Learning with Deep Cascades

Giulia DeSalvo[1], Mehryar Mohri[1,2], and Umar Syed[2]

[1] Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, NY 10012
[2] Google Research, 111 8th Avenue, New York, NY 10011

Abstract. We introduce a broad learning model formed by cascades of predictors, *Deep Cascades*, that is structured as general decision trees in which leaf predictors or node questions may be members of rich function families. We present new data-dependent theoretical guarantees for learning with Deep Cascades with complex leaf predictors and node questions in terms of the Rademacher complexities of the sub-families composing these sets of predictors and the fraction of sample points reaching each leaf that are correctly classified. These guarantees can guide the design of a variety of different algorithms for deep cascade models and we give a detailed description of two such algorithms. Our second algorithm uses as node and leaf classifiers SVM predictors and we report the results of experiments comparing its performance with that of SVM combined with polynomial kernels.

Keywords: decision trees, learning theory, supervised learning.

1 Introduction

Decision trees are learning models commonly used in classification, regression, and clustering applications [6, 23]. They can be defined as binary trees augmented with indicator functions at each internal node and assignment functions at each leaf. A sample point is processed by a decision tree by answering questions at each node of a tree until a leaf is reached. The label assignment at that leaf then determines the value returned by the tree for that point.

In standard decision trees, the node questions are selected from a fixed family of functions and similarly for the leaf predictors [6, 23]. The complexity of a decision tree directly depends on these two families of functions and the depth of the tree. Thus, in practice, to limit the risk of overfitting, relatively simple families of functions are used: node questions are typically selected to be threshold functions based on the input features, leaf predictors often chosen to be constant functions.

This paper considers a significantly broader learning model formed by cascades of predictors, *Deep Cascades*, structured as a decision tree. In this model, the leaf predictors can be chosen out of a complex hypothesis set H and, similarly, the node questions from a family Q. For some difficult learning tasks, the flexibility of allowing leaf predictors to be selected from a more complex set H (or node questions from Q) may be needed to achieve a high performance. However, cascades with leaf predictors freely selected from H are likely to be prone to overfitting, even with a relatively large number of training samples. Can we preserve the flexibility of using complex leaf predictors (or node questions) and yet avoid overfitting?

Suppose H can be decomposed as the union of p distinct hypothesis sets $H_1, \ldots, H_p$ with increasing complexity. For example, H_k could be the family of threshold functions based on feature monomials of degree k, or polynomial functions of degree k, or H_k could be the family of linear classifiers based on polynomial kernels of degree k. The simpler form of our theoretical analysis shows that, remarkably, it is possible to choose a leaf predictor function from H_k with a relatively large k while benefitting from strong learning guarantees, so long as the fraction of training sample points reaching that leaf is small compared to the complexity of H_k. Our full analysis provides finer guarantees that we will describe in detail.

We present data-dependent theoretical guarantees for learning with Deep Cascades with leaf predictors chosen from the hypothesis sets H_k and node question functions selected from different hypothesis sets Q_j. Our learning bounds are expressed in terms of the Rademacher complexities of the families of leaf predictors H_k and the families of node questions Q_j. These general guarantees can guide the design of a variety of different algorithms for deep cascade models. We describe in depth two such algorithms for learning deep cascades. Our second algorithm uses as node and leaf classifiers SVM predictors and we report the results of experiments comparing its performance with that of SVM combined with polynomial kernels.

Our theory and algorithm have many connections with the wide literature on decision trees and some more recent publications on cascades of classifiers. They are also related to classification with reject option and to a series of articles about combining decision trees with the SVM algorithm. We briefly discuss some of these connections and highlight our contributions here. A more detailed discussion of the previous work is presented in the full version of the paper [11].

Several types of generalization bounds have been given in the past for decision trees. Mansour and McAllester [19] provided non-trivial generalization bounds for decision trees where the node questions are selected from a single hypothesis set and where the leaves are simply labeled with zero or one. These are special cases of the deep cascades we are considering. As in our analysis, their bounds depend on the actual training sample and the tree structure, but the complexity term of their bound is the size of the tree, while ours are expressed in terms of the empirical Rademacher complexities of the hypothesis sets used. A similar approach was adopted by Nobel [21] who further proved the consistency of pruned trees under certain assumptions. Golea et al. [13] gave generalization bounds in terms of the VC-dimension of the node functions and the number of leaves but the trees analyzed are much less general than the deep cascades. Lastly, Scott and Nowak [26] presented an analysis of a specific family of decision trees, Dyadic Decision Trees (DDT).

Cascades have been extensively used in object detection starting with the work of Viola and Jones [29] who introduced attentional cascades and combined complex classifiers in a linear structure to create a highly accurate face detector. Their work inspired a number of variants of their training procedures [8, 16, 22, 25, 27]. Most of these papers focus on finding the best trade-off between computational cost and classification accuracy, which differs from our main objective here. Additionally, the deep cascades we consider admit a more general structure than those considered by this previous work.

Since one of our deep cascade algorithms uses SVMs, we also review the related previous work on combining SVMs with decision trees. Bennett and Blue [5] used SVMs as node questions in decision trees. They did not present a theoretical analysis of these models and did not address the issue of overfitting, but they proposed an optimization problem for which they gave a heuristic solution and presented preliminary empirical results. Some of the papers in this area focus on multi-class classification [12, 18, 28]. However, they partition the feature space in a different way from our cascade models. Other articles attempted to increase SVM's computational speeds by using decision trees [1, 2, 7, 15, 24], but both the splitting criteria and class assignments are very different from ours.

The layout of the paper is as follows. We introduce the notation adopted throughout the paper and give a formal definition of the family of deep cascades in Section 2. Next, in Section 3, we present data-dependent learning bounds for deep cascades, first in the case of leaf functions taking values in $\{-1, +1\}$, and later in the more general case where they take values in the interval $[-1, +1]$. In Section 4, we describe two binary classification algorithms whose design is guided by these bounds and which benefit from these learning guarantees. We report the results of several experiments using one of these algorithms in Section 5.

2 Preliminaries

Let $\mathcal{X}$ denote the input space. We consider the standard supervised learning scenario where the training and test points are drawn i.i.d. according to some distribution $\mathcal{D}$ over $\mathcal{X} \times \{-1, +1\}$ and denote by $S = ((x_1, y_1), \ldots, (x_m, y_m))$ a training sample of size m drawn according to $\mathcal{D}^m$.

Let $l \geq 1$. For any $k \in [1, l]$, let $\mathcal{S}_k$ denote a family of functions mapping $\mathcal{X}$ to $\{0, 1\}$ and let $\mathcal{H}$ denote a family of p hypothesis sets of functions mapping $\mathcal{X}$ to $[0, 1]$. A *deep cascade* with $l \geq 1$ leaves is a tree of classifiers which, in the most generic view, can be defined by a triplet $(\mathbf{H}, \mathsf{s}, \mathbf{h})$ where

- $\mathbf{H} = (H_1, \ldots, H_l)$ is an element of $\mathbf{H}^l$ which determines, for all k, the hypothesis set H_k used at leaf k;
- $\mathsf{s}\colon \mathcal{X} \times [1, l] \to \{0, 1\}$ is a *leaf selector*, that is $\mathsf{s}(x, k) = 1$ if x is assigned to leaf k, $\mathsf{s}(x, k) = 0$ otherwise; for each k, function $\mathsf{s}(\cdot, k)$ is an element of $\mathcal{S}_k$;
- $\mathbf{h} = (h_1, \ldots, h_l)$, with $h_k\colon \mathcal{X} \to [-1, +1]$ the *leaf classifier* for leaf k, which is an element of the family of functions H_k.

We denote by $\mathcal{H}_k = \{x \mapsto \mathsf{s}(x, k) h_k(x)\colon \mathsf{s}(\cdot, k) \in \mathcal{S}_k, h_k \in H_k\}$ the family composed of products of a k-leaf selector and a k-leaf classifier.

We will later assume, as in standard decision trees, that the leaf selector s can be decomposed into *node questions* (or their complements): for any $x \in \mathcal{X}$ and $k \in [1, p]$, $\mathsf{s}(x, k) = \prod_{j=1}^{d_k} q_j(x)$, where d_k is the depth of node k and where each function $q_j\colon \mathcal{X} \to \{0, 1\}$ is an element of a family Q_j.[3] Yet much of our analysis holds without this assumption.

[3] Each q_j is either a node question q or its complement $\bar{q}$ defined by $\bar{q}(x) = 1$ iff $q(x) = 0$. The family Q_j is assumed symmetric: it contains $\bar{q}$ when it contains q.

Each triplet $(\mathbf{H}, \mathsf{s}, \mathbf{h})$ defines a *deep cascade function* $f\colon \mathcal{X} \to [-1, +1]$ as follows:

$$\forall x \in \mathcal{X},\ f(x) = \sum_{k=1}^{l} \mathsf{s}(x, k) h_k(x). \tag{1}$$

We denote by $\mathcal{T}_l$ the family of all deep cascade functions f with l leaves thereby defined. We also denote by $R(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[1_{yf(x)\leq 0}]$ the binary classification error of a function $f \in \mathcal{T}_l$, by $\widehat{R}_S(f) = \mathbb{E}_{(x,y)\sim S}[1_{yf(x)\leq 0}]$ its empirical error and, for any $\rho > 0$, by $\widehat{R}_{S,\rho}(f) = \mathbb{E}_{(x,y)\sim S}[1_{yf(x)\leq \rho}]$ its empirical margin error over a labeled sample S, where the notation $(x, y)\sim S$ means that (x, y) is drawn according to the empirical distribution defined by S. We further denote by $\mathfrak{R}_m(H)$ the Rademacher complexity and by $\widehat{\mathfrak{R}}_S(H)$ the empirical Rademacher complexity of a hypothesis set H [3, 14].

3 Data-dependent Learning Guarantees

In this section, we present data-dependent learning guarantees for deep cascades that depend, for each leaf k, on the Rademacher complexity of the family $\mathcal{H}_k$ and on the fraction of the points in the training sample that reach leaf k and that are correctly classified, denoted by m_k^+/m. m_k^+ is thus defined by $m_k^+ = |\{i\colon y_i h_k(x_i) > 0, \mathsf{s}(x_i, k) = 1\}|$. Similarly, the number of points that reach leaf k that are incorrectly classified is denoted by m_k^- and defined by $m_k^- = |\{i\colon y_i h_k(x_i) \leq 0, \mathsf{s}(x_i, k) = 1\}|$.

We first analyse the case where the leaf classifiers h_k take values in $\{-1, +1\}$ (Section 3.1), and next consider the more general case where they take values in $[-1, +1]$ (Section 3.2). In the full version of this paper [11], we further extend our analysis and data-dependent learning guarantees to the setting of multi-class classification.

3.1 Leaf classifiers taking values in $\{-1, +1\}$

The main result of this section is Theorem 1, which provides a data-dependent generalization bound for deep cascade functions in the case where leaf classifiers take values in $\{-1, +1\}$. The following is a simpler form of that result: with high probability, for all $f \in \mathcal{T}_l$,

$$R(f) \leq \widehat{R}_S(f) + \sum_{k=1}^{l} \min\left(4\,\widehat{\mathfrak{R}}_S(\mathcal{H}_k), \frac{m_k^+}{m}\right) + \widetilde{O}\left(l\sqrt{\frac{\log pl}{m}}\right). \tag{2}$$

Remarkably, this suggests that a strong learning guarantee holds even when a very complex hypothesis set $\mathcal{H}_k$ is used in a deep cascade model, so long as m_k^+/m, the fraction of the points in the training sample that reach leaf k and are correctly classified, is relatively small. Observe that the result remains remarkable and non-trivial even if we upper bound m_k^+ by m_k, the total number of points reaching leaf k. The fraction of the points in the training sample that reach leaf k and are correctly classified depends on the choice of the cascade. Thus, the bound can provide a quantitative guide for the choice of the best deep cascade. Even for $p = 1$, the result is striking since, while in

the worst case the complexity term could be in $O(l\widehat{\mathfrak{R}}_S(\mathcal{H}_1))$, this data-dependent result suggests that it can be substantially less for some deep cascades since we may have $m_k^+/m \ll \widehat{\mathfrak{R}}_S(\mathcal{H}_1)$ for many leaves. Also, note that the dependency of the bound on the number of distinct hypothesis sets p is only logarithmic. In Section 4, we present several algorithms exploiting this generalization bound for deep cascades.

For clarity, we will sometimes use the shorthand $r_k = \widehat{\mathfrak{R}}_S(\mathcal{H}_k)$ for any $k \in [1, l]$. We will assume without loss of generality that the leaves are numbered in order of increasing depth and will denote by $\mathcal{K}$ the set of leaves k whose fraction of correctly classified sample points is greater than $4r_k$: $\mathcal{K} = \{k \in [1, l] \colon \frac{m_k^+}{m} > 4r_k\}$.

Theorem 1. *Fix $\rho > 0$. Assume that for all $k \in [1, l]$, the functions in H_k take values in $\{-1, +1\}$. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the choice of a sample S of size $m \geq 1$, the following holds for all $l \geq 1$ and all cascade functions $f \in \mathcal{T}_l$ defined by $(\mathbf{H}, \mathsf{s}, \mathbf{h})$:*

$$R(f) \leq \widehat{R}_S(f) + \sum_{k=1}^{l} \min\left(4\widehat{\mathfrak{R}}_S(\mathcal{H}_k), \frac{m_k^+}{m}\right) + \min_{\substack{\mathcal{L} \subseteq \mathcal{K} \\ |\mathcal{L}| \geq |\mathcal{K}| - \frac{1}{\rho}}} \sum_{k \in \mathcal{L}} \left[\frac{m_k^+}{m} - 4\widehat{\mathfrak{R}}_S(\mathcal{H}_k)\right] + C(m, p, \rho) + \sqrt{\frac{\log \frac{4}{\delta}}{2m}},$$

where $C(m, p, \rho) = \frac{2}{\rho}\sqrt{\frac{\log pl}{m}} + \sqrt{\frac{\log pl}{\rho^2 m} \log\left[\frac{\rho^2 m}{\log pl}\right]} = \widetilde{O}\Big(\frac{1}{\rho}\sqrt{\frac{\log pl}{m}}\Big)$.

Proof. First, observe that the classification error of a deep cascade function $f \in \mathcal{T}_l$ only depends on its sign $\operatorname{sgn}(f)$. Let Δ denote the simplex in $\mathbb{R}^l$ and $\operatorname{int}(\Delta)$ its interior. For any $\boldsymbol{\alpha} \in \operatorname{int}(\Delta)$, define $g_{\boldsymbol{\alpha}}$ by

$$\forall x \in \mathcal{X},\ g_{\boldsymbol{\alpha}}(x) = \sum_{k=1}^{l} \alpha_k \mathsf{s}(x, k) h_k(x). \tag{3}$$

Then, $\operatorname{sgn}(f)$ coincides with $\operatorname{sgn}(g_{\boldsymbol{\alpha}})$ since $\mathsf{s}(x, k)$ is non-zero for exactly one value of k. We can therefore analyze $R(g_{\boldsymbol{\alpha}})$ instead of $R(f)$, for any $\boldsymbol{\alpha} \in \operatorname{int}(\Delta)$.

Now, since $g_{\boldsymbol{\alpha}}$ is a convex combination of the functions $x \mapsto \mathsf{s}(x, k)h_k(x)$, we can apply to the set of functions $g_{\boldsymbol{\alpha}}$ the learning guarantees for convex ensembles with multiple hypothesis sets given by [9]:

$$R(f) \leq \inf_{\boldsymbol{\alpha} \in \operatorname{int}(\Delta)} \left[\widehat{R}_{S,\rho}(g_{\boldsymbol{\alpha}}) + \frac{4}{\rho} \sum_{k=1}^{l} \alpha_k \widehat{\mathfrak{R}}_S(\mathcal{H}_k)\right] + C(m, p, \rho) + \sqrt{\frac{\log \frac{4}{\delta}}{2m}}. \tag{4}$$

This bound is not explicit and depends on the choice of $\boldsymbol{\alpha} \in \operatorname{int}(\Delta)$. The crux of our proof now consists of removing $\boldsymbol{\alpha}$ and deriving an explicit bound. The first term of the right-hand side of (4) can be re-written as $\inf_{\boldsymbol{\alpha} \in \operatorname{int}(\Delta)} A(\boldsymbol{\alpha})$ with

$$A(\boldsymbol{\alpha}) = \frac{1}{m} \sum_{k=1}^{l} \sum_{\mathsf{s}(x_i, k) = 1} 1_{y_i \alpha_k h_k(x_i) < \rho} + \frac{4}{\rho} \sum_{k=1}^{l} \alpha_k \widehat{\mathfrak{R}}_S(\mathcal{H}_k), \tag{5}$$

since $\widehat{R}_{S,\rho}(g_{\boldsymbol{\alpha}}) = \frac{1}{m}\sum_{k=1}^{l}\sum_{\mathsf{s}(x_i,k)=1} 1_{y_i\alpha_k h_k(x_i)<\rho}$. Observe that function A can be decoupled as a sum, $A(\boldsymbol{\alpha}) = \sum_{k=1}^{l} A_k(\alpha_k)$, where

$$A_k(\alpha_k) = \frac{1}{m}\sum_{\mathsf{s}(x_i,k)=1} 1_{y_i\alpha_k h_k(x_i)<\rho} + \frac{4}{\rho}\alpha_k r_k$$

with $r_k = \widehat{\mathfrak{R}}_S(\mathcal{H}_k)$. For any $k \in [1,l]$, $A_k(\alpha_k)$ can be rewritten as follows in terms of m_k^- and m_k^+: $A_k(\alpha_k) = \frac{m_k^-}{m} + \frac{m_k^+}{m}1_{\alpha_k<\rho} + \frac{4}{\rho}\alpha_k r_k$. This implies $\inf_{\alpha_k>0} A_k(\alpha_k) = \frac{m_k^-}{m} + \min\left(\frac{m_k^+}{m}, 4r_k\right)$. However, we need to ensure the global condition $\sum_{k=1}^{l}\alpha_k \leq 1$. First, we let $l' = \min(|\mathcal{K}|, \frac{1}{\rho})$. For any $\mathcal{J} \subseteq \mathcal{K}$ with $|\mathcal{J}| \leq l'$, we choose $\alpha_k = \rho$ for $k \in \mathcal{J}$, $\alpha_k \to 0$ otherwise, which guarantees $\sum_{k=1}^{l}\alpha_k = \rho l' \leq 1$ and gives the infimum

$$\inf_{\boldsymbol{\alpha}\in\mathrm{int}(\Delta)} A(\boldsymbol{\alpha}) \quad = \quad \min_{\substack{\mathcal{J}\subseteq\mathcal{K}\\ |\mathcal{J}|\leq l'}} \Big(4\sum_{k\in\mathcal{J}} r_k \;+\; \sum_{k\in\mathcal{K}-\mathcal{J}} \frac{m_k^+}{m}\Big) \;+\; \sum_{k=1}^{l}\frac{m_k^-}{m} \;+\; \sum_{k\notin\mathcal{K}}\frac{m_k^+}{m}.$$

In order to simplify the bound, observe that the following equalities hold:

$$\begin{aligned}
&\min_{\mathcal{J}}\Big(4\sum_{k\in\mathcal{J}} r_k + \sum_{k\in\mathcal{K}-\mathcal{J}}\frac{m_k^+}{m}\Big) = \min_{\mathcal{J}}\Big(4\sum_{k\in\mathcal{J}} r_k + \sum_{k\in\mathcal{K}-\mathcal{J}}\frac{m_k^+}{m} + \sum_{k\in\mathcal{K}-\mathcal{J}} 4r_k - \sum_{k\in\mathcal{K}-\mathcal{J}} 4r_k\Big)\\
&= \min_{\mathcal{J}}\Big(4\sum_{k\in\mathcal{K}} r_k + \sum_{k\in\mathcal{K}-\mathcal{J}}\frac{m_k^+}{m} - 4r_k\Big) = 4\sum_{k\in\mathcal{K}} r_k + \min_{\mathcal{J}}\Big(\sum_{k\in\mathcal{K}-\mathcal{J}}\frac{m_k^+}{m} - 4r_k\Big).
\end{aligned}$$

By definition, $\sum_{k\in\mathcal{K}} 4r_k + \sum_{k\notin\mathcal{K}}\frac{m_k^+}{m} = \sum_{k=1}^{l}\min\left(4r_k, \frac{m_k^+}{m}\right)$. Now, let $\mathcal{L} = \mathcal{K} - \mathcal{J}$ and since $|\mathcal{J}| \leq l'$, $|\mathcal{L}| = |\mathcal{K}| - |\mathcal{J}| \geq |\mathcal{K}| - l' = |\mathcal{K}| - \min(|\mathcal{K}|, \frac{1}{\rho}) = \max(0, |\mathcal{K}| - \frac{1}{\rho})$ thus, $|\mathcal{L}| \geq |\mathcal{K}| - \frac{1}{\rho}$. Finally, we write the bound in the following simpler form:

$$\inf_{\boldsymbol{\alpha}\in\mathrm{int}(\Delta)} A(\boldsymbol{\alpha}) = \sum_{k=1}^{l}\min\Big(4r_k, \frac{m_k^+}{m}\Big) + \min_{\substack{\mathcal{L}\subseteq\mathcal{K}\\ |\mathcal{L}|\geq|\mathcal{K}|-\frac{1}{\rho}}}\Big(\sum_{k\in\mathcal{L}}\frac{m_k^+}{m} - 4r_k\Big) + \sum_{k=1}^{l}\frac{m_k^-}{m}.$$

Since $\widehat{R}_S(f) = \sum_{k=1}^{l}\frac{m_k^-}{m}$, this coincides with the bound of the theorem. □

These learning bounds are not straightforward and cannot be derived from standard Rademacher complexity bounds. A finer analysis is used in the proof to relate deep cascades to convex ensembles with multiple hypothesis sets [9].

We already commented on the simpler form (2) of this generalization bound. Our comments apply a fortiori to this finer version of the bound. Let us add that the theorem also provides new learning guarantees in the special case of decision trees. The result may seem surprising since it suggests that the complexity term depends on m_k^+/m when this ratio is sufficiently small; however, for such nodes, typically the fraction of points m_k/m would also be small, where m_k denotes the number of points at leaf k.

At a deeper level, these guarantees suggest that for cascades, the complexity of the hypothesis sets may not be the most critical measure, but rather a balance of those complexities and the fractions of points.

The bound of the theorem can be generalized to hold uniformly for all $\rho > 0$ at the price of an additional term in $O\Big(\frac{\log\log_2 \frac{1}{\rho}}{m}\Big)$. For $|\mathcal{K}| \leq \frac{1}{\rho}$, choosing $\mathcal{L} = \emptyset$ yields:

$$R(f) \leq \widehat{R}_S(f) + \sum_{k=1}^{l} \min\Big(4\,\widehat{\mathfrak{R}}_S(\mathcal{H}_k), \frac{m_k^+}{m}\Big) + C(m,p,\rho) + \sqrt{\frac{\log\frac{4}{\delta}}{2m}}. \tag{6}$$

As mentioned above, these learning bounds can be generalized to hold uniformly over all $\rho > 0$: thus, we can choose $\rho = \frac{1}{|\mathcal{K}|}$ at the price of an additional term in the bound varying only in $O\Big(\frac{\log\log_2 |\mathcal{K}|}{m}\Big) \leq O\Big(\frac{\log\log_2 l}{m}\Big)$. This gives the simpler form (2) of the bound of Theorem 1, using $C(m,p,\rho) = C(m,p,\frac{1}{|\mathcal{K}|}) \leq C(m,p,\frac{1}{l})$.

The learning bounds just presented are given in terms of the empirical Rademacher complexities $\widehat{\mathfrak{R}}_S(\mathcal{H}_k)$. To derive more explicit guarantees, we must bound each of these quantities in terms of $\widehat{\mathfrak{R}}_S(H_k)$ and $\widehat{\mathfrak{R}}_S(\mathcal{S}_k)$. The following lemma helps us achieve that.

Lemma 1. *Let G_1 be a family of functions mapping $\mathcal{X}$ to $\{0,1\}$ and let G_2 be a family of functions mapping $\mathcal{X}$ to $\{-1,+1\}$. Let $G = \{g_1 g_2 \colon g_1 \in G_1, g_2 \in G_2\}$. Then, the empirical Rademacher complexity of G for any sample S of size m can be bounded as follows:*

$$\widehat{\mathfrak{R}}_S(G) \leq \widehat{\mathfrak{R}}_S(G_1) + \widehat{\mathfrak{R}}_S(G_2).$$

Proof. Observe that for $g_1 \in G_1$ and $g_2 \in G_2$, $g_1 g_2 = |g_1 + g_2| - 1$. Since $x \mapsto |x| - 1$ is 1-Lipschitz over $[-1,2]$, by Talagrand's lemma in [20], the following holds: $\widehat{\mathfrak{R}}_S(G) \leq \widehat{\mathfrak{R}}_S(G_1 + G_2) \leq \widehat{\mathfrak{R}}_S(G_1) + \widehat{\mathfrak{R}}_S(G_2)$. □

Thus, in view of the lemma, for any $k \in [1,p]$, we can use the upper bound $\widehat{\mathfrak{R}}_S(\mathcal{H}_k) \leq \widehat{\mathfrak{R}}_S(H_k) + \widehat{\mathfrak{R}}_S(\mathcal{S}_k)$.

We now assume, as previously discussed, that leaf selectors are defined via node questions $q_j \colon \mathcal{X} \to \{0,1\}$, with $q_j \in Q_j$. Thus, to derive more explicit guarantees in that case, we need to bound $\widehat{\mathfrak{R}}_S(\mathcal{S}_k)$ in terms of the Rademacher complexities $\widehat{\mathfrak{R}}_S(Q_j)$.

Lemma 2. *Let H_1 and H_2 be two families of functions mapping $\mathcal{X}$ to $\{0,1\}$ and let $H = \{h_1 h_2 \colon h_1 \in H_1, h_2 \in H_2\}$. Then, the empirical Rademacher complexity of H for any sample S of size m can be bounded as follows:*

$$\widehat{\mathfrak{R}}_S(H) \leq \widehat{\mathfrak{R}}_S(H_1) + \widehat{\mathfrak{R}}_S(H_2).$$

Proof. Observe that for any $h_1 \in H_1$ and $h_2 \in H_2$, we can write $h_1 h_2 = (h_1 + h_2 - 1)1_{h_1+h_2-1\geq 0} = (h_1 + h_2 - 1)_+$. Since $x \mapsto (x-1)_+$ is 1-Lipschitz over $[0,2]$, by Talagrand's lemma in [20], the following holds: $\widehat{\mathfrak{R}}_S(H) \leq \widehat{\mathfrak{R}}_S(H_1 + H_2) \leq \widehat{\mathfrak{R}}_S(H_1) + \widehat{\mathfrak{R}}_S(H_2)$. □

In view of Lemmas 2 and 1, the Rademacher complexities of the hypothesis sets $\mathcal{H}_k$ can be explicitly bounded as follows for any $k \in [1, l]$: $\widehat{\mathfrak{R}}_S(\mathcal{H}_k) \leq \sum_{j=1}^{d_k} \widehat{\mathfrak{R}}_S(Q_j) + \widehat{\mathfrak{R}}_S(H_k)$. Clearly, if the same hypothesis set is used for all node questions, that is $Q_j = Q$ for all j for some Q, then the bound admits the following simpler form: $\widehat{\mathfrak{R}}_S(\mathcal{H}_k) \leq d_k \widehat{\mathfrak{R}}_S(Q) + \widehat{\mathfrak{R}}_S(H_k)$. The Rademacher complexity of the hypothesis sets $\mathcal{H}_k$ can also be bounded in terms of the growth function of H_k and of Q_j (see full paper [11]).

To the best of our knowledge, Lemmas 2 and 1 are both novel and can be used as general tools for the analysis of the Rademacher complexity of other families. In the full version of this paper [11], we also provide a lower bound for the Rademacher complexity of the product of two hypothesis sets as a linear combination of the Rademacher complexity of the two sets. This shows that the upper bounds given by Lemma 2 cannot be significantly improved in general.

3.2 Leaf classifiers taking values in $[-1, +1]$

A similar but somewhat more complex analysis can be given in the case where the leaf classifiers take values in $[-1, +1]$. Define $\rho_k = \min\{y_i h_k(x_i) \colon y_i h_k(x_i) > 0, \mathsf{s}(x_i, k) = 1\}$ as the smallest confidence value over the correctly classified sample points at leaf k. If there are no correctly classified points, then define $\rho_k = 0$. Let $\widetilde{\mathcal{K}} = \left\{k \in [1, l] \colon \frac{m_k^+}{m} > \frac{4r_k}{\rho_k}\right\}$ and denote its weighted cardinality as $|\widetilde{\mathcal{K}}|_{\tilde{w}} = \sum_{k=1}^{l} \frac{1}{\rho_k}$. Then, it can be shown that for any $\delta > 0$, for $|\widetilde{\mathcal{K}}|_{\tilde{w}} \leq \frac{1}{\rho}$, the following holds with probability at least $1 - \delta$:

$$R(f) \leq \widehat{R}_S(f) + \sum_{k=1}^{l} \min\left(\frac{4\widehat{\mathfrak{R}}_S(\mathcal{H}_k)}{\rho_k}, \frac{m_k^+}{m}\right) + \widetilde{O}\left(l\sqrt{\frac{\log pl}{m}}\right), \tag{7}$$

which is the analogue of the learning bound (2) obtained in the case of leaf classifiers taking values in $\{-1, +1\}$. The full proof of this result, as well as that of more refined results, is given in the full version of this paper in [11]. As in the discrete case, to derive an explicit bound, we need to upper bound for all $k \in [1, l]$ the Rademacher complexity $\widehat{\mathfrak{R}}_S(\mathcal{H}_k)$ in terms of those of H_k and Q_j. To do so, we will need a new tool provided by the following lemma.

Lemma 3. *Let H_1 and H_2 be two families of functions mapping $\mathcal{X}$ to $[0, +1]$ and let F_1 and F_2 be two families of functions mapping $\mathcal{X}$ to $[-1, +1]$. Let $H = \{h_1 h_2 \colon h_1 \in H_1, h_2 \in H_2\}$ and let $F = \{f_1 f_2 \colon f_1 \in F_1, f_2 \in F_2\}$. Then, the empirical Rademacher complexities of H and F for any sample S of size m are bounded as follows:*

$$\widehat{\mathfrak{R}}_S(H) \leq \frac{3}{2}\big(\widehat{\mathfrak{R}}_S(H_1) + \widehat{\mathfrak{R}}_S(H_2)\big) \qquad \widehat{\mathfrak{R}}_S(F) \leq 2\big(\widehat{\mathfrak{R}}_S(F_1) + \widehat{\mathfrak{R}}_S(F_2)\big).$$

Proof. Observe that for any $h_1 \in H_1$ and $h_2 \in H_2$, we can write $h_1 h_2 = \frac{1}{4}[(h_1 + h_2)^2 - (h_1 - h_2)^2]$. For bounding the term $(h_1 + h_2)^2$, note that the function $x \mapsto \frac{1}{4}x^2$ is 1-Lipschitz over $[0, 2]$. For the term $(h_1 - h_2)^2$, observe that the function $x \mapsto$

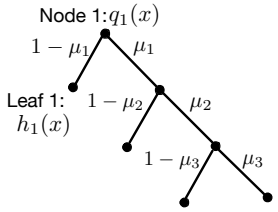

Fig. 1. Tree Topology of deep cascades for DEEPCASCADE and DEEPCASCADESVM Algorithm. The node question at node 1 is denoted by $q_1(x)$ and the leaf classifier at leaf 1 denoted by $h_1(x)$. A μ_k fraction of the points at node k is sent to the right child, and the remaining $(1-\mu_k)$ fraction of points to the left child. For DEEPCASCADE, all μ_ks are set to be equal: $\mu_k = \mu$ for all k.

$\frac{1}{4}x^2$ is $1/2$-Lipschitz over $[-1,1]$. Thus, by Talagrand's lemma (see [20]), $\widehat{\mathfrak{R}}_S(H) \leq \widehat{\mathfrak{R}}_S(H_1 + H_2) + \frac{1}{2}\widehat{\mathfrak{R}}_S(H_1 - H_2) \leq \frac{3}{2}\big(\widehat{\mathfrak{R}}_S(H_1) + \widehat{\mathfrak{R}}_S(H_2)\big)$. Similarly, the same equation holds for the product $f_1 f_2$ with $f_1 \in F_1$ and $f_2 \in F_2$, but now the function $x \mapsto \frac{1}{4}x^2$ is 1-Lipschitz over $[-2,2]$. Thus, by Talagrand's lemma [20], the following holds: $\widehat{\mathfrak{R}}_S(F) \leq \widehat{\mathfrak{R}}_S(F_1 + F_2) + \widehat{\mathfrak{R}}_S(F_1 - F_2) \leq 2\big(\widehat{\mathfrak{R}}_S(F_1) + \widehat{\mathfrak{R}}_S(F_2)\big)$, which completes the proof. □

Lemma 2 and Lemma 3 yield the following explicit bound for $\widehat{\mathfrak{R}}(\mathcal{H}_k)$ for any $k \in [1,l]$: $\widehat{\mathfrak{R}}_S(\mathcal{H}_k) \leq 2\sum_{j=1}^{d_k}\widehat{\mathfrak{R}}_S(Q_j) + 2\widehat{\mathfrak{R}}_S(H_k)$. When the same hypothesis set is used for all node questions, that is $Q_j = Q$ for all j for some Q, then the bound admits the following simpler form: $\widehat{\mathfrak{R}}_S(\mathcal{H}_k) \leq 2d_k\widehat{\mathfrak{R}}_S(Q) + 2\widehat{\mathfrak{R}}_S(H_k)$.

4 Algorithms

There are several algorithms that could be derived from the learning guarantees presented in the previous section. Here, we will describe two algorithms based on the simplest bound (2) of Section 3.1, which we further bound more explicitly by using the results from the previous section:

$$R(f) \leq \widehat{R}_S(f) + \sum_{k=1}^{l} \min\left(4\left[\sum_{j=1}^{d_k}\widehat{\mathfrak{R}}_S(Q_j) + \widehat{\mathfrak{R}}_S(H_k)\right], \frac{m_k^+}{m}\right) + \widetilde{O}\left(l\sqrt{\frac{\log pl}{m}}\right). \quad (8)$$

For both of our algorithms, we fix the topology of the deep cascade to be binary trees where every left child is a leaf as shown by Figure 2. Other more general tree topologies can be considered, which could further improve our results.

4.1 DEEPCASCADE

In this section, we describe a generic algorithm for deep cascades, named DEEPCASCADE. The algorithm first generates several deep cascades and then chooses the best among them by minimizing the generalization bound (8).

Algorithm 1 DEEPCASCADE($L, \mathcal{M}$)

$S_1 \leftarrow S$
for $l \in [1, \ldots, L], \mu \in \mathcal{M}, (H_k)_{1 \leq k \leq l} \subseteq \mathcal{H}, (Q_k)_{1 \leq k \leq l} \subseteq \mathcal{Q}$ **do**
  for $k = 1$ to l **do**
    $q_k \leftarrow \arg_{q \in Q_k} \{|q^{-1}(1) \cap S_k| = \mu |S_k|\}$
    $S_{k+1} \leftarrow q_k^{-1}(1) \cap S_k$
    $h_k \leftarrow \operatorname{argmin}_{h \in H_k} \{\hat{R}_{\bar{S}_{k+1}}(h) : \bar{S}_{k+1} = q_k^{-1}(0) \cap S_k\}$
  end for
  $f \leftarrow \sum_{k=1}^{l-1} (\prod_{j=1}^{k-1} q_j) \overline{q}_k h_k + (\prod_{j=1}^{l} q_j) h_l$
  $\mathcal{F} \leftarrow \mathcal{F} \cup \{f\}$
end for
$f^* \leftarrow \operatorname{argmin}_{f \in \mathcal{F}} R_S(f) + \sum_{k=1}^{l} \min\left(4\big(\sum_{j=1}^{d_k} \widehat{\mathfrak{R}}_S(Q_{f,j}) + \widehat{\mathfrak{R}}_S(H_{f,k})\big), \frac{m_k^+}{m}\right)$
return f^*

Let $\mathcal{H}$ be a set of p hypothesis sets from which the hypothesis sets H_k are selected. Here, we similarly allow each hypothesis set Q_k to be chosen out of a family of hypothesis sets $\mathcal{Q}$ of cardinality p – it is not hard to see that this affects our learning bound only by the $\log(pl)$ factors being changed into $l \log(p)$ and leaves the main terms we are minimizing unchanged; moreover, since we will be considering cascades with a relatively small depth, say $l \leq 4$, the effect will be essentially insignificant.

At any node k, the question q_k is selected so that a μ fraction of the points is sent to the right child. We assume for simplicity that for any node k and any choice μ, there exists a unique node question q_k with that property. For the topology of Figure 2, it is not hard to see that for any k, $\frac{m_k^+}{m}$ is at most μ^{k-1}. The parameter μ controls the fraction of points that proceed deeper into the tree and is introduced in order to find the best trade-off between the complexity term and fraction of points at each node. The subsample of the points reaching the internal node k is denoted by S_k and the subsample of those reaching leaf k is denoted by $\bar{S}_{k+1}$, with $|\bar{S}_{k+1}| = m_k$. The leaf classifier h_k is chosen to be the minimizer of the empirical error over $\bar{S}_{k+1}$ since, in this way, it further minimizes bound (8).

Algorithm 1 gives the pseudocode of DEEPCASCADE. The algorithm takes as input the maximum depth L for all the deep cascades generated and the set $\mathcal{M}$ of different fraction values for the parameter μ. For any depth $l \in [1, \ldots, L]$, any $\mu \in \mathcal{M}$, and any sequence of leaf hypothesis sets $(H_k)_{1 \leq k \leq l} \subseteq \mathcal{H}$ and sequence of node question hypothesis sets $(Q_k)_{1 \leq k \leq l} \subseteq \mathcal{Q}$, the algorithm defines a new deep cascade function f. At each node k, the question $q_k \in Q_k$ is selected with the μ-property already discussed and the leaf hypothesis $h_k \in H_k$ is selected to minimize the error over the leaf sample. For each f, we denote by $Q_{f,k}$ the question hypothesis set at node k that served to define f and similarly $H_{f,k}$ the hypothesis set at leaf k that was used to define f. The algorithm returns the deep cascade function f^* among these cascade functions f that minimizes the bound (8).

The empirical risk minimization (ERM) method used to determine the leaf classifiers h_k is intractable for some hypothesis sets. In the next section, we present an

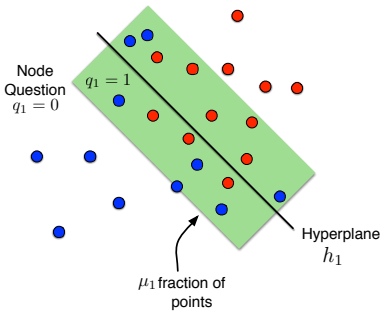

Fig. 2. Illustration of the first step of DEEPCASCADESVM. The hyperplane h_1 is learned using the SVM algorithm over sample points S_1. The node question q_1 equals one in the green area and zero otherwise. The green area contains a μ_1 fraction of the data points that will proceed to the next node.

alternative algorithm using SVMs which can be viewed as an efficient instantiation of this generic algorithm.

4.2 DEEPCASCADESVM

In this section, we describe an algorithm for learning deep cascades that makes use of SVMs and that we named DEEPCASCADESVM. In short, as with DEEPCASCADE, DEEPCASCADESVM first generates different deep cascade functions, but it uses the SVM algorithm at each node of the cascade and chooses the best among them by minimizing an upper bound of the generalization bound (8).

The deep cascade functions generated by the algorithm are based on repeatedly using SVMs combined with polynomial kernels of different degree. The leaf hypothesis sets H_k are decision surfaces defined by polynomial kernels. The hypothesis $h_k \in H_k$ is learned via the SVM algorithm with a polynomial kernel degree δ_k on subsample S_k. Note that in the pseudocode of Algorithm 2, we denote this step by SVM(δ_k, S_k). The node question hypothesis set Q_k is defined to be the set of indicator functions of $dist(h_k, x) \leq c$, where $dist(h_k, x)$ is the Euclidian distance of point x to hyperplane h_k in the feature space. The node question $q_k \in Q_k$ is chosen to be the indicator function of $dist(h_k, x) \leq c_k$ where c_k is such that $|q_k(1)^{-1} \cap S_k| = \mu_k |S_k|$, meaning the number of points in S_k that are within a distance c_k to hyperplane h_k equals $\mu_k |S_k|$. In other words, after learning the hyperplane via the SVM algorithm on subsample S_k, the algorithm

1. extracts a μ_k fraction of points closest to the hyperplane;
2. on the next node in the cascade, retrains on this extracted subsample using the SVM algorithm with a polynomial kernel of another degree.

We extract the fraction of points closest to the hyperplane because these points can be harder to classify correctly. Hence, these points will proceed deeper into the cascade in hope to find a better trade-off between the complexity and the fraction of correctly classified points.

Algorithm 2 DEEPCASCADESVM(L, $\mathcal{M}$, γ)

for $l \in [1, \dots, L], (\mu_k)_{1 \leq k \leq l} \subseteq \mathcal{M}, (\delta_k)_{1 \leq k \leq l} \subseteq \mathcal{G}$ **do**
  $S_1 \leftarrow S$
  for $k = 1$ to l **do**
    $h_k \leftarrow \text{SVM}(\delta_k, S_k)$
    $q_k \leftarrow \arg_{q \in Q_k} \{|q^{-1}(1) \cap S_k| = \mu_k |S_k|\}$
    $S_{k+1} \leftarrow q_k^{-1}(1) \cap S_k$
  end for
  $f(\cdot) = \sum_{k=1}^{l-1} (\prod_{j=1}^{k-1} q_j) \overline{q}_k h_k + (\prod_{j=1}^{l} q_j) h_l h_k(\cdot)$
  $\mathcal{F} \leftarrow \mathcal{F} \cup \{f\}$
end for

$$f^* \leftarrow \operatorname*{argmin}_{f \in \mathcal{F}} \widehat{R}_S(f) + \sum_{k=1}^{l} \min \left(4\gamma \left[\sum_{j=1}^{d_k} \sqrt{\frac{\mathsf{d}_{f,j} \log\left(\frac{em}{\mathsf{d}_{f,j}}\right)}{m}} + \sqrt{\frac{\mathsf{d}_{f,k} \log\left(\frac{em}{\mathsf{d}_{f,k}}\right)}{m}} \right], \frac{m_k^+}{m} \right)$$

return f^*

The algorithm generates several cascades functions with a given depth $l \in [1, \dots, L]$. For any depth $l \in [1, \dots, L]$, any sequence of fraction values $(\mu_k)_{1 \leq k \leq l} \subseteq \mathcal{M}$ and sequence of degree values $(\delta_k)_{1 \leq k \leq l} \subseteq \mathcal{G}$, the algorithm defines a new deep cascade function f. At each node k, the question $q_k \in Q_k$ and leaf hypothesis $h_k \in H_k$ are selected as already discussed. Similarly, as before, for each f, we denote by $Q_{f,k}$ the question hypothesis set at node k that served to define f and similarly $H_{f,k}$ the hypothesis set at leaf k that was used to define f. The best cascade f^* is chosen by minimizing an upper bound of the generalization bound (8). More precisely, we first bound the Rademacher complexity in terms of the VC-dimension of the hypothesis set:

$$\sum_{j=1}^{d_k} \widehat{\mathfrak{R}}_S(Q_{f,j}) + \widehat{\mathfrak{R}}_S(H_{f,k}) \leq \sum_{j=1}^{d_k} \sqrt{\frac{\mathsf{d}_{f,j} \log(\frac{em}{\mathsf{d}_{f,j}})}{m}} + \sqrt{\frac{\mathsf{d}_{f,k} \log(\frac{em}{\mathsf{d}_{f,k}})}{m}},$$

where $\mathsf{d}_{f,k}$ is the VC-dimension of $H_{f,k}$ and where we used the fact that $\text{VCdim}(Q_{f,j}) \leq \text{Pdim}(H_{f,j}) = \text{VCdim}(H_{f,j}) = \mathsf{d}_{f,j}$. Then, we rescale the complexity term by a parameter γ, which we will determine by cross-validation. Thus, for a given γ, we chose the deep cascade with the smallest value of the generalization bound :

$$R(f) \leq \widehat{R}_S(f) + \sum_{k=1}^{l} \min \left(4\gamma \left[\sum_{j=1}^{d_k} \sqrt{\frac{\mathsf{d}_{f,j} \log(\frac{em}{\mathsf{d}_{f,j}})}{m}} + \sqrt{\frac{\mathsf{d}_{f,k} \log(\frac{em}{\mathsf{d}_{f,k}})}{m}} \right], \frac{m_k^+}{m} \right). \quad (9)$$

DEEPCASCADESVM can be seen as a tractable version of the generic DEEPCASCADE algorithm with some minor differences in the following ways. Instead of choosing h_k to be the minimizer of the empirical error as done in DEEPCASCADE, the DEEPCASCADESVM chooses the h_k that minimizes a surrogate loss (hinge loss) of the empirical error by using the SVM algorithm. In fact, the γ parameter is introduced because the hinge loss used in the SVM algorithm needs to be re-scaled. Note that h_k is learned via the SVM algorithm on S_k and not on $\bar{S}_{k+1}$, namely the points that reach leaf k, as in the DEEPCASCADE algorithm. One could retrain SVM on the points reaching the leaf

Table 1. Results for DEEPCASCADESVM algorithm. The table reports the average test error and standard deviation for DEEPCASCADESVM(γ^*) and for the SVM algorithm. For each data set, the table also indicates the sample size, the number of features, and the depth of the cascade.

Dataset	Number of Examples	Number of Features	SVM Algorithm	DEEPCASCADESVM	Cascade Depth
breastcancer	683	10	0.0426 ± 0.0117	0.0353 ± 0.00975	4
german	1,000	24	0.297 ± 0.0193	0.256 ± 0.0324	4
splice	1,000	60	0.205 ± 0.0134	0.175 ± 0.0152	3
ionosphere	351	34	0.0971 ± 0.0167	0.117 ± 0.0229	4
a1a	1,000	123	0.195 ± 0.0217	0.209 ± 0.0233	2

to be consistent with the first algorithm, but this typically will not change the hypothesis h_k. The generic node question q_k of DEEPCASCADE are picked to be the distance to the classification hyperplane h_k for a given fraction μ_k of points in DEEPCASCADESVM algorithm. Technically, in the DEEPCASCADE, the μ fractions are the same, but this was done to simplify the exposition of the DEEPCASCADE algorithm. DEEPCASCADE minimizes exactly bound (8), while DEEPCASCADESVM minimizes an upper bound in terms of the VC-dimension.

5 Experiments

This section reports the results of some preliminary experiments with the DEEPCASCADESVM algorithm on several UC Irvine data sets. Since DEEPCASCADESVM uses only polynomial kernels as predictors, we similarly compared our results with those achieved by the SVM algorithm with polynomial kernels over the set $\mathcal{G}$ of polynomial degrees. Of course, a similar set of experiments can be carried out by using both Gaussian kernels or other kernels, which we plan to do in the future.

For our experiments, we used five different data sets from UC Irvine's data repository, http://archive.ics.uci.edu/ml/datasets.html: `breastcancer`, `german` (numeric), `ionosphere`, `splice`, and `a1a`. Table 1 gives the sample size and the number of features for each of these data sets. For each of them, we randomly divided the set into five folds and ran the algorithm five times using a different assignment of folds to the training set, validation set, and test set. For each $j \in \{0, 1, 2, 3, 4\}$, the sample points from the fold j was used for testing, the fold $j + 1$ (mod 5) used for validation, and the remaining sample points used for training.

The following are the parameters used for DEEPCASCADESVM: the maximum tree depth was set to $L = 4$, the set of fraction values was selected to be $\mathcal{M} = \{\frac{i}{10} : i = 1, \cdots, 10\}$ and the set of polynomial degrees $\mathcal{G} = \{1, \ldots, 4\}$. The regularization parameter $C_\delta \in \{10^i : i = -3, \cdots, 2\}$ of SVMs was selected via cross-validation for each polynomial degree $\delta \in \mathcal{G}$. To avoid a grid search at each node, for cascades, the regularization parameter C_{δ_k} for SVMs at node k was simply defined to be $\sqrt{\frac{m_k}{m}} C_\delta$ when using a polynomial degree δ_k.

For each value of the parameter $\gamma \in \{10^i : i = -2, \ldots, 0\}$, we generated several deep cascades and then chose the one that minimized the bound (9). Thus, for each γ, there was a corresponding deep cascade f^*_γ. The parameter γ was chosen via cross-

validation. More precisely, we chose the best γ^* by finding the deep cascade $f^*_{\gamma^*}$ that had the smallest validation error among the deep cascade functions f^*_γ. We report the average test error of the deep cascade $f^*_{\gamma^*}$ in Table 1. For SVMs, we report the test errors for the polynomial degree and regularization parameter with the smallest validation error.

The results of Table 1 show that DEEPCASCADESVM outperforms SVMs for three out of the five data sets: `breastcancer`, `german`, and `splice`. The `german` and `splice` results are statistically significant at the 5% level using a one-sided paired t-test while `breastcancer` result is not statistically significant. For the remaining two data sets where SVMs outperforms DEEPCASCADESVM, the `a1a` result is statistically significant at the 5% level while it is not statistically significant for the `ionosphere` data set.

Overall, the results demonstrate the benefits of DEEPCASCADESVM in several data sets. Note also that SVMs can be viewed as a special instance of the deep cascades with depth one. It is conceivable of course that for some data sets such simpler cascades would provide a better performance. There are several components in our algorithm that could be optimized more effectively to further improve performance. This includes optimizing over the regularization parameter C at each node of the cascade, testing polynomial degrees higher than 4, or searching over larger sets of μ fraction values and γ values. Yet, even with this rudimentary implementation of an algorithm that minimizes the simplest form of our bound (8), it is striking that it outperforms SVMs for several of the data sets and finds a comparable accuracy for the remaining data sets. More extensive experiments with other variants of the algorithms would be interesting to investigate in the future.

6 Conclusion

We presented two algorithms for learning Deep Cascades, a broad family of hierarchical models which offer the flexibility of selecting node or leaf functions from unions of complex hypothesis sets. We further reported the results of experiments demonstrating the performance for one of our algorithms using different data sets. Our algorithms benefit from data-dependent learning guarantees we derived, which are expressed in terms of the Rademacher complexities of the sub-families composing these sets of predictors and the fraction of sample points correctly classified at each leaf. Our theoretical analysis is general and can help guide the design of many other algorithms: different sub-families of leaf or node questions can be chosen and alternative cascade topologies and parameters can be selected. For the design of our algorithms, we used a simpler version of our guarantees. Finer algorithms could be devised to more closely exploit the quantities appearing in our learning bounds, which could further improve prediction accuracy.

Acknowledgments

We thank Vitaly Kuznetsov and Andrés Muñoz Medina for comments on an earlier draft of this paper. This work was partly funded by the NSF award IIS-1117591 and an NSF Graduate Research Fellowship.

References

1. Arreola, K., Fehr, J., Burkhardt, H.: Fast support vector machine classification using linear SVMs. In: *ICPR*. (2006)
2. Arreola, K., Fehr, J., Burkhardt, H.: Fast support vector machine classification of very large datasets. In: *GfKl Conference*. (2007)
3. Bartlett, P., Mendelson, S.: Rademacher and Gaussian complexities: Risk bounds and structural results. *JMLR*. (2002)
4. Bengio, S., Weston, J., Weston, D.: Label embedding trees for large multi-class tasks. In: *NIPS*. Vancouver, Canada, (2010)
5. Bennet, K., Blue, J.: A support vector machine approach to decision trees. In: *IJCNN*. Anchorage, Alaska, (1998)
6. Breiman, L., Friedman, J., Olshen, R., Stone, C.: *Classification and Regression Trees*. Wadsworth and Brooks. Monterey, CA, (1984)
7. Chang, F., Guo, C., Lin, X., Lu, C.: Tree decomposition for large-scale SVM problems. *JMLR*. (2010)
8. Chen, M., Xu, Z., Kedem, D., Chapelle, O.: Classifier cascade for minimizing feature evaluation cost. In: *AISTATS*. La Palma, Canary Islands, (2012)
9. Cortes, C., Mohri, M., Syed, U.: Deep boosting. In: *ICML*, (2014)
10. Deng, J., Satheesh, S., Berg, A., Fei-Fei, L.: Fast and balanced: Efficient label tree learning for large scale object recognition. In: *NIPS*. (2011)
11. DeSalvo, G., Mohri, M., Syed, U.: Learning with Deep Cascades. *arXiv*. (2015)
12. Dong, G., Chen, J.: Study on support vector machine based decision tree and application. In: *ICNC-FSKD*. Jinan, China, (2008)
13. Golea, M., Bartlett, P., Lee, W., Mason, L.: Generalization in decision trees and DNF: Does size matter? In: *NIPS*. (1997)
14. Koltchinskii, V., Panchenko, D.: Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*. 30, (2002)
15. Kumar, A., Gopal, M.: A hybrid SVM based decision tree. *JPR*. (2010)
16. Lefakis, L., Fleuret, F. Joint cascade optimization using a product of boosted classifiers. In: *NIPS*. (2010)
17. Littman, M., Li, L., Walsh, T.: Knows what it knows: A framework for self-aware learning. In: *ICML*. (2008)
18. Madjarov, G., Gjorgjevikj, D.: Hybrid decision tree architecture utilizing local SVMs for multi-label classification. In: *HAIS*. Salamanca, Spain, (2012)
19. Mansour, Y., McAllester, D.: Generalization bounds for decision trees. In: *COLT*. (2000)
20. Mohri, M., Rostamizadeh, R., Talwalkar, A.: Foundations of Machine Learning. *The MIT Press.* (2012)
21. Nobel, A.: Analysis of a complexity based pruning scheme for classification trees. *IEEE Trans. Inf. Theory*. (2002)
22. Pujara, J., Daume, H., Getoor, L.: Using classifier cascades for scalable e-mail classification. In: *CEAS*. (2011)
23. Quinlan, J.: Induction of decision trees. *Machine Learning*. 1(1):81–106, (1986)
24. Rodriguez-Lujan, I., Cruz, C., Huerta, R.: Hierarchical linear SVM. *JPR*. (2012)
25. Saberian, M., Vasconcelos, N.: Boosting classifier cascades. In: *NIPS*. Canada, (2010)
26. Scott, C., Nowak, R.: On adaptive properties of decision trees. In: *NIPS*. Canada, (2005)
27. Storcheus, D., Mohri, M., and Rostamizadeh, A. Foundations of Coupled Nonlinear Dimensionality Reduction. In: *arXiv*, (2015).
28. Takahashi, F., Abe, S.: Decision tree based multiclass SVMs. In: *ICONIP*. (2002)
29. Viola, P., Jones, M.: Robust real-time face detection. *IJCV*. (2004)

30. Wang, J., Saligrama, V.: Local supervised learning through space partitioning. In: *NIPS*. (2012)
31. Xu, Z., Kusner, M., Weinberger, K., Chen, M.: Cost-sensitive tree of classifiers. In: *ICML*. Altanta, USA, (2013)