Adaptive Softmax Trees for Many-Class Classification

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

1 Introduction

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 partially supported by NSF award IIS–2007147.

2 Adaptive Softmax Tree (AST)

Algorithm starts with a small ST (e.g. $\Delta = 2$) and large leaf softmaxes k_0. It learns both the parameters and the structure of a softmax tree:

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

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

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

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

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

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

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

- Allows one to compare the objective function before and after the expansion in order pursue a new architecture.
- *Softmax contraction coefficient* α controls shrinkage of leaf softmaxes.
- *Tolerance ratio* ρ controls performance of the expanded subtree.

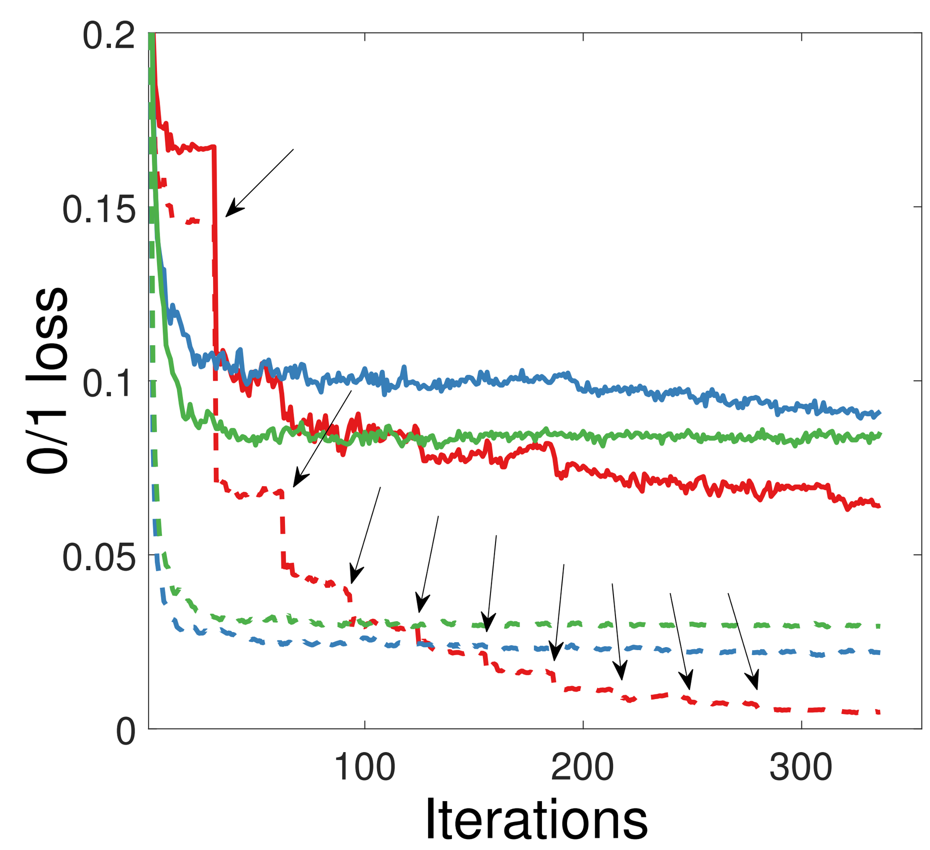
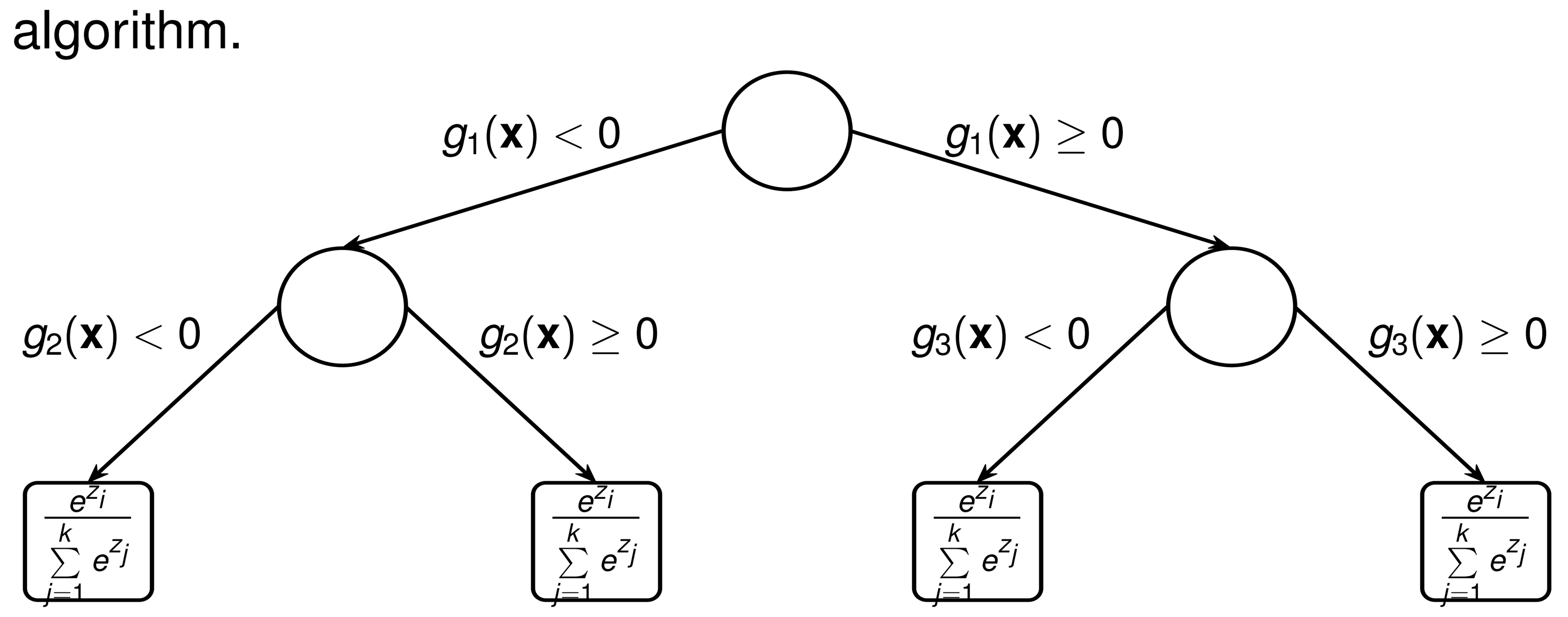


Figure: Training (dashed line) and test (solid line) error of ST (blue), ST initialized with AST architecture (green) and AST (red). The arrow points to expansion during AST training. This shows that the adaptive growth gradually enhances the performance of the model on both training and test tests (red 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.

3 Softmax Tree (ST)

Proposed by Zharmagambetov et al., EMNLP 2021.

- Each decision node $i \in \mathcal{N}_{\text{dec}}$ has a decision function $g_i(\mathbf{x};\boldsymbol{\theta}_i)$:
  "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 $\mathbf{f}_j(\mathbf{x};\boldsymbol{\theta}_j) = \sigma(\mathbf{W}_j\mathbf{x} + \mathbf{w}_{j0})$ that predicts a set of $k \leq K$ classes.
- Much faster inference compared to linear softmax model.
- Size grows exponentially with depth, and this limits their power in both accuracy and inference time.
- It can be trained with a variation of the Tree Alternating Optimization (TAO) algorithm.



4 Results

	Method	E_{test}%	Δ	L	$\bar{k}$	inf.(μs)	FLOPs
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: 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 $*$) $\mu = 0.1$.

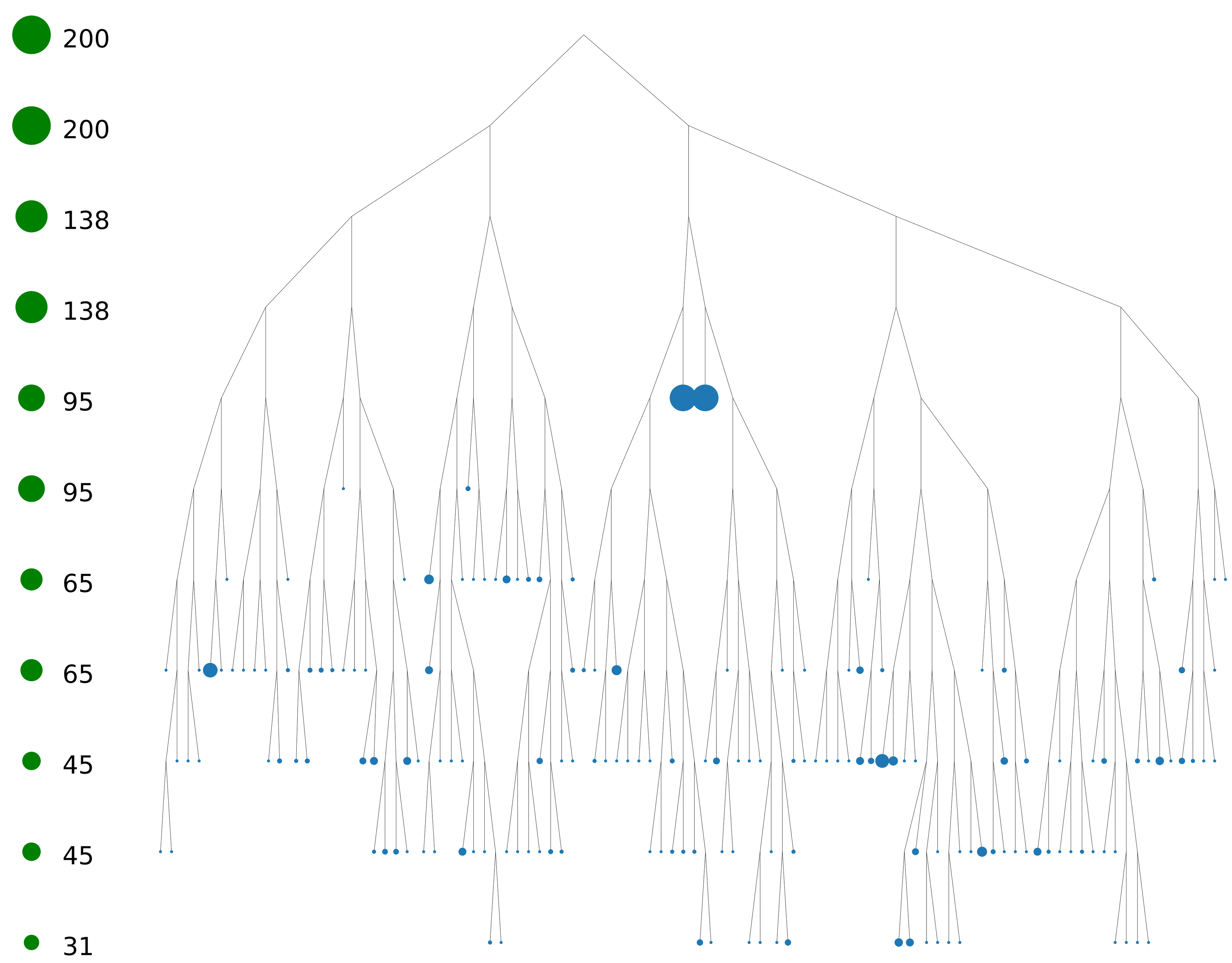


Figure: AST for the Wiki-Small subs. dataset. Size of the blue nodes (on the tree) shows the actual number of classes in the leaves after pruning. Green (left column) shows theoretical max. values at each aligned depth.