Budget Learning via Bracketing

Aditya Gangrade
Boston University
gangrade@bu.edu

Durmus Alp Emre Acar
Boston University
alpacar@bu.edu

Venkatesh Saligrama
Boston University
srv@bu.edu

Abstract

Conventional machine learning applications in the mobile/IoT setting transmit data to a cloud-server for predictions. Due to cost considerations (power, latency, monetary), it is desirable to minimise device-to-server transmissions. The budget learning (BL) problem poses the learner's goal as minimising use of the cloud while suffering no discernible loss in accuracy, under the constraint that the methods employed be edge-implementable.

We propose a new formulation for the BL problem via the concept of bracketings. Concretely, we propose to sandwich the cloud's prediction, g, via functions h^-, h^+ from a 'simple' class so that $h^- \leq g \leq h^+$ nearly always. On an instance x, if $h^+(x) = h^-(x)$, we leverage local processing, and bypass the cloud. We explore theoretical aspects of this formulation, providing PAC-style learnability definitions; associating the notion of budget learnability to approximability via brackets; and giving VC-theoretic analyses of their properties. We empirically validate our theory on real-world datasets, demonstrating improved performance over prior gating based methods.

Edge devices in mobile and IoT applications are battery and processing power limited. This imposes severe constraints on the methods implementable in such settings - for instance, the typical CPU-based structure of such devices precludes the use of many convolutional layers in vision tasks due to computational latency (Zhou et al. 2019), imposing architectural constraints. In particular, modern high accuracy methods like deep neural networks are seldom implementable in these settings. At the same time, edge devices are required to give fast and accurate decisions. Enabling such mechanisms is an important technical challenge.

Typically, practitioners either learn weak models that can be implemented on the edge (e.g. Hinton et al. 2015; Kumar et al. 2017; Wu et al. 2019), which suffer more errors, or they learn a complex model, which is implemented in a cloud[1]. The latter solution is also not ideal - cloud access must be purchased, the prediction pipeline suffers from communication latency, and, since communication consumes the majority of the battery power of such a device (Zhu et al. 2019), such solutions limit the device's operational lifetime (see also industry articles, e.g. Hollemans 2017; Norman 2019). A third option, largely unexplored in practice, is a hybrid of these strategies - we may learn mechanisms to filter out 'easy' instances, which may be classified at the edge, and send 'difficult' instances to the cloud. The reduction in cloud usage provides direct benefits in, e.g., battery life, yet accuracy may be retained. Similar concerns apply in many contexts, e.g. in medicine, security, and web-search (Nan and Saligrama 2017a; Xu et al. 2014).

The key challenge in these applications is to maintain a high accuracy while keeping the usage of the complex classifier, i.e. the budget, low. To keep accuracy high, we enforce that on the locally predicted instances, the prediction *nearly always agrees* with the cloud. This is thus a problem of 'bottom-up' budget learning (BL).

The natural approach to BL is via the 'gating formulation': one learns a gating function γ, and a local predictor π, such that if $\gamma = 1$ then π is queried, and otherwise the cloud is queried. Unfortunately, this setup is computationally difficult, since the overall classifier involves the product $\pi \cdot \gamma$, and optimising over the induced non-covexity is hard. Previous efforts try to meet this head on, but either yield inefficient methods, or require difficult to justify relaxations.

Our main contribution is a novel formulation of the BL problem, via the notion of brackets, that sidesteps

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

[1] or, more realistically, purchase access to a cloud-based model owned by a company that has sufficient data and computational power (e.g. CoreML 2019; ML Kit 2019).

this issue. For functions $h^- \leq h^+$, the bracket $[h^-, h^+] := \{f : h^- \leq f \leq h^+\}$. Brackets provide accurate pointwise control on a binary function - for $f \in [h^-, h^+]$, if $h^+(x) = h^-(x)$, then $f(x)$ takes the same value. We propose to learn a bracketing of the cloud, predicting locally when this condition holds.

The key advantage of this method arises from the surprising property that we may learn optimal brackets via two *decoupled* learning problems - separately approximating the function from above and from below. These one-sided problems are tractable under convex surrogates, with minimal statistical compromises. Further, this comes at negligible loss of expressivity compared to gating - the existence of good gates and predictors implies the existence of equally good brackets.

Since expressivity is retained, bracketings lead naturally to definitions of learnability that are theoretically analysable. We define a PAC-style approach to one-sided learning, and provide a VC-theoretic characterisation of the same. We also identify the key budget learning problem as an approximation theoretic question - *which complex classes have 'good' bracketings by simple classes?* We characterise this for a binary version of Hölder smooth classes, and also provide partial results for generic classes with bounded VC dimension.

Finally, to validate the formulation, we implement the bracketing framework on a binary versions of MNIST and CIFAR classification tasks. With a strong disparity in the cloud and edge models (§4), we obtain usages of $20-40\%$ at accuracies higher than 98% with respect to the cloud. Further, we outperform existing methods in usage by factors of $1.2-1.4$ at these high accuracies.

Related Work

A common approach is to simply learn local classifiers with no cloud usage. If the cloud model is available, one can use methods such as distillation (Hinton et al. 2015), and in general one can train classifiers in a resource aware way (e.g. Gupta et al. 2017; Kumar et al. 2017; Wu et al. 2019). The main limitation here of this approach is that if the setting is complex enough for a cloud to be needed, then in general such methods cannot attain a similar accuracy level.

Top-Down and Sequential Approaches. are based on successively learning classifiers of increasing complexity, incorporating the previously learned classifiers (see Bolukbasi et al. 2017; Nan, Wang, et al. 2016; Trapeznikov and Saligrama 2013; Wang et al. 2015; Xu et al. 2014). This approach suffers a combinatorial explosion in the complexity of the learning problems. Recent efforts utilise reinforcement learning methods to rectify this (e.g. Janisch et al. 2019a,b; Peng et al. 2018).

The BL problem is intimately related to *learning with abstention* (LwA). Indeed, sending an example to the cloud is the same as abstaining on it. The twist in BL is twofold - we assume that a noiseless ground truth, i.e., the 'cloud classifier' exists, while LwA tends to concentrate on settings where the labels are noisy; and the class of locally implementable models is much weaker than the class known to contain the cloud model, while the LwA literature is generally not concerned with 'simple' classifiers. In addition, no theoretical work on LwA captures this setting. Perhaps the closest is the study of 'perfect selective classification' by Wiener and El-Yaniv 2011; El-Yaniv and Wiener 2010, but this work focuses on the stringent condition of getting perfect agreement with certainty, and only gives analyses for classes with controlled disagreement coefficients.

Plug-in methods utilise a pre-trained low complexity model, and learn a gate by estimating its low-confidence regions. We note that much of the theoretical analysis for LwA concentrates on such methods, e.g. Bartlett and Wegkamp 2008; Denis and Hebiri 2019; Herbei and Wegkamp 2006; Shekhar et al. 2019.[2] The principal disadvantage here is that these classifiers are not tuned to the BL problem. However, even crude methods such as gating by thresholding the softmax response of a classifier are very effective (see §4), and serve as strong baselines as observed by Geifman and El-Yaniv 2017, 2019 in the setting of deep neural networks.

A number of methods aim at *jointly learning gating and prediction* functions (c.f. §1.4). Some of these belong to the LwA literature - Geifman and El-Yaniv 2019 propose to ignore the non-convexity, and use SGD to optimise a loss of the form $\widehat{\mu}(\pi \neq g|\gamma = 1)$ subject to a budget constraint, while Cortes et al. 2016 instead propose the relaxation $\pi\gamma \leq (\pi + \gamma)/2$, and optimise this upper bound via convex relaxations. In the BL literature, Nan and Saligrama 2017a,b relax the problem by introducing an auxiliary variable to decouple π and γ, and then perform alternating minimisation with a KL penalty between the gate and the auxiliary. Note that while each of these papers further specifies algorithms to train classifiers, their main conceptual contribution is the method they take to ameliorate the essential non-convexity of the gating setup. In contrast, our new formulation sidesteps this issue entirely.

Our approach to one-sided learning is related to *Neyman-Pearson classification* (Cannon et al. 2002; Scott and Nowak 2005), with the difference that instead of studying the conditional risks, we are concerned with restricting the total risk subject to one-sided constraints. This leads to the generalisation errors of one-sided learning scaling with the total sample size, as opposed to the per-class sample sizes (see §2.1).

[2] Herbei and Wegkamp 2006 also analyse ERM in the setting where a fixed cost for abstention is available.

Bracketings are important in empirical process theory - for instance, 'bracketable' classes characterise the universal Glivenko-Cantelli property (van Handel 2013). While there are generic estimates of the bracketing entropies of various function classes (e.g. Ch2 of van der Vaart and Wellner 1996), these typically do not constrain for complexity of the resulting brackets, and thus their application in our setting is limited. Instead, we explicitly aim to bracket functions by *simple* function classes (see §3.2). We note, however, that our results towards this are preliminary.

1 Definitions and Formulations

We will restrict discussion to binary functions on the domain $\mathcal{X}$, which is assumed to be compact[3]. $\mathcal{H}$ denotes the class of local classifiers, and $\mathcal{G}$ the class of cloud classifiers. We use $g \in \mathcal{G}$ to denote the high-complexity 'cloud' classifier. The training set is taken to be $\{(X_i, g(X_i)\}$, where $X_i \overset{\text{i.i.d.}}{\sim} \mu$, and μ is an unknown probability on $\mathcal{X}$[4]. For feasibility of various programs (particularly Def. 2), we assume that $\{0,1\} \subset \mathcal{H}$, and that $h \in \mathcal{H} \iff 1 - h \in \mathcal{H}$.

The main problem is to learn approximations to g in $\mathcal{H}$, with the option to 'fall back' to g. We aim at retaining high accuracy w.r.t. g while minimising usage of g itself.

1.1 Bracketing for Budget Learning

Definition *Given a measure μ and functions, $h_1 \le h_2$, the bracket $[h_1, h_2]$ is the set of all $\{0,1\}$-valued functions f such that $h_1 \le f \le h_2$ μ-a.s. The μ-size of such a bracket is $|[h_1, h_2]|_\mu := \mu(h_1 \ne h_2)$.*

As an example, on $[0,1]$, the functions $0(x)$ and $\mathbb{1}\{x > 1/2\}$ induce the bracket containing all functions that are 0 on $[0, 1/2]$. This bracket has size $\mu(X > 1/2)$.

Notice that if $h_1 \ne h_2$ in the above, it is forced that $h_1 = 0, h_2 = 1$. We will be concerned with the brackets that can be built using hs from the local class $\mathcal{H}$.

Definition *The set of brackets generated by a class $\mathcal{H}$ is $\{[h_1, h_2] : h_1 \le h_2, h_1, h_2 \in \mathcal{H}\}$. We also say that these are $\mathcal{H}$-brackets.*

Suppose we can find a bracket $[h^-, h^+]$ in $\mathcal{H}$ that contains g. Since $h^- = h^+$ forces g to take the same value, we offer the classifier

$$c_{[h^-,h^+]}(x) = \begin{cases} h^+(x) & \text{if } h^+(x) = h^-(x) \\ g(x) & \text{if } h^+(x) \ne h^-(x) \end{cases}.$$

[3] Issues of measurability, and of existence of minimisers of optimisation problems posed as infima are suppressed, as is common in learning theory.

[4] If instead we have a raw dataset and no g, we assume that g is obtained by training a function in $\mathcal{G}$ over this set.

The above has the *usage* $|[h^-, h^+]|_\mu$. The budget needed by a class $\mathcal{H}$ to bracket (g, μ) is the smallest such usage,

$$\mathsf{B}(g, \mu, \mathcal{H}) := \inf_{\mathcal{H}-\text{brackets}} \{|[h^-, h^+]|_\mu : g \in [h^-, h^+]\}.$$

This extends naturally to bracketing of sets.

Definition 1 *A set of function-measure pairs $\mathcal{S} = \{(g_i, \mu_i)\}$ is bracket-approximable by a class $\mathcal{H}$ if for every (g, μ), there exists a $\mathcal{H}$-bracket containing g. The budget required for bracket approximation of $\mathcal{S}$ by $\mathcal{H}$ is*

$$\mathsf{B}(\mathcal{S}, \mathcal{H}) := \sup_{(g,\mu)\in\mathcal{S}} \mathsf{B}(g, \mu, \mathcal{H}).$$

This is a very weak notion of approximation - all it demands is that for every g, we can find some $\mathcal{H}$-bracket. Typical study of bracketings concentrates on real valued functions, and studies how many brackets, or how large an $\mathcal{H}$, we need to make the loss B smaller than some given value. We defer such explorations to §2.2, where we define a notion of budget learning.

For the following discussion, it is useful to define a relaxed version of brackets.

Definition *Let $\alpha \in [0,1]$, and h_1, h_2 be $\{0,1\}$-valued functions such that $\mu(h_1 \le h_2) \ge 1 - \alpha/2$. The α-approximate bracket $[h_1, h_2]$ with respect to μ is the set of functions f such that $\mu(h_1 \le f \le h_2) \ge 1 - \alpha$. We call $1 - \alpha$ the accuracy of the bracketing.*

The above brackets are approximate in two ways: the order of h_1 and h_2 may be reversed, and the functions in the $[h_1, h_2]$ may leak out from within them.

1.2 One-sided Approximation and Decoupled Optimisation of Brackets

In order to discuss the decoupled optimisation of brackets, we introduce the notion of *one-sided approximation*.

Definition 2 *For a function-measure pair (g, μ), an* approximation from below *to g in a class $\mathcal{H}$ is any minimiser of the following optimisation problem*

$$\mathsf{L}(g, \mu, \mathcal{H}) := \inf\{\mu(h \ne g) : h \in \mathcal{H}, h \le g\}.$$

We refer to L as the inefficiency of approximation from below of (g, μ) by $\mathcal{H}$. We analogously define approximation from above *as $1 - h$, where h is an approximation of $1 - g$ from below.*

We use 'one-sided approximation' to refer to both approximation from above and below.

If we let h^- be an approximation of a function g from below, and h^+ an approximation from above, then it follows that $h^- \le g \le h^+$. Thus, the bracket $[h^-, h^+]$ is well-defined. Further, for any bracket containing g,

$$\begin{aligned}\mu(h^+ \ne h^-) &= \mu(h^+ \ne h^-, g = 1) + \mu(h^+ \ne h^-, g = 0) \\ &= \mu(h^- = 0, g = 1) + \mu(h^+ = 1, g = 0) \\ &= \mu(h^- \ne g) + \mu(h^+ \ne g).\end{aligned}$$

Thus, if h^+ and h^- are respectively the minimisers of the right hand side, they must also be minimisers of the left hand side. Immediately, we have

$$\mathsf{B}(g, \mu, \mathcal{H}) = \mathsf{L}(g, \mu, \mathcal{H}) + \mathsf{L}(1-g, \mu, \mathcal{H}),$$

and the respective minimisers of the Ls form a μ-optimal $\mathcal{H}$-bracketing of g!

This means that in order to bracket g optimally, it suffices to *separately* learn approximations to g from above and below. This decouples the optimisation problems inherent in learning these, and allows easy convex relaxations of both the above problems.

Note that the reverse direction trivially holds - the optimal bracket containing g provides two functions which upper and lower approximate g. These functions are optimal for the respective OSL problems.

1.3 Convex Surrogates and ERM

§1.2 suggests one-sided learning as a method for learning bracketings. However, in an ML context, the optimisation problem of Def. 2 is meaningless since μ is not available. We approach this via empirical risk minimisation (ERM) (see also §2.1,§3.1).

To handle the intractable $0-1$ loss, we take the standard approach of relaxing the h to take values in $[0,1]$, subsequently thresholded to get a binary function, and the loss to a convex surrogate ℓ. We let θ be a parameterisation of h.

Importantly, in a practical context, while solutions that are always below g may be limited, a slight relaxation to 'nearly always' below can yield tenable classifiers. Adopting this view, we also relax the constraint, possibly by a different surrogate ℓ'[5], and allow an explicit user determined leakage constraint ζ.

Finally, as is standard, we propose solving a Lagrangian form of the resulting optimisation problem via SGD over θ. This gives the practical program

$$\min_{\theta} \sum_{i:g(x_i)=1} \frac{\ell(1-h_\theta(x_i))}{n_1} + \xi \sum_{i:g(x_i)=0} \frac{\ell'(h_\theta(x_i))}{n_0}, \quad (1)$$

with a Lagrange multiplier ξ, & $n_b = |\{i : g(X_i) = b\}|$.

The resulting bracketing scheme is as follows. The user may specify (ℓ, ℓ'), and a leakage constraint $\zeta \in (0,1]$. Each ξ in (1) yields a solution θ_ξ. We propose scanning over $\xi \in \Xi$, for some gridding Ξ. Next, for each ξ, we utilise a validation set V to compute the empirical means $\hat{\mu}_V(h_{\theta_\xi} \neq g)$ and $\hat{\mu}_V(h_{\theta_\xi} = 1, g = 0)$, and select the θ_ξ which minimises the first, subject to the second being smaller than $\zeta/2 - \overline{\mathrm{Bin}}(|V|, \zeta/2, \delta/2|\Xi|)$, where $\overline{\mathrm{Bin}}$ is the binomial tail inversion function as studied by Langford 2005. Such a selection gives a h^-. Similarly, we may learn an approximation from above h^+. Notice that with probability at least $(1-\delta)$, the $[h^-, h^+]$ so constructed is a ζ-approximate bracket that contains g (and thus has accuracy at least $1-\zeta$ w.r.t. g).

[5] e.g. ℓ' may grow faster than ℓ to minimise leakage

Multi-class Extensions In passing, we point out that our framework can be extended to multi-class setting. For an M-class setting, we may represent g as the one-hot encoding $(g_1, \ldots, g_M)$. Consistency in g demands that $\sum g_i = 1$. We may learn lower-approximations h_i to each g_i, and predict when only one of the h_i is 1. One issue is that this leads to identifying class-specific leakage-levels, which then need to be optimised globally to achieve usage constraints.

1.4 Comparison to Gating Formulation

The BL problem is typically formulated as simultaneously learning a gating function γ and a local predictor π, so that for a point x, if $\gamma = 1$, we predict locally using π, and if $\gamma = 0$, we instead call the function g. This yields usage $\mu(\gamma = 0)$ for the overall classifier

$$c_{\gamma,\pi}(x) = \pi(x)\gamma(x) + g(x)(1-\gamma(x)).$$

Notice that bracketing is in fact a restricted form of gating and prediction - the gate $\gamma = \mathbb{1}\{h^+ = h^-\}$, and the predictor (say) h^+. In fact these are essentially identical in their expressive power for a given 'richness': Suppose one learns gating and predictor functions γ and π from classes Γ and Π respectively[6] Given this,

$$h^+ := \gamma \cdot \pi + 1 - \gamma; \qquad h^- := \gamma \cdot \pi$$

bracket g with the same usage[7]. Crucially, the class of functions $\mathcal{H}$ generated by doing the above for every $(\gamma, \pi) \in \Gamma \times \Pi$ is a class of complexity equivalent to that of the pair (Γ, Π), since it can be described by the same pair. Thus there is *no loss of expressivity* in restricting attention to the bracketing setup.

1.5 A Summary of the Conclusions

The sections above establish the core of this paper via two formal reductions. Let us encapsulate these.

The gating-prediction formulation of budget learning is equivalent to the bracketing formulation. Further, solving the bracketing problem is equivalent to solving the two decoupled one-sided learning problems of learning from below and from above.

This statement justifies all further explorations in the paper. Since the bracketing formulation is equivalent, we may define budget learnability via it. Further, finite

[6] Observe that these must have comparable complexities, since they are both to be implemented on the same system.

[7] if $c_{\gamma,\pi}$ has accuracy $a < 1$, then these form an approximate bracket of the same accuracy.

sample analyses for the BL problem may be carried out via the one-sided learning problems.

2 Learnability

As mentioned in the previous paragraph, we define notions of one-sided and budget learnability.

2.1 One-sided Learnability

With only finite data, it is impossible to certify that $h \leq f$ for most h, rendering the one-sided constraint tricky. We take the PAC approach, and relax this condition by introducing a 'leakage parameter' λ.

Definition 3 *A class $\mathcal{H}$ is* one-sided learnable *if for all $(\varepsilon, \delta, \lambda) \in (0,1)^3$, there exists a $m(\varepsilon, \delta, \lambda, \mathcal{H}) < \infty$, and a scheme $\mathscr{A} : (\mathcal{X} \times \{0,1\})^m \to \mathcal{H}$ such that for any function-measure pair (g, μ), given m samples of $(X_i, g(X_i))$, with $X_i \overset{i.i.d.}{\sim} \mu$, $\mathscr{A}$ produces a function $h \in \mathcal{H}$ such that with probability at least $1 - \delta$:*

$$\mu(g(X) = 0, h(X) = 1) \leq \lambda$$
$$\mu(g(X) = 1, h(X) = 0) \leq \mathsf{L}(g, \mathcal{H}, \mu) + \varepsilon.$$

The above definition closely follows that of PAC learning in the agnostic setting, with the deviations that leakage is explicitly controlled, and that the excess risk control, ε, is on L, i.e. it is only with respect to *entirely* non-leaking functions. A key shared feature is that one-sided learnability is a property only of the class $\mathcal{H}$, and is agnostic to (g, μ).

If the class $\mathcal{H}$ is learnable, then with $m(\varepsilon, \lambda, \delta, \mathcal{H})$ samples we may learn a approximate-bracketing of any g with usage at most $\mathsf{B}(g, \mu, \mathcal{H}) + 2(\varepsilon + \lambda)$ and accuracy at least $1 - 2\lambda$.

Let us distinguish the above from the Neyman-Pearson classification setting of Cannon et al. 2002; Scott and Nowak 2005. The latter can be seen as learning from below, but with explicit control on the *conditional probability* $\mu(h = 1 | g = 0)$.[8] This is too strong for our needs - we are only interested in emulating the behaviour of g *with respect to* μ, and so if $\mu(g = 0) < \lambda$, then it is fine for us to learn any h. This induces the difference that the error rates in the cited papers decay with $\min(n_0, n_1)$, while our setting is simpler and PAC guarantees follow the entire sample size. Nevertheless, our claims on the sample complexity(§3.1) are derived similarly to the setting of 'NP-ERM' in these papers, including a testing and an optimisation phase.

[8] In addition, the targeted control on this is some level $\alpha > 0$, not 0, and a relaxation of the form we use to $\alpha + \lambda$ is also utilised. Further, the property of only comparing against the best classifier at the target level of leakage (α in their case, 0 in ours) is also shared.

2.2 Budget Learnability

The bracket-approximation of Def. 1 suffers from two problems in the ML context. Firstly, approximation by classes that are *not* one-sided learnable is irrelevant. Secondly, the definition does not control for effectiveness: a bracket-approximation with $\mathsf{B}(\mathcal{S}, \mathcal{H}) = 1$, is not useful - indeed, the trivial class $\mathcal{H} = \{0(x), 1(x)\}$ attains this for every $\mathcal{S}$. We propose the following to remedy these.

Definition 4 *We say that a set of function-measure pairs $\mathcal{S} = \{(g, \mu), \dots\}$ is budget-learnable by a class $\mathcal{H}$ if $\mathcal{H}$ is one-sided learnable and $\mathsf{B}(\mathcal{S}, \mathcal{H}) < 1$.*
We also, say that $\mathcal{H}$ can budget learn $\mathcal{S}$, adding "with budget B" if $\mathsf{B}(\mathcal{S}, \mathcal{H}) \leq \mathsf{B}$.

Learning theoretic settings usually require measure independent guarantees, leading to

Definition 5 *A function class $\mathcal{G}$ on the measurable space $(\mathcal{X}, \mathscr{F})$ is said to be budget learnable by a class $\mathcal{H}$ if the set $\mathcal{S} := \mathcal{G} \times \mathcal{M}$ is budget learnable by $\mathcal{H}$, where $\mathcal{M}$ is the set of all probability measures on $(\mathcal{X}, \mathscr{F})$.*

Notice that strict inequality is required in Def. 4. This is the weakest notion that is relevant in an ML context. Also note the trivial but useful regularity property that if $\mathcal{H}$ is one-sided learnable, then $\mathsf{B}(\mathcal{H} \times \mathcal{M}, \mathcal{H}) = 0$ - indeed, every $h \in \mathcal{H}$, is bracketed by $[h, h]$.

3 Theoretical Properties

This section details some useful consequences of the above definitions, which serve to highlight their utility.

3.1 One-sided learnability

Standard PAC-learning is intrinsically linked to the VC-dimension. The same holds for one-sided learnability.

Theorem 1. *If $\mathcal{H}$ has finite VC-dimension d, then it is one-sided learnable with*

$$m(\varepsilon, \lambda, \delta, \mathcal{H}) = \widetilde{O}\left(\left(\frac{1}{\lambda} + \frac{1}{\varepsilon^2}\right)(d + \log(1/\delta))\right).$$

Conversely, if $\mathcal{H}$ is one-sided learnable and has VC-dimension $d > 1$, then for $\delta < 1/100$,

$$m(\varepsilon, \lambda, \delta, \mathcal{H}) > \frac{d-1}{32(\lambda + \varepsilon)}.$$

Particularly, one-sided learnable classes must have finite VC-dimension.

The proof is left to Appx. A.1. The lower bound is proved via a reduction to realisable PAC learning, while the upper bound's proof is similar to that for agnostic PAC learning, with the modification of adding a test that eliminates functions that leak too much.

The point of the Theorem 1 is to illustrate that sample complexity analyses for our formulation can be derived via standard approaches in learning theory. Alternate analyses via, e.g., Rademacher complexty or covering numbers are also straightforward (Appx. A.1.1).

3.2 Budget Learnability

The key question of budget learning is one of bias: what classes of functions can be budget learned by low complexity classes? This section offers some partial results towards an answer.

Before we begin, the (big) question of how one measures complexity itself remains. We take a simple approach - since one-sided learnability itself requires finite VC-dimension, we call $\mathcal{H}$ low complexity if $\text{VC}(\mathcal{H})$ is small. Certainly VC dimension is a crude notion of complexity. Nevertheless this study leads to interesting bounds, and outlines how one may give theoretical analyses for more realistic settings that may be pursued in further work.

Importantly, we do not expect any one class to be able to meaningfully budget learn *all* classes of a given complexity. This follows since the definition of budget learnability implies that if sets $\mathcal{S}_1, \mathcal{S}_2$ of function-measure pairs are budget learnable, then so is $\mathcal{S}_1 \cup \mathcal{S}_2$. Such unions can lead to arbitrary increase in complexity, which must weaken the budget attained.[9] Thus, at the very least, the classes $\mathcal{H}$ must depend on $\mathcal{G}$, although we would like them to not depend on the measure.

3.2.1 Budget Learnability of Regular Classes

The class of Hölder smooth functions is a classical regularity assumption in non-parametric statistics. In this section, we define a natural analogue for $\{0,1\}$-valued functions, and discuss its budget learnability by low VC dimension classes. For simplicity, we restrict the input domain to the compact set $\mathcal{X} = [0,1]^p$. We use Vol to denote the Lebesgue measure on $\mathcal{X}$.

Definition *Let g be a $\{0,1\}$-valued function. A partition $\mathscr{P}$ of $\mathcal{X}$ is said to be aligned with g if each set $\Pi \in \mathscr{P}$ has connected interior, and if g is a constant on each such set.*

We define a notion of regularity for partitions below. Recall that a p-dimensional rectangle is a p-fold product of 1-D intervals.

Definition *A partition $\mathscr{P}$ is said to be V-regular if every part $\Pi \in \mathscr{P}$ contains a rectangle R_Π such that $\text{Vol}(R_\Pi) \geq V$ and $\text{Vol}(\Pi \setminus R_\Pi) < V$.*

[9] Formally, this finite union property and the lower bound Thm. 3 part (i) indicate that if $\mathcal{H}$ can budget learn *all* classes of VC dimension D on all measures with budget $1-c$ for any $c > 0$ that depends only on D or $\mathcal{X}$, but not on $\mathcal{H}$, then $\forall k \in \mathbb{N}, \text{VC}(\mathcal{H}) \geq CkD$ for a constant C.

The above partitions are well aligned with rectangles in the ambient space. The notion of regularity for function classes we choose to study demands that each function in the class has an associated 'nice' partition.

Definition 6 *We say that a class of functions $\mathcal{G} = \{g : [0,1]^p \to \{0,1\}\}$ is V-regular if for each $g \in \mathcal{G}$, there exists a V-regular partition aligned with g.*

Essentially the above demands that the local structure induced by any g can be neatly expressed. This condition is satisfied by many natural function classes on the bulk of their support - An important example is the class of g of the form $\mathbb{1}\{G(x) > 0\}$ for some Hölder smooth G that admit a margin condition with respect to the Lebesgue measure (see, e.g. Mammen and Tsybakov 1999; Tsybakov 2004). Indeed, if $\{G\}$ satisfies the margin condition $\text{Vol}(|G| < t) \leq \eta$, and is L-Lipschitz, then $\{\mathbb{1}\{G > 0\}\}$ is V-regular on a region of mass $\geq 1 - \eta$ with $V \geq (2t/L)^p$.

We offer the obvious class that can budget learn V-regular functions over sufficiently nice measures - rectangles. For $\kappa \in \mathbb{N}$, we define the class $\mathcal{R}_\kappa^{0,1}$ to consist of functions h that may be parametrised by k rectangles $\{R_i\}$ and a label $s \in \{0,1\}$, and take the form

$$h(x; \{R_i\}, s) = s\mathbb{1}\{x \in \cup R_i\} + (1-s)\mathbb{1}\{x \notin \cup R_i\}.$$

The class $\mathcal{R}_\kappa^{0,1}$ above has VC dimension at most $2p(\kappa+1)$. The theorem below offers bounds on the budgets required to learn V-regular classes in p dimensions:

Theorem 2. *Let $\kappa := \lfloor d/2p - 1 \rfloor \leq 1/V$. Suppose $\mu \ll \text{Vol}$, and $\frac{\mathrm{d}\mu}{\mathrm{d}\,\text{Vol}} \geq \rho$, and $\mathcal{G}$ is V-regular. Then $\text{VC}(\mathcal{R}_\kappa^{0,1}) = d$, and it can budget learn $\mathcal{G} \times \{\mu\}$ with*

$$\mathsf{B}\left(\mathcal{G} \times \{\mu\}, \mathcal{R}_\kappa^{0,1}\right) \leq 1 - \rho \left\lfloor \frac{d}{2p} - 1 \right\rfloor V/3.$$

Conversely, for $V \leq 1/2$, there exists a V-regular class $\mathcal{G}'$ such that if $\text{VC}(\mathcal{H}) \leq d$, then

$$\mathsf{B}(\mathcal{G}' \times \{\text{Vol}\}, \mathcal{H}) \geq 1 - \sqrt{3Vd\log(2e/V)}.$$

For the Lipschitz functions with margin discussed above, V scales as $\widetilde{\Theta}((C/L)^{-p})$, where L is the bound on the gradient, and C is some constant. The above shows that all such classes are learnable with budget $1 - \Omega(1)$ and VC-dim. d iff $d \gtrsim L^{-p+O(\log p)}$

3.2.2 Budget Learnability of bounded VC classes

Typically the function classes $\mathcal{G}$ that a cloud can implement are not nearly as rich as the set of all V-regular functions. This merits the investigation of classes with bounded (but large) complexity. Following the lines of study above, we investigate the budget learnability of finite VC classes, assuming $\text{VC}(\mathcal{G}) = D$ for large D.

Unlike covering numbers, bracketing numbers do not, in general, admit control for VC classes (e.g. constructions of van Handel 2013 and Malykhin 2012). This renders the budget learnability problem for bounded VC classes difficult. This is further complicated by the fact that we are interested in whether such classes can be meaningfully bracketed by *low-complexity* classes. Such questions are non-trivial to answer, and, frankly speaking, we do not solve the same. However, we offer two lower bounds, illustrating that if one wishes to non-trivially budget learn such classes with budget $1 - \Omega(1)$, and with VC dim. d, then d must grow as $\Omega(D)$. Further, we a present a few simple, natural cases where one *can* budget learn, irrespective of measure, with budget $\approx 1 - d/D$. We briefly discuss an open question that these classes stimulate.

3.3 Lower Bounds

For simplicity, we assume that $\mathcal{X} = [1 : N]$ for some $N \gg 1$, and that $\mathscr{F} = 2^{\mathcal{X}}$. The classes $\mathcal{G}, \mathcal{H}$ can then be identified as members of $2^{\mathscr{F}}$. Our lower bounds are captured by the following statements

Theorem 3.
(i) *(Varying measure) Let $\mathcal{G}$ be any class with $\mathrm{VC}(\mathcal{G}) = D$, and $\mathcal{H}$ with $\mathrm{VC}(\mathcal{H}) = d$. Then there exists a measure μ such that $\mathsf{B}(\mathcal{G} \times \{\mu\}, \mathcal{H}) \geq 1 - \sqrt{3\frac{d}{D}\log\frac{eD}{d}}$.*
(ii) *(Uniform measure) Let $N \in \mathbb{N}$, be a multiple of D such that $D \leq N/8e$. There exists a class $\mathcal{G}$ of VC-dimension D on $[1 : N]$such that for any class $\mathcal{H}$, if $\mathsf{B}(\mathcal{G} \times \{\mathrm{Unif}([1 : N])\}, \mathcal{H}) \leq \mathsf{B} \in (D/N, 1/4e)$, then $\mathrm{VC}(\mathcal{H}) \geq D\frac{\log(1/4e\mathsf{B})}{\log(eN)}$.*

The above bounds, while not very effective, indicate that to get small budget it is necessary that d grows linearly with D.

3.4 Some natural budget learnable classes

We present three simple examples:

- Sparse VC class: on the space $\mathcal{X} = [1 : N]$, let $\mathcal{G} = \binom{\mathcal{X}}{\leq D}$. Then this $\mathcal{G}$ can be budget learned by the class $\binom{\mathcal{X}}{\leq d}$ of VC dimension d with budget $1 - d/D$.

- Convex Polygons in the plane: Let $\mathcal{X} = \mathbb{R}^2$, and $\mathcal{P}_D$ be the set of concepts defined by marking the convex hull of any D points as $b \in \{0, 1\}$, and its exterior by $1 - b$. Takács 2007 shows that $\mathcal{P}_D$ has VC dimension $2D + 2$. For $d \geq 4$, the class $\mathcal{P}_d$ (of VC dimension $2d + 2$) can budget learn $\mathcal{P}_D$ with budget $1 - \lceil \frac{D}{d-2} \rceil^{-1} \approx 1 - (d/D)$ for $D \gg d$.

- Tensorisation of thresholds: Let $\mathcal{X} = [1 : N]$, and let $\mathcal{G}$ be defined as the following class: Let $\mathcal{G}_0$ be the class on $[1 : N/D]$ of the form $\mathbb{1}\{x \geq k\}$ for some k. We let $\mathcal{G} = \sum_{i=1}^{D} g_i$ where $g_i : [1+iN/D, (i+1)N/D] \to \{0, 1\}$ are of the form $g_i(x) = g_i'(x - iN/D)$ for some $g_i' \in \mathcal{G}_0$. Again, there exists a $\mathcal{H} \subset \mathcal{G}$ of VC dimension d that can budget learn $\mathcal{G}$ with budget $1 - d/D$.

Proofs for the above claims are left to Appendix A.4. There are two important features of the above classes, and their budget approximation

1. For each of the classes, there is a *subset* of these classes that has small VC dimension and can budget learn at (roughly) the budget $1 - d/D$. This subclass can be chosen irrespective of measure.
2. These classes are all extremal in the sense of satisfying the sandwich lemma with equality. In the first two cases they are maximal, while the third class is ample (see, e.g. Chalopin et al. 2018).

Maximal classes are known to admit unlabelled compression schemes of size equal to their VC dimension, and have many regularity properties - for instance, subclasses formed by restricting the class to some subset of the input are also maximal (see Chalopin et al. 2018 and references within). It is an interesting open question whether maximal classes of VC dimension D can be budget learned by *subclasses* of VC dimension d with usage $1 - cd/D$ for some constant c.

4 Experiments

This section presents empirical work implmenting the BL via bracketing schema on standard machine learning data. We explore three binary classification tasks

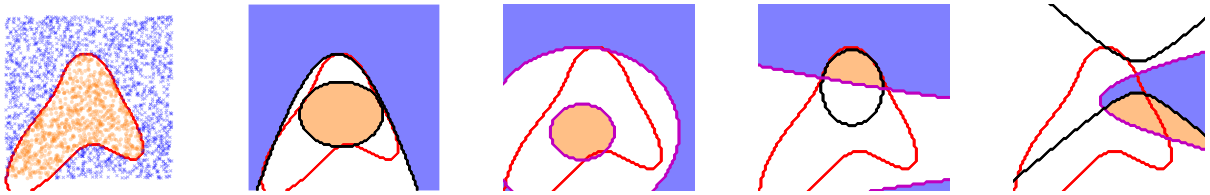
1. A simple *synthetic* task in $\mathbb{R}^2$ that allows easy visualization.
2. The *MNIST odd/even* task, which requires discrimination between odd and even MNIST digits.
3. The *CIFAR random pair* task, which requires discrimination between a pair of randomly chosen CIFAR-10 classes.[10]

The models considered are presented in Table 2. Each of the local classes chosen are far sparser than the corresponding cloud classes, which are taken to be the state of the art models for these tasks.

Bracketing is implemented as described in §1.3. See Appx. B for detailed descriptions. We compare the bracketing method to four existing approaches.

1. *Sum relaxation* (Sum Relax.) as proposed by Cortes et al. 2016, which relaxes the gating formulation to a sum as $\pi\gamma \leq (\pi + \gamma)/2$, and then further relaxes this to real valued outputs and convex surrogate losses.
2. *Alternating Minimiation* (Alt. Min.) as proposed

[10] Note: supervision is provided *after* this choice. That is, if class a and b are chosen, then the algorithms are provided the class a and class b data.



(a) Cloud boundary, and the training set. (b) Bracketing (Acc: 0.997; Usg:0.295) (c) Local thresholding (Acc:0.997; Usg:0.537) (d) Alt. Min. (Acc:0.996; Usg:0.563) (e) Sum Relax. (Acc:0.948; Usg:0.819)

Figure 1: Visualisation of classifiers resulting from the various approaches on a synthetic dataset. The red curve indicates the decision boundary of the cloud classifier, and figure (a) indicates this, and also shows the training set used as coloured dots. Figures (b)-(e) depict the budget learners learnt by various approaches. In these, the white region is the set of inputs on which the cloud is queried, while the orange and blue regions describe the decisions of the local predictor when it is queried. The black lines indicate decision boundaries of the various classifiers, and in figures (c)-(e), the magenta line indicates the boundary of the gate. Minimum usage solutions with accuracy at least 99.5% (when found) are presented.

Task	Target Acc.	Bracketing Usg.	Bracketing ROL	Local Thr. Usg.	Local Thr. ROL	Alt. Min. Usg.	Alt. Min. ROL	Sum relax. Usg.	Sum relax. ROL	Sel. Net. Usg.	Sel. Net. ROL	Gain
MNIST Odd/Even	0.995	0.457	2.19	0.653	1.53	0.830	1.20	0.785	1.27	0.658	1.52	1.431×
	0.990	0.387	2.58	0.515	1.94	0.740	1.35	0.651	1.54	0.544	1.84	1.332×
	0.980	0.299	3.35	0.358	2.79	0.604	1.66	0.651	1.54	0.423	2.37	1.199×
CIFAR Random Pair	0.995	0.363	4.01	0.510	2.25	0.854	1.19	0.620	2.07	0.436	3.04	1.280×
	0.990	0.294	5.66	0.399	3.41	0.754	1.40	0.488	3.31	0.347	4.30	1.265×
	0.980	0.214	9.97	0.276	6.38	0.611	1.87	0.345	5.81	0.257	11.67	1.195×

Table 1: Performances on BL tasks studied. Usage (usg.) and *relative operational lifetimes* (ROL), a common metric in BL which is the inverse of usage, are reported. In each case, the models attain the target accuracy (with respect to cloud) to less than 0.5% error - see Table 3 in Appx. B.6. *Gain* is the factor by which the bracketing usages are smaller than the best competitor. The CIFAR entries are averaged over 10 runs of the random choices. The results for all runs for the best two methods for are reported in Table 4 in Appx. B.6. Note that these are averages of each entry for each run and so average ROL is *not* the same as the inverse of the average usage.

Task	Cloud Classifier	Cloud Accuracy	Local Classifier	Local Accuracy
Synthetic	4th order curve	1.00	Axis-aligned Conic Sections (2nd order curves)	0.840
MNIST Odd/Even	LeNet 2conv + maxpool layers 43.7K params	0.995	Linear 1.57K params	0.898
CIFAR Random Pair	RESNET-32 0.46M params	0.984	Narrow LeNet 2conv + maxpool layers 1.63K params	0.909

Table 2: Classification tasks studied, and the corresponding cloud and local classifier classes selected. Cloud accuracy is reported with respect to true labels, but local accuracy is with respect to cloud labels.

by Nan and Saligrama 2017a, which introduces an auxiliary function u to serve as proxy for γ during training, replacing $\gamma\pi$ by $u\pi$. The algorithm then optimises a loss over (γ, π, u) via alternating minimisation over (γ, π) and then u, using a KL penalty $D(u\|\gamma)$ to promote $u \approx \gamma$.

3. *Selective Net* (Sel. Net.) as proposed by Geifman and El-Yaniv 2019 is an architectural modification for deep networks that essentially optimises the raw gating setup without any relaxation via SGD.

4. *Local Thresholding* (Local Thresh.). This is a naïve baseline - one learns a local classifier, and then rejects points if the entropy of its (soft) output at the point is too high.

In line with the focus of the paper, we only report solutions at high target accuracy ($\geq 98\%$). We note that local thresholding strictly outperforms the sum relaxation and alternating minimisation methods. The results are reported in Fig. 1 and Table 1. Observe that the bracketing methods show a consistent gain in usages at high accuracy, with reductions in usage by a factor of 1.2 to 1.5 times over the best competitors which are local thresholding for MNIST and Sel. Net. for CIFAR. In addition, the usages themselves are in the range 20-40% in most of the cases.

It is important to contextualise these usage numbers. In our choice of cloud and edge models, we are demanding that the edge models punch far above their weight when we try to budget learn the stated cloud classifiers - indeed, the edge models do not come even close to the clouds in standard accuracy. However, in Table 1, we see usages of 20-40% at high accuracies, and relative operational lifetimes of 2.5-5. For settings like IoT devices, where communication dominates energy costs, this is a significant gain in operational lifetimes of the prediction pipeline at near SOTA accuracy.

These results demonstrate that the bracketing methodology is practically implementable and effective, with the resulting budget learners clearly outperforming existing methods on the studied tasks.

Acknowledgements

Our thanks to Pengkai Zhu for help with implementing experiments. This work was supported partly by the National Science Foundation Grant 1527618, the Office of Naval Research Grant N0014-18-1-2257 and by a gift from the ARM corporation.

References

Bartlett, Peter L and Marten H Wegkamp (2008). "Classification with a reject option using a hinge loss". In: *Journal of Machine Learning Research* 9.Aug, pp. 1823–1840.

Bolukbasi, Tolga, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama (2017). "Adaptive neural networks for efficient inference". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 527–536.

Cannon, Adam, James Howse, Don Hush, and Clint Scovel (2002). "Learning with the Neyman-Pearson and min-max criteria". In: *Los Alamos National Laboratory, Tech. Rep. LA-UR*, pp. 02–2951.

Chalopin, Jérémie, Victor Chepoi, Shay Moran, and Manfred K Warmuth (2018). "Unlabeled sample compression schemes and corner peelings for ample and maximum classes". In: *arXiv preprint arXiv:1812.02099*.

CoreML, Apple Inc (2019). *CoreML Documentation*. Note: Product documentation, not peer-reviewed. Accessed on 2020-2-28. URL: https://developer.apple.com/documentation/coreml.

Cortes, Corinna, Giulia DeSalvo, and Mehryar Mohri (2016). "Learning with rejection". In: *International Conference on Algorithmic Learning Theory*. Springer, pp. 67–82.

Denis, Christophe and Mohamed Hebiri (2019). "Consistency of plug-in confidence sets for classification in semi-supervised learning". In: *Journal of Nonparametric Statistics*, pp. 1–31.

Geifman, Yonatan and Ran El-Yaniv (2017). "Selective classification for deep neural networks". In: *Advances in neural information processing systems*, pp. 4878–4887.

— (2019). "SelectiveNet: A Deep Neural Network with an Integrated Reject Option". In: *International Conference on Machine Learning*, pp. 2151–2159.

Gupta, Chirag, Arun Sai Suggala, Ankit Goyal, Harsha Vardhan Simhadri, Bhargavi Paranjape, Ashish Kumar, Saurabh Goyal, Raghavendra Udupa, Manik Varma, and Prateek Jain (2017). "ProtoNN: Compressed and Accurate kNN for Resource-scarce Devices". In: *International Conference on Machine Learning*, pp. 1331–1340.

Haussler, David (1995). "Sphere packing numbers for subsets of the Boolean n-cube with bounded Vapnik-Chervonenkis dimension". In: *Journal of Combinatorial Theory, Series A* 69.2, pp. 217–232.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Herbei, Radu and Marten H Wegkamp (2006). "Classification with reject option". In: *The Canadian Journal of Statistics/La Revue Canadienne de Statistique*, pp. 709–721.

Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean (2015). "Distilling the knowledge in a neural network". In: *arXiv preprint arXiv:1503.02531*.

Hollemans, Matthijs (2017). *Machine learning on mobile: on the device or in the cloud?* Note: Blog post, not peer-reviewed. Accessed on 2020-2-28. URL: http://machinethink.net/blog/machine-learning-device-or-cloud/.

Idelbayev, Yerlan (2019). *Proper ResNet Implementation for CIFAR10/CIFAR100 in pytorch*. Accessed on 2020-2-28. URL: https://github.com/akamaster/pytorch_resnet_cifar10.

Janisch, Jaromír, Tomáš Pevný, and Viliam Lisý (2019a). "Classification with Costly Features as a Sequential Decision-Making Problem". In: *arXiv preprint arXiv:1909.02564*.

— (2019b). "Classification with costly features using deep reinforcement learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 3959–3966.

Kumar, Ashish, Saurabh Goyal, and Manik Varma (2017). "Resource-efficient Machine Learning in 2 KB RAM for the Internet of Things". In: *International Conference on Machine Learning*, pp. 1935–1944.

Langford, John (2005). "Tutorial on practical prediction theory for classification". In: *Journal of machine learning research* 6, pp. 273–306.

Malykhin, Yu. V. (2012). "Bracketing entropy and VC-dimension". In: *Mathematical Notes* 91.5, pp. 800–807. ISSN: 1573-8876. DOI: 10.1134/S0001434612050264.

Mammen, Enno and Alexandre Tsybakov (1999). "Smooth discrimination analysis". In: *The Annals of Statistics* 27.6, pp. 1808–1829.

ML Kit, Google LLC (2019). *ML Kit Documentation*. Note: Product documentation, not peer-reviewed. Accessed on 2020-2-28. URL: https://developers.google.com/ml-kit.

Mohri, M., A. Rostamizadeh, and A. Talwalkar (2018). *Foundations of Machine Learning*. Adaptive Com-

putation and Machine Learning series. MIT Press. ISBN: 9780262039406.

Nan, Feng and Venkatesh Saligrama (2017a). "Adaptive classification for prediction under a budget". In: *Advances in Neural Information Processing Systems*, pp. 4727–4737.

— (2017b). "Dynamic model selection for prediction under a budget". In: *arXiv preprint arXiv:1704.07505*.

Nan, Feng, Joseph Wang, and Venkatesh Saligrama (2016). "Pruning random forests for prediction on a budget". In: *Advances in neural information processing systems*, pp. 2334–2342.

Norman, Hellen (2019). *Living on the Edge: Why On-Device ML is Here to Stay*. Note: Popular article, not peer-reviewed. Accessed on 2020-2-28. URL: https://community.arm.com/developer/ip-products/processors/b/ml-ip-blog/posts/why-on-device-ml-is-here-to-stay.

Peng, Yu-Shao, Kai-Fu Tang, Hsuan-Tien Lin, and Edward Chang (2018). "Refuel: Exploring sparse features in deep reinforcement learning for fast disease diagnosis". In: *Advances in Neural Information Processing Systems*, pp. 7322–7331.

Scott, Clayton and Robert Nowak (2005). "A Neyman-Pearson approach to statistical learning". In: *IEEE Transactions on Information Theory* 51.11, pp. 3806–3819.

Shekhar, Shubhanshu, Mohammad Ghavamzadeh, and Tara Javidi (2019). "Binary Classification with Bounded Abstention Rate". In: *arXiv preprint arXiv:1905.09561*.

Takács, Gábor (2007). "The vapnik-chervonenkis dimension of convex n-gon classifiers". In: *Hungarian Electronic Journal of Sciences*.

Trapeznikov, Kirill and Venkatesh Saligrama (2013). "Supervised Sequential Classification Under Budget Constraints". In: *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*. Vol. 31. Proceedings of Machine Learning Research. PMLR, pp. 581–589.

Tsybakov, Alexandre (2004). "Optimal aggregation of classifiers in statistical learning". In: *The Annals of Statistics* 32.1, pp. 135–166.

van der Vaart, Aad W. and Jon A Wellner (1996). *Weak convergence and empirical processes: with applications to statistics*. Springer.

van Handel, Ramon (2013). "The universal Glivenko–Cantelli property". In: *Probability Theory and Related Fields* 155.3, pp. 911–934. ISSN: 1432-2064. DOI: 10.1007/s00440-012-0416-5.

Wang, Joseph, Kirill Trapeznikov, and Venkatesh Saligrama (2015). "Efficient Learning by Directed Acyclic Graph For Resource Constrained Prediction". In: *Advances in Neural Information Processing Systems 28*.

Wiener, Yair and Ran El-Yaniv (2011). "Agnostic selective classification". In: *Advances in neural information processing systems*, pp. 1665–1673.

Wu, Bichen, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer (2019). "Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10734–10742.

Xu, Zhixiang (Eddie), Matt J. Kusner, Kilian Q. Weinberger, Minmin Chen, and Olivier Chapelle (2014). "Classifier Cascades and Trees for Minimizing Feature Evaluation Cost". In: *Journal of Machine Learning Research* 15, pp. 2113–2144. URL: http://jmlr.org/papers/v15/xu14a.html.

El-Yaniv, Ran and Yair Wiener (2010). "On the foundations of noise-free selective classification". In: *Journal of Machine Learning Research* 11.May, pp. 1605–1641.

Zhou, Li, Hao Wen, Radu Teodorescu, and David HC Du (2019). "Distributing deep neural networks with containerized partitions at the edge". In: *2nd {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 19)*.

Zhu, Pengkai, Durmus Alp Emre Acar, Nan Feng, Prateek Jain, and Venkatesh Saligrama (2019). "Cost aware inference for iot devices". In: *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2770–2779.