_{\hat{n}} = 0, \mathbf{z} \setminus \{z_{\hat{n}}, \mathbf{z}_{C_{\hat{n}}}\}, \mathbf{x})\ \mathbb{P}(\mathbf{z} \setminus \mathbf{z}_{C_{\hat{n}}}|\mathbf{x}) \tag{29}$$

$$= \mathbb{P}(\mathbf{z}_{C_{\hat{n}}} = \mathbf{0}|z_{\hat{n}} = 0, \mathbf{x})\ \mathbb{P}(\mathbf{z} \setminus \mathbf{z}_{C_{\hat{n}}}|\mathbf{x}) \tag{30}$$

$$= \mathbb{P}(\mathbf{z}_{C_{\hat{n}}} = \mathbf{0}|z_{\hat{n}} = 0, \mathbf{x}) \prod_{n \in \mathcal{I}(\overline{\mathcal{T}}) : z_n = 1} \mathbb{P}(\mathbf{z}_{C_n}|z_n = 1, \mathbf{x}) \tag{31}$$

$$= 1 * \prod_{n \in \mathcal{I}(\overline{\mathcal{T}}) : z_n = 1} \mathbb{P}(\mathbf{z}_{C_n}|z_n = 1, \mathbf{x}) \tag{32}$$

$$= \prod_{n \in \mathcal{I}(\mathcal{T}) : z_n = 1} \mathbb{P}(\mathbf{z}_{C_n}|z_n = 1, \mathbf{x}) \tag{33}$$

If $z_{\hat{n}} = 1$ then by (11)

$$\mathbb{P}(\mathbf{z}(\mathcal{T})|\mathbf{x}) = \mathbb{P}(\mathbf{z}_{C_{\hat{n}}}|z_{\hat{n}} = 1, \mathbf{z} \setminus \{z_{\hat{n}}, \mathbf{z}_{C_{\hat{n}}}\}, \mathbf{x})\ \mathbb{P}(\mathbf{z} \setminus \mathbf{z}_{C_{\hat{n}}}|\mathbf{x}) \tag{34}$$

$$= \mathbb{P}(\mathbf{z}_{C_{\hat{n}}}|z_{\hat{n}} = 1, \mathbf{x})\ \mathbb{P}(\mathbf{z} \setminus \mathbf{z}_{C_{\hat{n}}}|\mathbf{x}) \tag{35}$$

$$= \mathbb{P}(\mathbf{z}_{C_{\hat{n}}}|z_{\hat{n}} = 1, \mathbf{x}) \prod_{n \in \mathcal{I}(\overline{\mathcal{T}}) : z_n = 1} \mathbb{P}(\mathbf{z}_{C_n}|z_n = 1, \mathbf{x}) \tag{36}$$

$$= \prod_{n \in \mathcal{I}(\overline{\mathcal{T}}) : z_n = 1} \mathbb{P}(\mathbf{z}_{C_n}|z_n = 1, \mathbf{x}) \tag{37}$$

As a result the hypothesis holds for any subtree $\mathcal{T}$ and consequently it holds for the given label tree as well. □

2.3 Prediction

Gain functions defined over the top ranked relevant predictions tend to be preferred for evaluating real-world ranking, recommendation and tagging applications as compared to traditional multi-label loss functions. Parabel's predictions therefore optimize such gain functions, including precision@r and nDCG@r, based on the following theorems.

THEOREM 2.4. *Let $\mathbb{P}(\mathbf{y}|\mathbf{x})$ represent the joint probability that a set of labels $\mathbf{y}$ is relevant to point $\mathbf{x}$. Then, the ranking of labels according to their marginal probabilities as rank $\left(\{\mathbb{P}(y_l = 1|\mathbf{x})\}_{l=1}^{L}\right)$ maximizes the expected gain of functions defined over the top ranked predictions alone such as precision@r and nDCG@r.*

PROOF. Following [2], precision@r and nDCG@r are defined as follows

$$\text{Precision@}r = \frac{1}{r} \sum_{l} y_l \hat{y}_l \tag{38}$$

$$\text{nDCG@}r = \frac{\sum_l \frac{y_l \hat{y}_l}{\log(b_l+1)}}{\left(\sum_{l=1}^{r} \frac{1}{\log(1+l)}\right)} \tag{39}$$

$$\tag{40}$$

where $\mathbf{y} \in \{0,1\}^L$ is the true label vector, $\hat{\mathbf{y}} \in \{0,1\}^L$ is the predicted label vector which has only r non-zero entries and b_l represents the rank of label l in $\hat{\mathbf{y}}$.

Both precision@r and nDCG@r can be expressed as

$$\text{Gain@}r = g(r) \sum_l f(b_l) y_l \hat{y}_l \tag{41}$$

$$\text{where } \; g(r) = \frac{1}{r}, f(b_l) = 1 \;\; \text{for precision@}r \tag{42}$$

$$g(r) = 1/\sum_{l=1}^{r} \frac{1}{\log(1+l)}, f(b_l) = \frac{1}{\log(b_l+1)} \;\; \text{for nDCG@}r \tag{43}$$

$$\text{and } f(x) \text{ is assumed to be a non-increasing function.} \tag{44}$$

Given $\mathbb{P}(\mathbf{y}|\mathbf{x})$, the ideal label predictions $\hat{\mathbf{y}}^*$ with respect to Gain@r are given by

$$\hat{\mathbf{y}}^* = \arg\max_{\hat{\mathbf{y}}} \sum_{\mathbf{y}} \mathbb{P}(\mathbf{y}|\mathbf{x}) \; g(r) \sum_l f(b_l) y_l \hat{y}_l \tag{45}$$

$$= \arg\max_{\hat{\mathbf{y}}} \sum_l \sum_{\mathbf{y}} \mathbb{P}(\mathbf{y}|\mathbf{x}) f(b_l) y_l \hat{y}_l \quad \text{–since } g(r) \text{ is constant} \tag{46}$$

$$= \arg\max_{\hat{\mathbf{y}}} \sum_{l:\hat{y}_l=1} f(b_l) \sum_{\mathbf{y}} y_l \mathbb{P}(\mathbf{y}|\mathbf{x}) \tag{47}$$

$$= \arg\max_{\hat{\mathbf{y}}} \sum_{l:\hat{y}_l=1} f(b_l) \sum_{\mathbf{y}:y_l=1} \mathbb{P}(\mathbf{y}|\mathbf{x}) \tag{48}$$

$$= \arg\max_{\hat{\mathbf{y}}} \sum_{l:\hat{y}_l=1} f(b_l) \mathbb{P}(y_l = 1|\mathbf{x}) \tag{49}$$

It is straightforward to see that the solution to the above maximization problem is obtained by ranking labels by decreasing $\mathbb{P}(y_l = 1|\mathbf{x})$ values and predicting the top r labels $\{l_1, \cdots, l_r\}$ where $b_{l_k} = k$. Therefore $\hat{y}_l = 1$ if $l \in \{l_1, \ldots, l_r\}$ and $\hat{y}_l = 0$ otherwise. □

Theorem 2.5. *Given a joint probability distribution $\mathbb{P}(\mathbf{y}|\mathbf{x})$ defined as in (8) over a label tree, the marginal probability of label l in leaf node n being relevant to point $\mathbf{x}$ is given by*

$$\mathbb{P}(y_l = 1|\mathbf{x}) = \mathbb{P}(y_l = 1|z_n = 1, \mathbf{x}) \prod_{\hat{n} \in \mathcal{A}_n} \mathbb{P}(z_{\hat{n}} = 1|z_{\mathcal{P}_{\hat{n}}} = 1, \mathbf{x}) \tag{50}$$

where $\mathcal{A}_n$ is the set of ancestors of node n apart from the root and $\mathcal{P}_{\hat{n}}$ is the parent of $\hat{n}$.

Proof. Let $\mathcal{N}_1 = \{n_1, \cdots, n_H\}$ be a path of length H from the root node n_1 to a leaf node n_H containing a label l. Let $\mathcal{N}_h = \{n_h, \cdots, n_H\}$ be a partial path from an internal node n_h to the leaf node n_H. The proof uses induction over the length of such partial paths.

Hypothesis: For a node $n_h \in \mathcal{N}_1$ such that the length of the path from n_h to leaf n_H containing label l is $H - h + 1$:

$$\mathbb{P}(y_l = 1|z_{n_h} = 1, \mathbf{x}) = \mathbb{P}(y_l = 1|z_{n_H} = 1, \mathbf{x}) \prod_{\hat{h}=h}^{H-1} \mathbb{P}(z_{n_{\hat{h}+1}} = 1|z_{n_{\hat{h}}} = 1, \mathbf{x}) \tag{51}$$

Base case: For path length of 1 *i.e.* $h = H$

$$\mathbb{P}(y_l = 1|z_{n_h} = 1, \mathbf{x}) = \mathbb{P}(y_l = 1|z_{n_H} = 1, \mathbf{x}) \tag{52}$$

which satisfies the hypothesis.

Induction: Let the hypothesis be satisfied for all path lengths $< H - h + 1$, then we will prove that the hypothesis holds for path length of $H - h + 1$. Since $y_l = 1$, by unvisited node assumption (9), $z_{n_h} = 1 \forall h \in \{1, \cdots, H\}$. Now

$$\mathbb{P}(y_l = 1|z_{n_h} = 1, \mathbf{x}) = \mathbb{P}(y_l = 1, z_{n_{h+1}} = 1|z_{n_h} = 1, \mathbf{x}) + \mathbb{P}(y_l = 1, z_{n_{h+1}} = 0|z_{n_h} = 1, \mathbf{x}) \tag{53}$$

$$= \mathbb{P}(y_l = 1, z_{n_{h+1}} = 1|z_{n_h} = 1, \mathbf{x}) \tag{54}$$

$$= \mathbb{P}(y_l = 1|z_{n_{h+1}} = 1, z_{n_h} = 1, \mathbf{x}) \; \mathbb{P}(z_{n_{h+1}} = 1|z_{n_h} = 1, \mathbf{x}) \tag{55}$$

$$= \mathbb{P}(y_l = 1|z_{n_{h+1}} = 1, \mathbf{x}) \; \mathbb{P}(z_{n_{h+1}} = 1|z_{n_h} = 1, \mathbf{x}) \quad \text{– since } z_{n_{h+1}} = 1 \implies z_{n_h} = 1 \tag{56}$$

$$= \mathbb{P}(y_l = 1 | z_{n_H} = 1, \mathbf{x}) \prod_{\hat{h}=h+1}^{H-1} \mathbb{P}(z_{n_{\hat{h}}+1} = 1 | z_{n_{\hat{h}}} = 1, \mathbf{x}) \, \mathbb{P}(z_{n_{h+1}} = 1 | z_{n_h} = 1, \mathbf{x}) \tag{57}$$

$$= \mathbb{P}(y_l = 1 | z_{n_H} = 1, \mathbf{x}) \prod_{\hat{h}=h}^{H-1} \mathbb{P}(z_{n_{\hat{h}}+1} = 1 | z_{n_{\hat{h}}} = 1, \mathbf{x}) \tag{58}$$

Thus the hypothesis holds true for path of length $H - h + 1$ and by induction it holds for all path lengths including length of H which proves the theorem. □

3 RESULTS

Table 1: Results comparing Parabel's performance to tree, embedding and 1-vs-All based baseline algorithms where accuracy is measured in terms of precision@r (Pr), nDCG@r (Nr), propensity-scored precision@r (PSPr) and propensity-scored nDCG@r (PSNr). Missing numbers will be updated soon.

Method	P1 (%)	P3 (%)	P5 (%)	N1 (%)	N3 (%)	N5 (%)	PSP1 (%)	PSP3 (%)	PSP5 (%)	PSN1 (%)	PSN3 (%)	PSN5 (%)
EURLex-4K												
PfastreXML	75.45	62.70	52.51	75.45	65.97	60.78	43.86	45.72	46.97	43.86	45.23	46.03
PLT	73.64	60.27	49.87	73.64	63.64	58.16	32.33	37.57	40.29	32.33	36.14	37.96
CS	58.52	45.51	32.47	58.52	48.67	40.79	24.97	27.46	25.04	24.97	26.82	25.57
CPLST	72.28	58.16	47.73	72.28	61.64	55.92	28.60	32.49	34.46	28.60	31.45	32.77
WSABIE	68.55	55.11	45.12	68.55	58.44	53.03	31.16	34.85	36.82	31.16	33.85	35.17
LEML	63.40	50.35	41.28	63.40	53.56	48.57	24.10	27.20	29.09	24.10	26.37	27.62
SLEEC	79.26	64.30	52.33	79.26	68.13	61.60	34.25	39.83	42.76	34.25	38.35	40.30
PD-Sparse	76.43	60.37	49.72	76.43	64.31	58.78	38.28	42.00	44.89	38.28	40.96	42.84
DiSMEC	82.40	68.50	57.70	82.40	72.50	66.70	41.20	45.40	49.30	41.20	44.30	46.90
Parabel-l-T=3	81.91	68.50	57.54	81.91	71.88	66.40	37.39	45.04	48.85	37.39	42.91	45.50
Parabel-s-T=3	82.25	68.71	57.53	82.25	72.17	66.54	36.44	44.08	48.46	36.44	41.99	44.91
Parabel-s-T=1	81.52	67.83	56.49	81.52	71.32	65.51	36.07	43.48	47.39	36.07	41.45	44.07
WikiLSHTC-325K												
PfastreXML	56.05	36.79	27.09	56.05	50.59	50.13	30.66	31.55	33.12	30.66	31.24	32.09
PLT	41.62	26.78	20.38	41.62	36.88	37.11	13.06	15.96	18.59	13.06	15.03	16.49
SLEEC	54.83	33.42	23.85	54.83	47.25	46.16	20.27	23.18	25.08	20.27	22.27	23.35
PD-Sparse	61.26	39.48	28.79	61.26	55.08	54.67	28.34	33.50	36.62	28.34	31.92	33.68
DiSMEC	64.40	42.50	31.50	64.40	58.50	58.40	29.10	35.60	39.50	29.10	35.90	39.40
Parabel-l-T=3	64.38	42.40	31.14	64.38	58.25	57.85	28.27	33.63	36.77	28.27	31.99	33.77
Parabel-s-T=3	**65.04**	**43.23**	**32.05**	65.04	59.15	58.93	26.90	33.47	37.46	26.90	31.44	33.70
Parabel-s-T=1	63.00	41.35	30.36	63.00	56.78	56.24	26.03	31.67	34.93	26.03	29.93	31.79
Amazon-3M												
PfastreXML	43.83	41.81	40.09	43.83	42.68	41.75	21.38	23.22	24.52	21.38	22.75	23.68
Parabel-l-T=3	42.54	40.14	38.22	42.54	41.10	40.03	13.89	16.43	18.24	13.89	15.77	17.05
Parabel-s-T=3	47.51	44.68	42.58	47.51	45.77	44.58	12.84	15.63	17.75	12.84	14.91	16.40
Parabel-s-T=1	46.14	43.35	41.23	46.14	44.41	43.18	12.50	15.20	17.21	12.50	14.50	15.92
Amazon-670K												
PfastreXML	39.46	35.81	33.05	39.46	37.78	36.69	29.30	30.80	32.43	29.30	30.40	31.49
PLT	36.86	32.48	29.15	36.86	34.39	32.74	21.86	24.26	26.27	21.86	23.64	25.01
SLEEC	35.05	31.25	28.56	34.77	32.74	31.53	20.62	23.32	25.98	20.62	22.63	24.43
DiSMEC	**44.70**	**39.70**	**36.10**	44.70	42.10	40.50	27.80	30.60	34.20	27.80	28.80	30.70
Parabel-l-T=3	43.90	39.42	36.09	43.90	41.65	40.25	27.34	30.85	34.03	27.34	29.93	32.10
Parabel-s-T=3	44.90	39.81	35.99	44.90	42.16	40.37	26.32	29.99	33.17	26.32	29.04	31.21
Parabel-s-T=1	43.29	38.03	34.07	43.29	40.36	38.39	25.43	28.60	31.27	25.43	27.77	29.61

Continued on next page

Table 2: Results of Parabel and baseline algorithms on benchmark datasets where data points were represented by dense deep XML-CNN [3] embeddings. Parabel is significantly more accurate than tree and embedding based baselines. Parabel is also $2x - 500x$ faster at training and $150x$ faster at prediction as compared to 1-vs-All classifiers while being up to 4% worse in terms of precisions.

Method	P1 (%)	P3 (%)	P5 (%)	Training time (hr)	Test time / point (ms)
EURLex-D-4K					
PfastreXML	73.63	60.31	49.69	0.037	1.82
SLEEC	74.31	60.00	49.11	0.35	4.87
LEML	60.34	47.45	37.96	0.67	2.24
WSABIE	76.09	61.69	49.11	0.13	2.24
DiSMEC	76.12	62.91	51.51	0.13	4.36
PD-Sparse	73.53	60.80	49.37	0.12	4.36
PPDSparse	**76.32**	**62.79**	**51.40**	0.013	4.36
Parabel-l-T=3	74.54	61.72	50.48	**0.01**	**0.91**
Amazon-D-670K					
PfastreXML	28.51	26.06	24.17	2.85	19.35
SLEEC	18.77	16.50	14.97	7.12	22.54
DiSMEC	**37.60**	**33.62**	**30.64**	788.84	429
PPDSparse	33.16	29.60	26.85	3.90	429
Parabel-l-T=3	33.93	30.38	27.49	**1.54**	**2.85**
Wikipedia-D-500K					
PfastreXML	55.00	36.14	27.38	11.14	6.36
DiSMEC	**63.70**	**42.49**	**32.26**	2133	316.29
PPDSparse	50.40	33.15	25.54	**5.85**	316.29
Parabel-l-T=3	59.34	39.05	29.35	6.29	**2.94**

Table 1 – continued from previous page

Method	P1 (%)	P3 (%)	P5 (%)	N1 (%)	N3 (%)	N5 (%)	PSP1 (%)	PSP3 (%)	PSP5 (%)	PSN1 (%)	PSN3 (%)	PSN5 (%)
DSA-2M												
PfastreXML	28.52	17.05	12.5	28.52	29.12	30.11	27.37	33.94	36.07	27.37	27.56	28.47
Parabel-l-T=3	32.07	18.64	13.52	32.06	37.92	40.26	29.66	37.67	41.14	29.66	34.86	36.69
Parabel-s-T=3	**33.44**	**20.21**	**14.79**	33.44	40.25	42.86	29.93	39.86	44.14	29.93	36.37	38.63
Parabel-s-T=1	31.26	18.83	13.74	31.26	37.45	39.79	28.01	37.10	40.90	28.01	33.91	35.92
DSA-7M												
PfastreXML	28.09	25.79	23.21	28.09	28.81	29.86	26.36	26.64	27.66	26.36	26.55	27.21
Parabel-l-T=3	31.95	29.42	26.40	31.95	32.59	33.52	28.91	29.36	30.49	28.91	29.22	29.97
Parabel-s-T=3	**32.84**	**30.28**	**27.35**	32.84	33.49	34.48	28.66	29.11	30.43	28.66	28.97	29.83
Parabel-s-T=1	30.77	28.35	25.61	30.77	31.37	32.28	26.89	27.29	28.51	26.89	27.17	27.97

Table 3: Parabel variants on EURLex-4K

Method	P1 (%)	P3 (%)	P5 (%)	Training time (hr)	Test time/point (ms)	Model size (GB)
Parabel-l-T=3	81.91	68.50	57.54	0.063	1.01	0.038
Parabel-s-T=3	82.25	68.71	57.53	0.018	0.88	0.026
Parabel-s-T=1	81.52	67.83	56.49	**0.005**	0.28	**0.0086**

Table 4: Variation in Parabel's performance with the number of label trees, *i.e.* hyperparameter T, on WikiLSHTC dataset. Parabel's accuracy increases by 2% with an ensemble of 3 trees and witnesses diminishing returns with more trees.

Trees T	P1 (%)	P3 (%)	P5 (%)	Train time (Hr)	Test time /point (ms)	Model size (GB)
1	62.68	41.25	30.40	0.26	0.74	1.06
3	64.57	43.00	31.95	0.79	1.61	3.17
5	65.03	43.43	32.33	1.34	2.65	5.28
10	65.47	43.83	32.68	2.67	13.05	10.56
20	65.67	44.00	32.84	5.34	11.36	21.13
30	65.74	44.07	32.90	8.01	18.44	31.69
40	65.78	44.11	32.93	10.68	25.48	42.25

Table 5: Variation in Parabel's performance with the number of maximum labels in the leaf nodes, *i.e.* hyperparameter M, on WikiLSHTC dataset. Both accuracy and test time increase with larger M, with Parabel achieving around $1ms$ test time per point and minimal loss in accuracies at $M = 100$. Results are reported for Parabel-s-T=3 with 3 trees.

Max. labels M	P1 (%)	P3 (%)	P5 (%)	Train time (Hr)	Test time /point (ms)	Model size (GB)
400	65.01	43.30	32.14	1.68	5.43	2.72
200	64.81	43.19	32.08	1.07	2.82	2.92
100	64.57	43.00	31.95	0.78	1.58	3.17
50	64.21	42.72	31.76	0.65	0.96	3.49
25	63.80	42.35	31.49	0.62	0.69	3.90
12	63.90	41.94	31.20	0.64	0.55	4.36

Table 6: Variation in Parabel's performance with the beam search width, *i.e.* hyperparameter P, on WikiLSHTC dataset. Higher P values indicate more thorough tree search. Parabel accuracy initially increases with P and quickly saturates at around $P = 10$. Results are reported for Parabel-s-T=3 with 3 trees.

Beam Width P	P1 (%)	P3 (%)	P5 (%)	Test time / point (ms)
1	60.48	34.53	23.30	0.18
3	64.49	42.33	30.84	0.51
5	64.62	42.90	31.68	0.82
10	64.57	43.00	31.95	1.58
20	64.54	42.97	31.95	3.28
40	64.52	42.95	31.94	8.59

REFERENCES

[1] A. Bertoni, M. Goldwurm, J. Lin, and F. Saccà. 2012. Size Constrained Distance Clustering: Separation Properties and Some Complexity Results. 115 (2012), 125–139.

[2] H. Jain, Y. Prabhu, and M. Varma. 2016. Extreme Multi-label Loss Functions for Recommendation, Tagging, Ranking & Other Missing Label Applications. In *KDD*.

[3] J. Liu, W. Chang, Y. Wu, and Y. Yang. 2017. Deep Learning for Extreme Multi-label Text Classification. In *SIGIR*. 115–124.