Functional Gradient Boosting for Learning Residual-like Networks with Statistical Guarantees

Atsushi Nitanda[1,2,3]
nitanda@mist.i.u-tokyo.ac.jp

Taiji Suzuki[1,2]
taiji@mist.i.u-tokyo.ac.jp

[1]Graduate School of Information Science and Technology, The University of Tokyo
[2]Center for Advanced Intelligence Project, RIKEN
[3]PRESTO, Japan Science and Technology Agency

Abstract

Recently, several studies have proposed progressive or sequential layer-wise training methods based on the boosting theory for deep neural networks. However, most studies lack the global convergence guarantees or require *weak learning conditions* that can be verified *a posteriori* after running methods. Moreover, generalization bounds usually have a worse dependence on the network depth. In this paper, to resolve these problems, we propose a new functional gradient boosting for learning deep residual-like networks in a layer-wise fashion with its statistical guarantees on multi-class classification tasks. In the proposed method, each residual block is recognized as a functional gradient (i.e., weak learner), and the functional gradient step is performed by stacking it on the network, resulting in a strong optimization ability. In the theoretical analysis, we show the global convergence of the method under a standard margin assumption on a data distribution instead of a weak learning condition, and we eliminate a worse dependence on the network depth in a generalization bound via a fine-grained convergence analysis. Moreover, we show that the existence of a learnable function with a large margin on a training dataset significantly improves a generalization bound. Finally, we experimentally demonstrate that our proposed method is certainly useful for learning deep residual networks.

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. Copyright 2020 by the author(s).

1 Introduction

Residual networks (ResNets) (He et al., 2016) are state-of-the-art models equipped with a notable structure called *skip connections* or *resblocks*. Thus, several studies have been devoted to understanding the reason for their success. Recently, interesting aspects of ResNets have been reported; one is the ensemble view (Veit et al., 2016; Littwin and Wolf, 2016), where resblocks are assembled as weak learners to make a stronger model, and the other is the optimization or ordinary differential equations view (Jastrzebski et al., 2017), where resblocks provide a discretization of gradient flows or ordinary differential equations to minimize an objective function. Motivated by these perspectives and gradient boosting machines (Mason et al., 1999; Friedman, 2001; Chen and Guestrin, 2016; Ke et al., 2017), functional gradient boosting (*ResFGB*) (Nitanda and Suzuki, 2018a) for learning deep ResNets has been proposed. In their method, each resblock can be recognized as an approximation to a functional gradient, and functional gradient descent is performed by adding a resblock on the top of the feature extraction layer to optimize a network in a function space. Simultaneously, another boosting method (Huang et al., 2018) for ResNets has been proposed for the same purpose.

Usually, a generalization bound is composed of an approximation term and a variance term. In Huang et al. (2018); Nitanda and Suzuki (2018a), the approximation term for a network obtained in their method was evaluated through an algorithmic convergence rate analysis of the method, and the variance term was estimated by utilizing a conventional analysis of Rademacher complexity (Koltchinskii and Panchenko, 2002) for neural networks. In their analysis, a type of weak learning condition which is verified only after running algorithms was assumed to guarantee global convergence as well as many other boosting methods. In the generalization bounds, linear and exponential dependence on the network depth appeared in Huang et al. (2018)

and Nitanda and Suzuki (2018a), respectively, which is usually unavoidable in analyses using Rademacher complexity with norm constraints (Neyshabur et al., 2015) due to a lack of understanding of the approximation ability. That is, a generalization bounds is getting worse theoretically as the number of layers increases.

In this paper, we further develop this line of research. We first propose a new functional gradient boosting called *ResFGB-FW* for multi-class classification problems by combining ResFGB (Nitanda and Suzuki, 2018a) with the Frank-Wolfe method (Frank and Wolfe, 1956; Guélat and Marcotte, 1986; Jaggi, 2013) which is a basic scheme of boosting methods, leading to a slight modification of the network architecture of ResNets. In a theoretical analysis, we show our network architecture is helpful in ensuring the global convergence and provide a generalization bound without a type of weak learning condition. In addition, we demonstrate that a worse dependence on the depth can be eliminated by precisely estimating the required norms of parameters for global convergence of ResFGB-FW via a fine-grained convergence analysis. A key step in the proof for global convergence is to show that an obtained weak learner in the method exhibits comparable performance with a pure functional gradient in terms of optimization ability in a function space. We show this property by making a margin assumption where a given training dataset is separable with a sufficient margin by a good classifier in a set of learnable functions assembled by a shallow neural network, while this type of property is usually guaranteed by a weak learning condition in other methods. As a result, we can show not only that a worse dependence on the network depth can be eliminated but also that the existence of a good classifier providing a large margin can significantly improve a generalization bound. Finally, we experimentally compare our proposed method with state-of-the-art methods and demonstrate certain superior performances.

Contributions

- We propose a new functional gradient boosting for learning deep ResNets by combining ResFGB with the Frank-Wolfe method and show superior empirical performance over existing methods on multiclass classification tasks.
- We provide a generalization bound for a trained ResNet via a global convergence analysis of the method under a margin condition instead of a weak learning condition. Theoretical contributions in this study are that (i) we guarantee the global convergence without a type of weak learning condition, (ii) we show that our network architecture is helpful in ensuring the convergence, (iii) we eliminate a worse dependence on the network depth in a generalization bound via a convergence analysis.

Related Work

Recently, it is known that a progressive learning scheme or sequential layer-wise training has a powerful generalization ability as well as a conventional learning scheme on the image classification task (Belilovsky et al., 2019; Nøkland and Eidnes, 2019) and it is useful in network architecture search (Liu et al., 2018; Kim et al., 2018; Weill et al., 2019) which is considered as one of important tasks in deep learning community. So far, many studies (Han et al., 2016; Cortes et al., 2017; Mosca and Magoulas, 2017; Wang et al., 2017; Kulkarni and Karande, 2017; Huang et al., 2018; Brock et al., 2017; Opitz et al., 2017; Malach and Shalev-Shwartz, 2018; Marquez et al., 2018; Nitanda and Suzuki, 2018b,a; Belilovsky et al., 2019; Nøkland and Eidnes, 2019) have developed progressive learning methods and have analyzed the theoretical properties of these methods.

Especially, AdaNet (Cortes et al., 2017), BoostResNet (Huang et al., 2018), and ResFGB (Nitanda and Suzuki, 2018a) are closely related to our method in the sense that components of neural networks are iteratively added to optimize a network structure based on the boosting theory. However, our method has several different theoretical properties from these methods, for instance, BoostResNet and ResFGB required weak learning conditions to guarantee the global convergence of the methods and linear and exponential dependence on the network depth appeared in their generalization bounds. On the other hand, our theory guarantees the global convergence without such a weak learning condition and eliminates a worse dependence on the depth in a generalization bound. As for AdaNet, the dependence on the network depth in the variance term (i.e., estimation error) was logarithmic, but the approximation guarantee (i.e., the global convergence guarantee) was not provided in their theory.

2 Preliminary

In this section, we provide several notations to describe a problem setting of the classification and a notion of functional gradients used in the proposed method and theoretical analyses. Moreover, we provide a basic skeleton of generalization bound to derive an explicit bound later.

2.1 Problem setting

Let $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y}$ be a feature space and a finite label set of cardinality c. We denote by ν a true Borel probability measure on $\mathcal{X} \times \mathcal{Y}$ and by ν_n an empirical probability measure by observations $(x_i, y_i)_{i=1}^n$ independently drawn from ν, i.e., $d\nu_n(X,Y) = \sum_{i=1}^n \delta_{(x_i,y_i)}(X,Y)dXdY/n$, where δ denotes the Dirac delta function. For ν_n, we denote by ν_n^X the marginal distribution on X. In general, for a probability measure μ, we denote by $\mathbb{E}_\mu$ the expectation with respect to a random variable according to μ. We denote by $L_2(\nu_n^X)$ the real-valued function space on $\mathcal{X}$ equipped with inner product $\langle\cdot,\cdot\rangle_{L_2(\nu_n^X)}$ and by $L_2^m(\nu_n^X)$ the product space of $L_2(\nu_n^X)$: for $\forall\xi, \forall\zeta \in L_2^m(\nu_n^X)$,

$$\langle \xi, \zeta \rangle_{L_2^m(\nu_n^X)} \overset{def}{=} \mathbb{E}_{\nu_n^X}\left[\sum_{j=1}^m \xi_j(X)\zeta_j(X)\right].$$

L_p-norm is defined by: $\|\xi\|_{L_p^m(\nu_n^X)}^p \overset{def}{=} \mathbb{E}_{\nu_n^X}[\|\xi(X)\|_p^p] = \mathbb{E}_{\nu_n^X}\left[\sum_{j=1}^m |\xi_j(X)|^p\right]$ for $p \in \{1,2\}$ and $\xi \in L_2^m(\nu_n^X)$.

Let $\mathcal{F}$ be a given function class of predictors. The problem considered in this paper is formalized as the empirical risk minimization problem:

$$\min_{f\in\mathcal{F}} \left\{ \mathcal{L}_n(f) \overset{def}{=} \mathbb{E}_{\nu_n}[l(f(X),Y)] \right\}, \tag{1}$$

where l be the logistic loss function for multiclass classification problems:

$$l(\zeta, y) = -\log\left(\frac{\exp(\zeta_y)}{\sum_{\overline{y}\in\mathcal{Y}} \exp(\zeta_{\overline{y}})}\right).$$

Generally, regularization or constraint is needed for a function class to guarantee generalization. Thus, we next describe a network architecture belonging to $\mathcal{F}$.

2.2 Network architecture

We split the predictor f into the feature extractor and linear predictor: $f(x) = w^\top \phi(x)$, where $w \in \mathbb{R}^{d\times c}$ is a weight for the last layer and ϕ is a feature extractor from $\mathcal{X}$ to $\mathcal{X}$. For simplicity, we pre-train $\overline{w}$ on a distribution ν_n as a linear classifier and fix it before optimizing ϕ. In addition, we assume $\|(\overline{w})_{*,y}\|_1 \le \Lambda_w$ for $\forall y \in \mathcal{Y}$. Thus, we focus on the problem of learning a feature extractor ϕ and denote $\mathcal{R}_n(\phi) = \mathcal{L}_n(\overline{w}^\top\phi)$. We herein describe the entire network architecture that is a type of residual networks with slight modifications (see Figure 1).

Definition 1. *Let $\mathcal{B}_1$ and $\mathcal{B}_2$ be sets of neural networks: $\mathcal{X} \to \mathcal{X}$, including the zero function. We denote by ϕ_t the output of ϕ at the t-th layer. A set of residual blocks (resblocks) at the t-th layer is*

$$\overline{\mathcal{B}}_{\phi_t} \overset{def}{=} \{\iota_1 + \iota_2 \circ \phi_t \mid \iota_1 \in \mathcal{B}_1,\ \iota_2 \in \mathcal{B}_2\}.$$

A connection between t- and $(t+1)$-th layers of ϕ is described as follows:

$$\phi_{t+1}(x) \overset{def}{=} (1-\eta_t)\phi_t(x) + \eta_t\overline{\iota}_t(x)$$
$$= ((1-\eta_t)id + \eta_t\iota_2) \circ \phi_t(x) + \eta_t\iota_1(x),$$

where $\eta_t \in [0,1)$ and $\overline{\iota}_t \in \overline{\mathcal{B}}_{\phi_t}$. A class of predictors is defined as follows: for the depth $T \in \mathbb{Z}_+$,

$$\mathcal{F} \overset{def}{=} \left\{\overline{w}^\top\phi_T \mid \overline{\iota}_t \in \overline{\mathcal{B}}_{\phi_t},\ t \in \{0,\ldots,T-1\}\right\}.$$

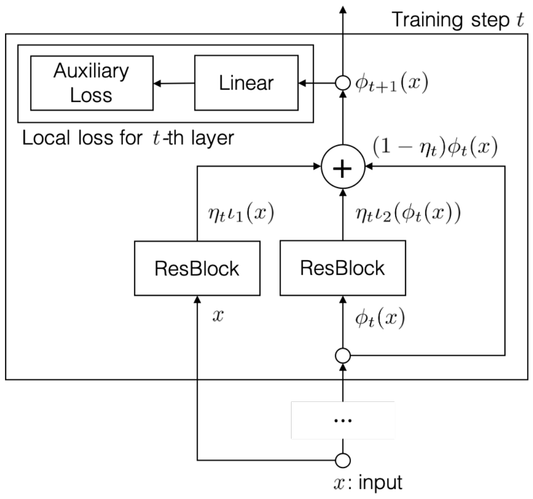

Figure 1: High level diagram of t-th layer of the network obtained in the proposed method. A linear classifier and an auxiliary loss are used to train resblocks at t-layer and discarded after training t-th layer.

Remark Our model has two major differences from the conventional residual networks and the model in Nitanda and Suzuki (2018a); one is that the output $\phi_t(x)$ is reduced at the rate of $1-\eta_t$ stemming from the Frank-Wolfe method and the other is that resblocks receive input data directly like DenseNet (Huang et al., 2017). These modifications lead to provable statistical guarantees without weak learning assumptions.

To provide theoretical analyses, some restrictions on resblocks are required.

Assumption 1. *Let Λ_i^C, Λ_i^D $(i \in \{1,2\})$ be positive values and σ be a 1-Lipschitz activation function. Assume that residual blocks are shallow neural networks defined as follows:*

$$\mathcal{B}_i = \Big\{ z \mapsto C_i\sigma(D_iz) \mid \max_j \|(C_i)_{j,*}\|_1 \le \Lambda_i^C,$$
$$\max_k \|(D_i)_{k,*}\|_1 \le \Lambda_i^D, C_i \in \mathbb{R}^{d\times d'}, D_i \in \mathbb{R}^{d'\times d} \Big\}.$$

Assumption 1 is used for guaranteeing the generalization bound. Although, we adopt shallow networks as resblocks in this condition for simplicity, but we can extend them to deeper resblocks. A typical example of σ is the sigmoid function.

2.3 Functional gradient

The key notion used in this study is the functional gradient in a function space. We define

$$\nabla_\phi \mathcal{R}_n(\phi)(x) \overset{def}{=} \begin{cases} \partial_z l(\overline{w}^\top z, y_i)\big|_{z=\phi(x_i)} & (x = x_i), \\ 0 & (\text{otherwise}). \end{cases}$$

This is nothing but a functional gradient (Fréchet differential) in $L_2^d(\nu_n^X)$, i.e., for $\forall \phi, \forall \xi \in L_2^d(\nu_n^X)$,

$$\mathcal{R}_n(\phi+\xi) = \mathcal{R}_n(\phi) + \langle \nabla_\phi \mathcal{R}_n(\phi), \xi \rangle_{L_2^d(\nu_n^X)} + o(\|\xi\|_{L_2^d(\nu_n^X)}).$$

Thus, the functional gradient descent using $\nabla_\phi \mathcal{R}_n$ optimizes $\mathcal{R}_n$ in $L_2^d(\nu_n^X)$, but we need a smoothing technique introduced later to guarantee generalization because $\nabla \mathcal{R}_n$ has no information on unseen data; in other words, this method is meaningless for expected minimization problems. We similarly define a functional gradient $\nabla_f \mathcal{L}_n(f)$ and see $\nabla_\phi \mathcal{R}_n(\phi) = \overline{w} \nabla_f \mathcal{L}_n(\overline{w}^\top \phi)$

The following proposition (Nitanda and Suzuki, 2018a) shows the Fréchet differentiability and Lipschitz smoothness for the multiclass logistic loss.

Proposition 1 (Nitanda and Suzuki (2018a))**.** *$\mathcal{R}_n(\phi)$ is convex and Fréchet differentiable in $L_2^d(\nu_n^X)$. Moreover, there exists $\exists L > 0$ such that $\forall \phi, \forall \xi \in L_2^d(\nu_n^X)$,*

$$\mathcal{R}_n(\psi + \xi) \le \mathcal{R}_n(\phi) + \langle \nabla_\phi \mathcal{R}_n(\phi), \xi \rangle_{L_2^d(\nu_n^X)} + \frac{L}{2}\|\xi\|^2_{L_2^d(\nu_n^X)}.$$

Remark Although, the Lipschitz parameter of $\mathcal{R}_n(\phi)$ depends on $\overline{w}$, we consider it as a uniform constant because $\overline{w}$ is fixed in our theoretical analysis.

2.4 Generalization Bound

We provide a skeleton of generalization bound on an expected classification error which is derived by a conventional analysis of Rademacher complexity for neural networks (Koltchinskii and Panchenko, 2002). This bound is composed of the L_1-functional gradient norm and the conventional margin defined below:

$$\|\nabla_f \mathcal{L}_n(f)\|_{L_1^c(\nu_n^X)} \overset{def}{=} \frac{1}{n}\sum_{i=1}^n \sum_{y \in \mathcal{Y}} \left|\partial_{\zeta_y} l(f(x_i), y_i)\right|,$$

$$m_f(x,y) \overset{def}{=} f_y(x) - \max_{y' \neq y} f_{y'}(x).$$

Theorem 1. *Suppose Assumption 1 holds, $\Lambda_2^C \Lambda_2^D \le 1/2$, $\eta_t \le 1$, and $\|x\|_2 \le \Lambda_\infty$ on $\mathcal{X}$. Fix $\delta > 0$. Then, for $\forall \rho > 0$, with probability at least $1 - \rho$ over the random choice from ν^n, we have that for $\forall f \in \mathcal{F}$ with the network depth $T \in \mathbb{Z}_+$,*

$$\mathbb{P}_\nu[m_f(X,Y) \le 0] \le \frac{2c^3 \Lambda_w \Lambda_\infty}{\delta \sqrt{n}} \left(1 + \sum_{t=0}^{T-1} \eta_t \Lambda_1^C \Lambda_1^D \prod_{s=t+1}^{T-1} \left(1 - \frac{\eta_s}{2}\right)\right) + \sqrt{\frac{1}{2n}\log \rho^{-1}} + \frac{1}{2}(1 + \exp(\delta)) \|\nabla_f \mathcal{L}_n(f)\|_{L_1^c(\nu_n^X)}.$$

A remaining problem for deriving an explicit generalization bound is to confirm how small functional gradient norms can be achievable by given rates of η_t and upper bounds: $\Lambda_1^C, \Lambda_1^D, \Lambda_2^C$, and Λ_2^D. We estimate these quantities by analyzing a convergence behavior of a proposed functional gradient boosting for learning deep residual networks.

3 Brief Review of Functional Gradient

Functional gradient methods including gradient boosting (Mason et al., 1999; Friedman, 2001) and gradient descent for kernel methods (Kivinen et al., 2004; Smale and Yao, 2006; Ying and Zhou, 2006; Raskutti et al., 2014; Wei et al., 2017) and over-parameterized neural networks (Jacot et al., 2018; Du et al., 2018; Arora et al., 2019) have powerful optimization ability because these methods naturally perform in function spaces.

In gradient boosting, $\nabla_\phi \mathcal{R}_n(\phi)$ is approximated in a given set of weak learners $\mathcal{G}$ as follows:

$$\xi_\phi \in \operatorname*{argmax}_{\xi \in \mathcal{G}} \langle \nabla_\phi \mathcal{R}_n(\phi), \xi \rangle_{L_2^d(\nu_n^X)} \tag{2}$$

and gradient method in a function space is performed using a descent direction $-\xi_\phi$. This approximation procedure is recognized as a type of smoothing of functional gradients.

In particular, this smoothing procedure is realized by using the following *kernel smoothing technique* in the kernel method and over-parameterized neural networks:

$$\begin{aligned} T_k \nabla_\phi \mathcal{R}_n(\phi) &\overset{def}{=} \mathbb{E}_{\nu_n^X}[\nabla_\phi \mathcal{R}_n(\phi)(X) k(X, \cdot)] \\ &= \frac{1}{n}\sum_{i=1}^n \nabla_\phi \mathcal{R}_n(\phi)(x_i) k(x_i, \cdot), \end{aligned} \tag{3}$$

where, k is a kernel function that is usually prefixed in kernel methods; on the contrary, it depends on a map ϕ for over-parameterized neural networks and is called *neural tangent kernel* (Jacot et al., 2018).

We note that a kernel smoothing (3) is characterized as a special case of (2) through the following relationship. Let $(\mathcal{H}_k, \langle,\rangle_{\mathcal{H}_k})$ be the reproducing Hilbert space associated with a kernel k. Then,

$$\frac{T_k \nabla_\phi \mathcal{R}_n(\phi)}{\|T_k \nabla_\phi \mathcal{R}_n(\phi)\|_{\mathcal{H}_k^d}} \in \operatorname*{argmax}_{\|\xi\|_{\mathcal{H}_k^d} \le 1} \langle \nabla_\phi \mathcal{R}_n(\phi), \xi \rangle_{L_2^d(\nu_n^X)}, \tag{4}$$

where $\|\xi\|_{\mathcal{H}_k^d}^2 = \sum_{\alpha=1}^d \|\xi^\alpha\|_{\mathcal{H}_k}^2$ for $\xi = (\xi^\alpha)_{\alpha=1}^d \in \mathcal{H}_k^d$, which is confirmed by $\langle \psi, \xi \rangle_{L_2^d(\nu_n^X)} = \langle T_k\psi, \xi\rangle_{\mathcal{H}_k^d}$ for $\forall \xi \in \mathcal{H}_k^d$. Hence, these methods using kernel smoothing are considered as variants of functional gradient boosting.

To further improve the optimization ability compared to (4), one interesting idea is to maximize the following quantity, which corresponds to the right hand side of (4), with respect to k:

$$\langle \nabla_\phi \mathcal{R}_n(\phi), T_k \nabla_\phi \mathcal{R}_n(\phi)\rangle_{L_2^d(\nu_n^X)} = \|T_k \nabla_\phi \mathcal{R}_n(\phi)\|_{\mathcal{H}_k^d}^2. \tag{5}$$

This approach was proposed in Nitanda and Suzuki (2018a) for learning residual networks by successively stacking layers, although there is a less explicit explanation about this viewpoint. In this paper, we also adopt this idea with several modifications for guaranteeing the global convergence property.

4 New Functional Gradient Boosting

We here propose a new functional gradient boosting for learning our models. The proposed method is essentially a combination of kernel learning for improving the approximation ability of kernel smoothing to functional gradients and the Frank-Wolfe method which is a basic scheme of boosting methods wherein weak learners are successively added into a prediction function. Usually, the Frank-Wolfe method is used for learning shallow neural networks (Barron, 1993; Bach, 2014; Ping et al., 2016), but we adapt it for learning deep residual networks by regarding residual blocks as weak learners. Since, in our method, residual blocks are composed of kernel smoothing of functional gradients and kernel functions are composition of a feature extractor and small networks, a modified Frank-Wolfe method makes a feature extractor one level deeper by adding this resblock. As a result, a network gradually grows as the algorithm proceeds, and finally, a deep residual network is obtained.

4.1 Functional gradient boosting for learning deep residual networks

We herein describe the detail of the proposed method. We note that ResFGB (Nitanda and Suzuki, 2018a) is a special case of our method if a learning rate strategy is ignored and $\iota_1 = 0$. In our method, kernel functions used for kernel smoothing for resblocks are obtained by inner products of embedding maps. Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be sets of such embedding maps[2]: $\mathbb{R}^d \to \mathbb{R}^{D'}$ (D' may be set to ∞), and suppose $0 \in \mathcal{K}_i$ ($i \in \{1,2\}$). In our theory, the sets $\mathcal{K}_1$ and $\mathcal{K}_2$ are sufficiently restrictive to guarantee global convergence of the proposed method and the generalization ability of an obtained residual network, which is described later.

[2] $\mathcal{K}_i$ corresponds to outputs of intermediate layer of $\mathcal{B}_i$

For a current feature extractor ϕ_t, a kernel functions used for the smoothing of functional gradients is defined in the following manner: for $e_1 \in \mathcal{K}_1, e_2 \in \mathcal{K}_2$,

$$k_{e_1}(x,x') = e_1(x)^\top e_1(x'),$$
$$k_{e_2}(x,x') = e_2(\phi_t(x))^\top e_2(\phi_t(x')).$$

As explained earlier, a kernel function is optimized to well approximate the functional gradient:

$$\max_{e_1 \in \mathcal{K}_1} \|T_{k_{e_1}} \nabla_\phi \mathcal{R}_n(\phi_t)\|_{\mathcal{H}_{k_{e_1}}^d}^2$$
$$\max_{e_2 \in \mathcal{K}_2} \|T_{k_{e_2}} \nabla_\phi \mathcal{R}_n(\phi_t)\|_{\mathcal{H}_{k_{e_2}}^d}^2. \tag{6}$$

Instead of (6), solving the following problem is practically better: for $D' = d$,

$$\min_{(e_1,e_2)\in\mathcal{K}_1\times\mathcal{K}_2} \left\| e_1 + e_2\circ\phi_t - \frac{\nabla_\phi \mathcal{R}_n(\phi_t)}{\|\nabla_\phi \mathcal{R}_n(\phi_t)(\cdot)\|_2} \right\|_{L_2^d(\nu_n^X)} \tag{7}$$

because this problem (7) is much easier to solve and $\nabla_\phi \mathcal{R}_n(\phi_t)/\|\nabla_\phi \mathcal{R}_n(\phi_t)(\cdot)\|_2$ is a good solution of (6) if it is contained in a set: $\mathcal{K}_1 + \mathcal{K}_2 \circ \phi_t$ as shown in Nitanda and Suzuki (2018a). However, we note that this approximation is just a heuristics and there is a room for improvement by exploring another approximation. We denote by $\overline{e}_1, \overline{e}_2$ approximate solutions of these problems.

Using kernels $k_{\overline{e}_i}$, a residual block that is a weak learner to be added into a feature extractor ϕ_t in the proposed method is constructed as follows:

$$\overline{\iota}_t = \underbrace{-\frac{T_{k_{\overline{e}_1}} \nabla_\phi \mathcal{R}_n(\phi_t)}{Z_1}}_{\iota_1} \underbrace{-\frac{T_{k_{\overline{e}_2}} \nabla_\phi \mathcal{R}_n(\phi_t)}{Z_1}}_{\iota_2\circ\phi_t}, \tag{8}$$

where Z_1 is the normalization term: $Z_1 = \|T_{k_{\overline{e}_1}} \nabla_\phi \mathcal{R}_n(\phi_t)\|_{\mathcal{H}_{k_{\overline{e}_1}}^d}$. We note that the components of a resblock, ι_1 and ι_2, are also defined in the above equation and are composed of neural networks:

$$\iota_1 = A_1 \overline{e}_1(\cdot),\ \iota_2\circ\phi_t = A_2\overline{e}_2\circ\phi_t(\cdot),$$

where A_1, A_2 are (d,d)-matrices:

$$\begin{cases} A_1 &= -\frac{1}{nZ_1}\sum_{i=1}^n \nabla_\phi \mathcal{R}_n(\phi_t)(x_i)\overline{e}_1^\top(x_i), \\ A_2 &= -\frac{1}{nZ_1}\sum_{i=1}^n \nabla_\phi \mathcal{R}_n(\phi_t)(x_i)(\overline{e}_2\circ\phi_t)^\top(x_i). \end{cases} \tag{9}$$

Finally, a Frank-Wolfe-like update is performed as follows: for a given learning rate η_t,

$$\phi_{t+1} \leftarrow (1-\eta_t)\phi_t + \eta_t \overline{\iota}_t.$$

As the right hand side of the above iteration is equal to $((1-\eta_t)id + \eta_t\iota_2)\circ\phi_t(x) + \eta_t\iota_1(x)$, we notice that ϕ_{t+1} is one level deeper than ϕ_t and a deep residual network is constructed.

We give a full description of the proposed method called *ResFGB-FW* in Algorithm 1. Although, we made the assumption where a weight $\overline{w}$ for the last layer is pre-trained due to theoretical tractability, it may be practically better to optimize $\overline{w}$ at each iteration, so that Algorithm 1 contains it as an option. We here note that this procedure is certainly similar to a progressive learning scheme in Belilovsky et al. (2019), except for details of the architecture of each layer, which exhibits comparable high performance with end-to-end training schemes on the image classification tasks.

Algorithm 1 ResFGB-FW

```
Input: $S=(x_i,y_i)_{i=1}^n$, the number of iterations $T$, and learning rates $\eta_t$
$\phi_0 \leftarrow id,\ \overline{w} \leftarrow \mathrm{argmin}_{w\in\mathbb{R}^{d\times c},\|(w)_{*.y}\|_1\le\Lambda_w} \mathcal{L}_n(w^\top\phi_0)$

for $t=0$ to $T-1$ do
    (Option) $\overline{w} \leftarrow \mathrm{argmin}_{w\in\mathbb{R}^{d\times c},\|(w)_{*.y}\|_1\le\Lambda_w} \mathcal{L}_n(w^\top\phi_t)$

    Get $\overline{e}_i$ by solving (6) or (7) on $S$
    $A_1 \leftarrow -\frac{1}{nZ_1}\sum_{i=1}^n \nabla_\phi\mathcal{R}_n(\phi_t)(x_i)\overline{e}_1^\top(x_i)$
    $A_2 \leftarrow -\frac{1}{nZ_1}\sum_{i=1}^n \nabla_\phi\mathcal{R}_n(\phi_t)(x_i)(\overline{e}_2\circ\phi_t)^\top(x_i)$
    $\overline{\iota}_t \leftarrow A_1\overline{e}_1(\cdot) + A_2\overline{e}_2\circ\phi_t(\cdot)$
    $\phi_{t+1} \leftarrow (1-\eta_t)\phi_t + \eta_t\overline{\iota}_t$
end for
Return $\overline{w}^\top\phi_T$
```

Remark There are two major differences from the method in Nitanda and Suzuki (2018a); one is that the output $\phi_t(x)$ is reduced at the rate of $1-\eta_t$ stemming from the Frank-Wolfe method and the other is that resblocks receive input data directly like DenseNet (Huang et al., 2017). These modifications lead to provable statistical guarantees without weak learning assumptions adopted in Nitanda and Suzuki (2018a).

5 Convergence Analyses

We here provide the convergence analysis of Algorithm 1. To do so, we have to specify the architecture of $\mathcal{K}_i$ and provide the required condition on resblocks obtained in Algorithm 1 to guarantee the global convergence property. We assume $\mathcal{K}_i$ are sets of shallow neural networks: $F_i\sigma(G_ix)$ with norm constraints. Note that the parameters of resblocks introduced in Section 2 can be obtained by

$$C_i = A_iF_i,\ D_i = G_i. \tag{10}$$

The set of learnable functions by our method is described as follows. We denote by $D_{\mathcal{K}_1}$ the union of unit balls in RKHSs defined by $e_1\in\mathcal{K}_1$:

$$D_{\mathcal{K}_1} \overset{def}{=} \cup_{e_1\in\mathcal{K}_1}\{\xi\in\mathcal{H}^d_{k_{e_1}} \mid \|\xi\|_{\mathcal{H}^d_{k_{e_1}}}\le 1\}.$$

Then, we define $\mathrm{conv}(D_{\mathcal{K}_1})$ as the convex combination of $D_{\mathcal{K}_1}$. We here make the following assumptions for theoretical guarantees.

Assumption 2.

(A1) *Let Λ_i^F, Λ_i^G $(i\in\{1,2\})$ be positive values and σ be a 1-Lipschitz activation function. Let $\mathcal{K}_1,\mathcal{K}_2$ be sets of shallow neural networks:*

$$\mathcal{K}_i = \{z\mapsto F_i\sigma(G_iz) \mid \sum_j \|(F_i)_{*,j}\|_2 \le \Lambda_i^F,$$
$$\max_k \|(G_i)_{k,*}\|_1 \le \Lambda_i^G\}.$$

In addition, $\|e_i(z)\|_2\le\sqrt{K}$ $(i\in\{1,2\},\ K>0)$ over $z\in\mathbb{R}^d$ and the maximum singular value of $\overline{w}$ is upper bounded by $\Sigma_{\overline{w}}$.

(A2) *Resblocks: $\iota_1 = A_1\overline{e}_1$ and $\overline{\iota}_t = A_1\overline{e}_1 + A_2\overline{e}_2\circ\phi_t$ satisfy the following:*

$$-\langle\nabla_\phi\mathcal{R}_n(\phi_t),\iota_1\rangle_{L_2^d(\nu_n^X)} \ge \frac{1}{2}\max_{\xi\in D_{\mathcal{K}_1}}\langle\nabla_\phi\mathcal{R}_n(\phi),\xi\rangle_{L_2^d(\nu_n^X)},$$
$$-\langle\nabla_\phi\mathcal{R}_n(\phi_t),\overline{\iota}_t\rangle_{L_2^d(\nu_n^X)} \ge \max_{\xi\in D_{\mathcal{K}_1}}\langle\nabla_\phi\mathcal{R}_n(\phi),\xi\rangle_{L_2^d(\nu_n^X)}.$$

(A3) *For a training set $S=(x_i,y_i)_{i=1}^n$ sampled from ν^n, there exist $M_n,\gamma_n>0$ and $\psi_n\in\mathrm{conv}(D_{\mathcal{K}_1})$ such that for $v_{i,y}\overset{def}{=}\overline{w}_{*,y}^\top\psi_n(x_i)$, $\forall i\in\{1,\dots,n\}$,*

$$v_{i,y_i}\ge M_n+\gamma_n,\ v_{i,y}\le M_n\ (\forall y\ne y_i).$$

Remark Note that since the first component of resblock: $-\iota_1 = -A_1\overline{e}_1$ maximizes $\langle\nabla_\phi\mathcal{R}_n(\phi),\xi\rangle_{L_2^d(\nu_n^X)}$ through equations (4) and (5) over $\xi\in D_{\mathcal{K}_1}$, a kernel $\overline{\iota}_t$ always satisfies the condition **(A2)** if problems (6) are solved exactly, but **(A2)** also allows approximate solution thanks to a support by ι_2. In other words, a deep architecture alleviates the requirement of the optimality of weak learners for subproblems and is certainly useful in enhancing the global convergence. The condition **(A3)** means that a given training set is separable by $\mathrm{conv}(D_{\mathcal{K}_1})$ with a margin γ_n. We note that this assumption is reasonable because $\mathrm{conv}(D_{\mathcal{K}_1})$ is essentially a set of infinite-width shallow neural networks and has a universal approximation property. Hence, a margin γ_n actually exists as long as a data distribution is separable by a continuous function.

To ensure the convergence and generalization properties, we should derive upper bounds on norms of parameters of resblock $\bar{\iota}$ and norm $\|\bar{\iota}(z)\|_2$, but it is not obvious because resblocks contain a normalization term Z_1. An interesting technical contribution is to derive the lower bound on Z_1 under Assumption 2.

Proposition 2. *Suppose Assumption 2 holds. Then, iterates of Algorithm 1 satisfy*

$$\frac{\gamma_n}{4\Sigma_{\overline{w}}}\|\nabla_\phi \mathcal{R}_n(\phi_t)\|_{L_1^d(\nu_n^X)} \leq \|T_{k_{\overline{e}_1}}\nabla\mathcal{R}_n(\phi_t)\|_{\mathcal{H}^d_{k_{\overline{e}_1}}}. \quad (11)$$

Proposition 2 says that the kernel smoothing in Algorithm 1 produces a sufficient reduction of objective in a function space as the margin γ_n is large. As a result, norms of parameters and outputs of resblocks can be made small.

Proposition 3. *Suppose Assumption 2 holds. Then, it follows that for resblocks $\iota_i = A_iF_i\sigma(G_i\cdot)$ obtained in Algorithm 1,*

$$\|(A_iF_i)_{\alpha,*}\|_1 \leq \frac{4}{\gamma_n}\Sigma_{\overline{w}}\Lambda_i^F\sqrt{K}, \quad (12)$$

$$\|\iota_1\|_{L_2^d(\nu_n^X)},\ \|\iota_2\circ\phi\|_{L_2^d(\nu_n^X)} \leq \frac{4}{\gamma_n}\Sigma_{\overline{w}}K. \quad (13)$$

This proposition is useful for both convergence and generalization guarantees, namely, the equation (12) is used for deriving a generalization bound by substituting it into the bound in Theorem 1, and the equation (13) is used for deriving a convergence rate.

Global Convergence. The next theorem shows the convergence property of Algorithm 1 to a better model than $\mathrm{conv}(D_{\mathcal{K}_1})$, which can be shown by adapting a proof for the vanilla Frank-Wolfe method (Jaggi, 2013) to our method. For simplicity, we set $\mathcal{R}_n^* \overset{def}{=} \inf_{\phi\in\mathrm{conv}(D_{\mathcal{K}_1})}\mathcal{R}_n(\phi)$ and set for $t_0, \Lambda_\infty > 0$,

$$C \overset{def}{=} \max\left\{t_0(\mathcal{R}_n(\phi_0)-\mathcal{R}_n^*), 2L\left(\Lambda_\infty + \frac{16\Sigma_{\overline{w}}K}{\gamma_n}\right)^2\right\}.$$

Theorem 2 (Convergence rate)**.** *Suppose Assumption 2 holds and $\|x\|_2 \leq \Lambda_\infty$ on $\mathcal{X}$. Then, iterates of Algorithm 1 with learning rates $\eta_t = \frac{2}{t+t_0}$ $(t_0 \geq 0)$ satisfy*

$$\mathcal{R}_n(\phi_T) - \mathcal{R}_n^* \leq \frac{C}{T+t_0}.$$

Note that this theorem provides a global convergence guarantee. An important point of our analysis is that a type of weak learning condition is not required unlike existing studies (Huang et al., 2018; Nitanda and Suzuki, 2018a). In addition, we remark that the convergence becomes faster when the input distribution has a high degree of linear separability and when using a sufficient large t_0.

Generalization Bound. By combining Proposition 2, 3, and Theorem 2 with generalization bound in Theorem 1, we obtain the main result in this study. To describe the result, we define

$$\epsilon_n \overset{def}{=} (c-1)\exp(-\gamma_n).$$

We can easily see that ϵ_n is an upper bound on $\mathcal{R}_n^*$ as follows. We set $f = \overline{w}^\top\psi_n$ where $\psi_n \in \mathrm{conv}(D_{\mathcal{K}_1})$ is defined in Assumption 2-**(A3)**. Under Assumption 2-**(A3)**, for $i \in \{1,\ldots,n\}$,

$$\begin{aligned} l(f(x_i),y_i) &= \log(1+\textstyle\sum_{y'\neq y_i}\exp(f_{y'}(x_i)-f_{y_i}(x_i))) \\ &\leq \log(1+(c-1)\exp(-\gamma_n)) \leq \epsilon_n. \end{aligned}$$

Thus, taking an expectation with respect to ν_n^X, we see that $\mathcal{R}_n^* \leq \mathcal{R}_n(\psi_n) \leq \epsilon_n$. Moreover, we note that $\epsilon_n \to 0$ as $\gamma_n \to \infty$.

Corollary 1 (Main result)**.** *Suppose Assumption 2 holds and $\|x\|_2 \leq \Lambda_\infty$ on $\mathcal{X}$. Assume $\Lambda_2^F\Lambda_2^G \leq \gamma_n/8\Sigma_{\overline{w}}\sqrt{K}$. Fix $\delta > 0$. Then, for $\forall\rho > 0$, with probability at least $1-\rho$ over random choice from ν^n, a predictor f_T obtained by Algorithm 1 with $\eta_t = \frac{2}{t+t_0}$ $(t_0 \geq 2)$ and T-iterates satisfies the following bound.*

$$\mathbb{P}_\nu[m_f(X,Y)\leq 0] \leq (1+\exp(\delta))\left(\epsilon_n + \frac{C}{T+t_0}\right)$$
$$\frac{2c^3\Lambda_w\Lambda_\infty}{\delta\sqrt{n}}\left(1+\frac{8\Sigma_{\overline{w}}\Lambda_1^F\Lambda_1^G\sqrt{K}}{\gamma_n}\right) + \sqrt{\frac{1}{2n}\log\rho^{-1}}.$$

From Corollary 1, we can see that a worse dependence on the network depth disappears; hence, the best generalization performance is achieved when $T \to \infty$. Specifically, if $\delta = \gamma_n/2$ and γ_n are uniformly positive, then we get for a sufficiently large T,

$$\mathbb{P}_\nu[m_f(X,Y)\leq 0] \leq O\left(\epsilon_n + \frac{1}{\gamma_n\sqrt{n}} + \sqrt{\frac{1}{n}\log\rho^{-1}}\right).$$

From this bound, we can see that the standard margin γ_n helps the generalization ability of the method, that is, the complexity term is considerably reduced when the margin γ_n is sufficiently large.

6 Numerical Experiments

We present experimental results on the binary and multiclass classification tasks. Our implementation is done using Theano (Theano Development Team, 2016). The experiments are conducted on Intel Core i7-8700K and TITAN Xp 12GB with Ubuntu 16.04 (64-bit). We compare Algorithm 1 with ResFGB (Nitanda and Suzuki, 2018a), multilayer perceptron (MLP), support vector machine (SVM), random forest (RF), and gradient boosting methods (GBM).

Table 1: Test classification accuracies on benchmark datasets.

Method	letter	usps	ijcnn1	mnist	covtype
ResFGB-FW	**0.981 (0.0007)**	**0.954 (0.0011)**	0.988 (0.0008)	**0.988 (0.0004)**	0.969 (0.0008)
ResFGB	0.976 (0.0019)	**0.954 (0.0009)**	**0.989 (0.0004)**	0.986 (0.0007)	0.965 (0.0007)
MLP	0.970 (0.0060)	0.949 (0.0040)	0.988 (0.0011)	0.986 (0.0010)	0.965 (0.0015)
SVM	0.961 (0.0060)	0.947 (0.0020)	0.979 (0.0019)	0.970 (0.0040)	0.830 (0.0061)
RF	0.966 (0.0014)	0.940 (0.0020)	0.980 (0.0006)	0.973 (0.0005)	0.950 (0.0010)
GBM	0.967 (0.0013)	0.939 (0.0040)	0.982 (0.0011)	0.982 (0.0007)	**0.971 (0.0005)**

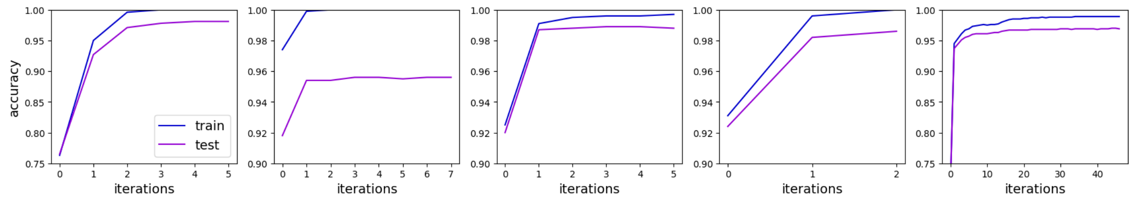

Figure 2: Learning curves for Algorithm 1 on multiclass classification problems. The figure shows classification accuracy on training (blue) and test (purple) sets versus the number of iterations.

We use the following datasets: letter, usps, ijcnn1, mnist, and covtype. We first divide each train dataset randomly into two sets: 80% for training and the rest for validation. We run each method on the train dataset with several hyperparameter settings and choose a setting providing the best classification accuracy on the validation dataset. We finally train each model on an entire train dataset and report a classification accuracy on the test dataset. We perform the above procedure 5-times.

We explain hyper-parameter settings. For Algorithm 1, three hidden-layers neural networks with 1000-nodes per layer are used as $e_i \in \mathcal{K}_i$ ($i \in \{1,2\}$) to define resblocks. Linear classifiers and e_i are trained by Nesterov's momentum method at each loop. The number of iterations T and a parameter $t_0 \in \{10, 100, 1000\}$ for learning rates $\eta_t = 2/(t+t_0)$ are tuned based on the performance on the validation set. For ResFGB (Nitanda and Suzuki, 2018a), constant learning rates are chosen from $\{10^{-3}, 10^{-2}, 10^{-1}\}$ and similar settings as Algorithm 1 are adopted for the other hyper-parameters. For MLP, the ReLU is used as an activation and network depth is set to three, four, or five layers. The number of hidden units per layer is 100 or 1000. For SVM, a random Fourier feature (Rahimi and Recht, 2007) is adopted with an embedding dimension of 10^3 or 10^4. For RF, the maximum depth is set to 10, 20, or 30 and the number of trees is set to 100, 500, or 1000. For GBM, the learning rate is chosen from $\{10^{-3}, 10^{-2}, 10^{-1}, 1\}$ with the maximum number of trees being set to 1000, and number of leaves in one tree is chosen from $\{2^4, 2^5, \dots, 2^{10}\}$.

Table 1 shows the mean classification accuracies and standard deviations and Figure 2 depicts learning curves on training and test datasets. As seen in the table, the proposed method shows superior or comparable performance over other state-of-the-art methods. Therefore, we can conclude that our proposed method is certainly useful for training deep ResNets with a provable statistical guarantee.

7 Conclusion

We have proposed a new functional gradient boosting by combining ResFGB with the Frank-Wolfe method for learning deep ResNets. In theoretical analyses, we have provided a generalization bound via a global convergence analysis of the method under a margin condition instead of a weak learning condition. Compared to the previous study, our bound has two important points; one is the elimination of a worse dependence on the network depth, and the other is a significant improvement by the existence of a learnable classifier with a large margin on train datasets. Finally, we have demonstrated superior empirical performance of the proposed method over the state-of-the-art methods. An interesting future work is to adjust our method to the network architecture search by combining with a usual end-to-end training such as stochastic gradient descent method. To do so, it will be necessary to develop an effective criterion to switch the functional gradient boosting and the end-to-end training schemes.

Acknowledgement AN was partially supported by JSPS Kakenhi (19K20337) and JST-PRESTO. TS was partially supported by JSPS Kakenhi (26280009, 15H05707 and 18H03201), Japan Digital Design and JST-CREST.

References

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. (2019). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*.

Bach, F. (2014). Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(19):1–53.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945.

Belilovsky, E., Eickenberg, M., and Oyallon, E. (2019). Greedy layerwise learning can scale to imagenet. In *International Conference on Machine Learning*, pages 583–593.

Brock, A., Lim, T., Ritchie, J. M., and Weston, N. (2017). Freezeout: Accelerate training by progressively freezing layers. *arXiv preprint arXiv:1706.04983*.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.

Cortes, C., Gonzalvo, X., Kuznetsov, V., Mohri, M., and Yang, S. (2017). Adanet: Adaptive structural learning of artificial neural networks. In *International Conference on Machine Learning 34*, pages 874–883.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. (2018). Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*.

Frank, M. and Wolfe, P. (1956). An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, pages 1189–1232.

Guélat, J. and Marcotte, P. (1986). Some comments on wolfe's 'away step'. *Mathematical Programming*, 35(1):110–119.

Han, S., Meng, Z., Khan, A.-S., and Tong, Y. (2016). Incremental boosting convolutional neural network for facial action unit recognition. In *Advances in neural information processing systems 29*, pages 109–117.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Huang, F., Ash, J., Langford, J., and Schapire, R. (2018). Learning deep resnet blocks sequentially using boosting theory. In *International Conference on Machine Learning 35*, pages 2058–2067.

Huang, G., Liu, Z., Weinberger, K. Q., and van der Maaten, L. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700—4708.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31*, pages 8580–8589.

Jaggi, M. (2013). Revisiting frank-wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning 30*, pages 427–435.

Jastrzebski, S., Arpit, D., Ballas, N., Verma, V., Che, T., and Bengio, Y. (2017). Residual connections encourage iterative inference. *arXiv preprint arXiv:1710.04773*.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30*, pages 3149–3157.

Kim, J., Lee, S., Kim, S., Cha, M., Lee, J. K., Choi, Y., Choi, Y., Cho, D.-Y., and Kim, J. (2018). Autometa: Automated gradient based meta learner search. *arXiv preprint arXiv:1806.06927*.

Kivinen, J., Smola, A. J., and Williamson, R. C. (2004). Online learning with kernels. *IEEE transactions on signal processing*, 52(8):2165–2176.

Koltchinskii, V. and Panchenko, D. (2002). Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30(1):1–50.

Kulkarni, M. and Karande, S. (2017). Layer-wise training of deep networks using kernel similarity. *arXiv preprint arXiv:1703.07115*.

Littwin, E. and Wolf, L. (2016). The loss surface of residual networks: Ensembles and the role of batch normalization. *arXiv preprint arXiv:1611.02525*.

Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.-J., Fei-Fei, L., Yuille, A., Huang, J., and Murphy, K. (2018). Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34.

Malach, E. and Shalev-Shwartz, S. (2018). A provably correct algorithm for deep learning that actually works. *arXiv preprint arXiv:1803.09522*.

Marquez, E. S., Hare, J. S., and Niranjan, M. (2018). Deep cascade learning. *IEEE transactions on neural networks and learning systems*, 29(11):5475–5485.

Mason, L., Baxter, J., Bartlett, P. L., and Frean, M. R. (1999). Boosting algorithms as gradient descent. In *Advances in Neural Information Processing Systems 12*, pages 512–518.

Mosca, A. and Magoulas, G. D. (2017). Deep incremental boosting. *arXiv preprint arXiv:1708.03704*.

Neyshabur, B., Tomioka, R., and Srebro, N. (2015). Norm-based capacity control in neural networks. In *Proceedings of Conference on Learning Theory 28*, pages 1376–1401.

Nitanda, A. and Suzuki, T. (2018a). Functional gradient boosting based on residual network perception. In *International Conference on Machine Learning 35*, pages 3819–3828.

Nitanda, A. and Suzuki, T. (2018b). Gradient layer: Enhancing the convergence of adversarial training for generative models. In *Proceedings of International Conference on Artificial Intelligence and Statistics 21*, pages 1008–1016.

Nøkland, A. and Eidnes, L. H. (2019). Training neural networks with local error signals. In *International Conference on Machine Learning 36*, pages 4839–4850.

Opitz, M., Waltner, G., Possegger, H., and Bischof, H. (2017). Bier-boosting independent embeddings robustly. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5189–5198.

Ping, W., Liu, Q., and Ihler, A. T. (2016). Learning infinite rbms with frank-wolfe. In *Advances in Neural Information Processing Systems 29*, pages 3063–3071.

Rahimi, A. and Recht, B. (2007). Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20*, pages 1177–1184.

Raskutti, G., Wainwright, M. J., and Yu, B. (2014). Early stopping and non-parametric regression: an optimal data-dependent stopping rule. *Journal of Machine Learning Research*, 15(1):335–366.

Smale, S. and Yao, Y. (2006). Online learning algorithms. *Foundations of computational mathematics*, 6(2):145–170.

Theano Development Team (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv preprint arXiv:1605.02688*.

Veit, A., Wilber, M. J., and Belongie, S. (2016). Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems 29*, pages 550–558.

Wang, G., Xie, X., Lai, J., and Zhuo, J. (2017). Deep growing learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2812–2820.

Wei, Y., Yang, F., and Wainwright, M. J. (2017). Early stopping for kernel boosting algorithms: A general analysis with localized complexities. In *Advances in Neural Information Processing Systems 30*, pages 6065–6075.

Weill, C., Gonzalvo, J., Kuznetsov, V., Yang, S., Yak, S., Mazzawi, H., Hotaj, E., Jerfel, G., Macko, V., Adlam, B., et al. (2019). Adanet: A scalable and flexible framework for automatically learning ensembles. *arXiv preprint arXiv:1905.00080*.

Ying, Y. and Zhou, D.-X. (2006). Online regularized classification algorithms. *IEEE Transactions on Information Theory*, 52(11):4775–4788.