Adaptive Softmax Trees for Many-Class Classification

Rasul Kairgeldin, **Magzhan Gabidolla** and **Miguel Á. Carreira-Perpiñán**
Dept. Computer Science & Engineering, UC Merced

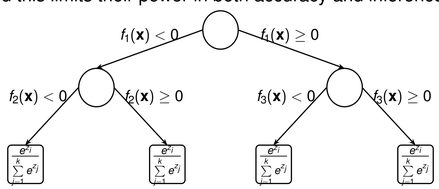
1 Abstract

NLP tasks such as language models or document classification involve classification problems with thousands of classes. In these situations, it is difficult to get high predictive accuracy and the resulting model can be huge in number of parameters and inference time. A recent, successful approach is the softmax tree (ST): a decision tree having sparse hyperplane splits at the decision nodes (which make hard, not soft, decisions) and small softmax classifiers at the leaves. Inference here is very fast because only a small subset of class probabilities need to be computed, yet the model is quite accurate. However, a significant drawback is that it assumes a complete tree, whose size grows exponentially with depth. We propose a new algorithm to train a ST of arbitrary structure. The tree structure itself is learned optimally by interleaving steps that grow the structure with steps that optimize the parameters of the current structure. This makes it possible to learn STs that can grow much deeper but in an irregular way, adapting to the data distribution. The resulting STs improve considerably the predictive accuracy while reducing the model size and inference time even further, as demonstrated in datasets with thousands of classes. In addition, they are interpretable to some extent.

Work supported by NSF award IIS–2007147

2 Softmax Tree (ST)

- Each decision node $i \in \mathcal{N}_{\text{dec}}$ has a decision function $g_i(\mathbf{x};\theta_i)$: "if $\mathbf{w}_i^T\mathbf{x} + w_{i0} \geq 0$ then $g_i(\mathbf{x}) = \texttt{right}_i$, otherwise $g_i(\mathbf{x}) = \texttt{left}_i$"
- Each leaf $j \in \mathcal{N}_{\text{leaf}}$ contains a softmax function $\mathbf{f}_j(\mathbf{x};\theta_j) = \sigma(\mathbf{W}_j\mathbf{x} + \mathbf{w}_{j0})$ that predicts a set of $k \leq K$ classes.
- Training is done using Tree Alternating Optimization (**TAO**): general method for optimizing a given objective function over a given decision tree model
- Assumes a complete tree structure, whose size grows exponentially with depth, and this limits their power in both accuracy and inference time



Given a Softmax Tree $\tau(\mathbf{x};\Theta)$ of fixed structure (e.g. a complete tree of depth Δ) and initial parameters (e.g. random), the goal of TAO is to minimize the following objective:

$$E(\Theta) = \sum_{n=1}^{N} L(\mathbf{y}_n, \tau(\mathbf{x}_n)) + \lambda \sum_{i \in \mathcal{N}_{\text{dec}}} \|\mathbf{w}_i\|_1 + \mu \sum_{j \in \mathcal{N}_{\text{leaf}}} \|\mathbf{W}_j\|_1$$

where $L(\cdot,\cdot)$ is the cross-entropy loss, $\Theta = \{\mathbf{w}_i, w_{i0}\}_{i \in \mathcal{N}_{\text{dec}}} \cup \{\mathbf{W}_j, \mathbf{w}_{j0}\}_{j \in \mathcal{N}_{\text{leaf}}}$ are the set of all learnable model parameters, and there is an ℓ_1 penalty over the weight vectors to promote sparsity via hyperparameters $\lambda, \mu \geq 0$.

3 Adaptive Softmax Tree (AST)

- Improve accuracy and inference of ST by exploring deeper structures
- The inference time for a leaf j is $\mathcal{O}(D(\Delta_j + k_j))$. The improvement is in much smaller values of k_j at the expense of slightly large values of Δ_j (thin softmaxes in deep leaves).
- The training of AST consists of two steps: *regular step* and *expansion step*

Algorithm starts with a small ST (e.g. $\Delta = 2$) and large leaf softmaxes k_0.

Regular step include optimizing ST of current structure $\tau(\cdot;\Theta)$ using TAO:

- For a decision node $i \in \mathcal{N}_{\text{dec}}$ reduced problem is a *weighted 0/1 loss binary classification problem*:

$$E_i(\mathbf{w}_i, w_{i0}) = \sum_{n \in \mathcal{R}_i} c_n \overline{L}(\overline{y}_n, g_i(\mathbf{x}_n)) + \lambda \|\mathbf{w}_i\|_1$$

where $\overline{L}(\cdot,\cdot)$ is the 0/1 loss, $\overline{y}_n \in \{\texttt{left}_i, \texttt{right}_i\}$ is a pseudolabel indicating the "best" child and $c_n \geq 0$ is the loss difference between the "other" child and the "best" one for the instance $\mathbf{x}_n$.

- For leaf node $j \in \mathcal{N}_{\text{leaf}}$:

$$E_j(\mathbf{W}_j, \mathbf{w}_{j0}) = \sum_{n \in \mathcal{R}_j} L(\mathbf{y}_n, \mathbf{f}_j(\mathbf{x}_n)) + \mu \|\mathbf{W}_j\|_1$$

where $L(\cdot,\cdot)$ is the original cross-entropy loss

Expansion step on the current leaf replaces is with shallow ST with narrower softmaxes:

- *Softmax contraction coefficient* α controls shrinkage of leaf softmaxes after each expansion
- *Tolerance ratio* ρ controls performance of the expanded subtree
- The expansion move allows us to compare the objective function before and after the expansion in order to decide whether or not we should pursue a new architecture

input training set $\{\mathbf{x}_n, y_n\}_{n=1}^N$, initial depth Δ_0,
  softmax contraction coefficient $\alpha \in (0,1)$,
  tolerance ratio for node expansion $\rho > 1$.
$k_0 \leftarrow \alpha K$
initialize $\tau(\cdot;\Theta)$ of depth Δ_0 and k_0-class leaves;
fit $\tau(\cdot;\Theta)$ using TAO;
repeat
  update reduced sets $\mathcal{R}_j$ for all $j \in \mathcal{N}_{\text{leaf}}$;
  for $j \in \mathcal{N}_{\text{leaf}}$
    initialize ST $\hat{\tau}_j(\cdot;\hat{\Theta}_j)$ of depth $\Delta = 1$ or 2
    and with (αk_j)-class softmax leaves;
    fit $\hat{\tau}_j(\cdot;\hat{\Theta}_j)$ using TAO on $\{\mathbf{x}_n, y_n\}_{n \in \mathcal{R}_j}$;
    if $\frac{\texttt{loss}(\hat{\tau}_j(\cdot;\hat{\Theta}_j))}{\texttt{loss}(\mathbf{f}_j(\cdot;\theta_j))} < \rho$ **then** accept the expansion of leaf j
  end for
  update the tree $\tau(\cdot;\Theta)$ and reoptimize with TAO;
until no changes to the tree structure
return adaptively grown $\tau(\cdot;\Theta)$

4 Experiments Results

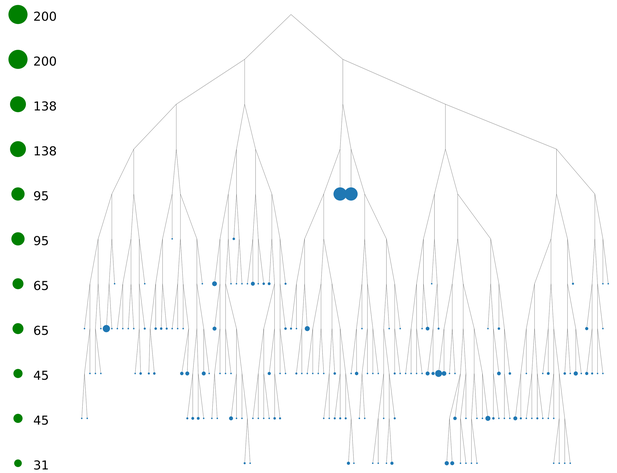


Figure 1: AST for the Wiki-Small subs. dataset. Size of the blue nodes (on the tree) shows the actual number of classes in the leaves after pruning. Green (left column) shows theoretical max. values at each aligned depth.

	Method	E_{test}%	Δ	L	$\bar{k}$	inf.(µs)	FLOPs
ALOI	Softmax	13.0	–	–	–	411	128000
	ST*($k=90$)	12.3	7	126	64.9	24	1493
	ST($k=75$)	12.0	6	64	74.9	29	1871
	ST(from AST)	12.8	8	177	38.4	18	1102
	AST*(α=0.75,ρ=1.01)	**9.9**	10	326	23.8	**15**	**1016**
LSHTC1	Softmax	61.4	–	–	–	10680	423722
	ST($k=70$)	62.7	7	128	70.0	65	12279
	ST($k=50$)	61.2	8	256	49.4	55	9218
	ST(from AST)	68.7	9	511	49.7	62	9388
	AST*(α=0.9,ρ=1.2)	**60.8**	10	1006	11.5	**40**	**3756**
WIKI-Small subs.	Softmax	50.2	–	–	–	16500	9214
	ST*($k=4$)	51.5	8	30	4.6	36	691
	AST*(α=0.35,ρ=1.2)	**49.5**	11	73	4.1	**16**	**586**
	ST*($k=9$)	48.3	8	50	8.0	27	918
	AST($\alpha=0.38, \mu=0.1$)	**46.9**	11	**13**	44	8.4	**791**
	ST($k=13, \mu=0.1$)	48.3	8	13	40	12.1	1104
	AST*(α=0.39,ρ=1.2)	**47.5**	11	34	11.7	**12**	**929**
	ST($k=67, \mu=0.01$)	48.4	8	21	256	8.11	2291
	ST($k=95$)	44.1	8	256	5.7	30	3065
	ST(from AST)	44.0	8	65	12.5	19	3296
	AST(α=0.69,ρ=1.2)	**42.7**	13	184	2.8	**13**	**1437**

Table 1: AST vs ST. We report: test errors; depth Δ, number of leaves L, average leaf softmax size $\bar{k}$ of the tree; and average inference time and FLOPs per test instance. For ST we specify its leaf softmax size k, for AST the softmax contraction coefficient α and tolerance ratio of expansion ρ. ASTs are trained with $\mu = 0.01$ or (if marked with $*$) $\mu = 0.1$.

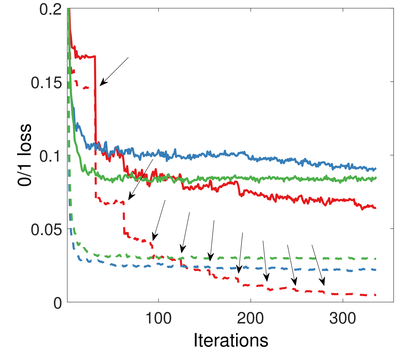


Figure 2: 0/1 loss of the final AST model for training (dashed line) and test (solid line), compared with the complete Softmax Tree. The arrows point to where expansions of the AST happened. The line colors indicate the performance of the ST (blue), ST(AST) (green) and AST (red). This shows that the adaptive growth gradually enhances the performance of the model on both training and test tests (red solid and dashed lines). On the other hand, a ST initialized randomly (blue line) or on the final structure of AST (green line) is unable to improve after a certain number of iterations.

Method	E_{test}(%)	Δ	inf.(µs)	PPL	(%nnz)
HSM	91.1	18	421	575	(100%)
one-vs-all	87.5	0	705	220	(100%)
ST(k = 50)	86.5	8	58	17	(44%)
ST(k = 100)	86.5	7	58	27	(51%)
ST(k = 400)	86.4	5	64	71	(67%)
AST($\alpha = 0.3$)	86.4	12	**17**	10	(37%)
AST($\alpha = 0.4$)	**86.1**	12	18	13	(44%)
AST($\alpha = 0.5$)	86.2	11	19	24	(51%)
AST($\alpha = 0.75$)	86.3	12	20	7	(33%)

Table 2: Results on the language modeling dataset PTB. We report test error, depth Δ of the tree, the average inference time per sample in microseconds and the average perplexity (PPL) over the test set instances for which the model outputs nonzero probability. The percentage of such instances is shown in parenthesis. For AST models $\rho = 1.0$.