THE FORGETRON: A KERNEL-BASED PERCEPTRON ON A BUDGET

OFER DEKEL*, SHAI SHALEV-SHWARTZ†, AND YORAM SINGER‡

Abstract. The Perceptron algorithm, despite its simplicity, often performs well in online classification tasks. The Perceptron becomes especially effective when it is used in conjunction with kernel functions. However, a common difficulty encountered when implementing kernel-based online algorithms is the amount of memory required to store the online hypothesis, which may grow unboundedly as the algorithm progresses. Moreover, the running time of each online round grows linearly with the amount of memory used to store the hypothesis. In this paper, we present the *Forgetron* family of kernel-based online classification algorithms, which overcome this problem by restricting themselves to a predefined memory budget. We obtain different members of this family by modifying the kernel-based Perceptron in various ways. We also prove a unified mistake bound for all of the Forgetron algorithms. To our knowledge, this is the first online kernel-based learning paradigm which, on one hand, maintains a *strict* limit on the amount of memory it uses and, on the other hand, entertains a relative mistake bound. We conclude with experiments using real datasets, which underscore the merits of our approach.

Key words. online classification, kernel methods, the Perceptron algorithm, learning theory

AMS subject classifications. 68T05, 68Q32

1. Introduction. The introduction of the Support Vector Machine (SVM) [11] sparked a widespread interest in kernel methods as a means of solving binary classification problems. Although SVM was initially stated as a batch-learning technique, it significantly influenced the development of kernel methods in the online-learning setting. Online classification algorithms that can incorporate kernels include the Perceptron [10], ROMMA [9], ALMA [5], NORMA [7] and the Passive-Aggressive family of algorithms [2]. Each of these algorithms observes examples in a sequence of rounds, and constructs its classification function incrementally, by storing a subset of the observed examples in its internal memory. The classification function is then defined by a kernel-dependent combination of the stored examples. This set of stored examples is the online equivalent of the *support set* in SVMs, however in contrast to the support, it constantly changes as learning progresses. In this paper, we call this set the *active set*, as it includes those examples that actively define the current classifier. Typically, an example is added to the active set every time the online algorithm makes a prediction mistake, or when its confidence in a prediction is inadequately low. Under certain circumstances, the active set often grow to be very big, and this can lead to significant computational difficulties. Naturally, since computing devices have bounded memory resources, there is the danger that an online algorithm would require more memory than is physically available. This problem becomes especially eminent in cases where the online algorithm is implemented as part of a specialized hardware system with a small memory, such as a mobile telephone or an autonomous robot. Moreover, the growth of the active set can lead to unacceptably long running times, as the time-complexity of each online round scales linearly with the size of the active set.

Crammer, Kandola, and Singer [3] first addressed this problem by describing an online kernel-based modification of the Perceptron algorithm in which the active set

*School of CS and Eng., The Hebrew University, Jerusalem, 91904, Israel (oferd@cs.huji.ac.il)
†School of CS and Eng., The Hebrew University, Jerusalem, 91904, Israel (shais@cs.huji.ac.il)
‡Google Inc., 1600 Amphitheatre Parkway, Moutain View, California, 94043 (singer@google.com)

does not exceed a predefined *budget*. Their algorithm removes redundant examples from the active set in an attempt to make the best use of the limited memory resource. Weston, Bordes and Bottou [12] followed with their own online kernel machine on a budget. Both techniques work relatively well in practice, however they both lack formal guarantees on prediction accuracy.

In this paper we present an online kernel-based classifier which is restricted to a fixed budget of active examples and for which we derive a formal learning-theoretic analysis. To the best of our knowledge, this is the first online algorithm on a budget for which a rigorous mistake bound has been proven. Like [3], our approach also uses the kernel-based Perceptron as a starting point, and enforces the budget constraint by removing an example from the active set whenever the size of this set exceeds the predefined limit. We name our algorithm *Forgetron*, since it is a variation of the Perceptron algorithm which *forgets* active examples as necessary.

Besides forgetting active examples, the Forgetron algorithm also shrinks the online hypothesis every time it performs an update. This repeated shrinking technique is the key ingredient that makes our theoretical analysis possible. Every time a new example is added to the active set, the entire hypothesis is multiplied by a positive scalar which is at most 1, and often smaller than 1. This causes the weight of each active example to diminish from update to update. If this scaling procedure is done correctly, it ensures that there always exists an active example with a small weight and a minor influence on the current hypothesis. This example can be safely removed from the active set without causing serious damage to the accuracy of our online classifier. The scaling step should be performed carefully, since an over-aggressive scaling policy could significantly impair the algorithm's prediction abilities. The delicate balance between safe removal of active examples and over-aggressive scaling is the main accomplishment of this paper.

Following the preliminary presentation of the Forgetron algorithm [4], Cesa-Bianchi and Gentile devised a *randomized* online classification algorithm on a budget [1]. They also proved an upper bound on the expected number of prediction mistakes made by their algorithm. We revisit the algorithm of Cesa-Bianchi and Gentile in Sec. 8.

This paper is organized as follows. In Sec. 2 we begin with a more formal presentation of our problem and discuss a profound difficulty in proving mistake bounds for kernel-methods on a budget. In Sec. 3 and Sec. 4 we lay the groundwork for our algorithm by analyzing two possible modifications to the Perceptron algorithm. In Sec. 5 we derive the basic Forgetron algorithm, and in Sec. 6 and Sec. 7 we present two possible improvements to the basic algorithm. We conclude with an empirical evaluation of our algorithms in Sec. 8 and a discussion in Sec. 9.

2. Problem Setting. Online learning is performed in a sequence of consecutive rounds. On round t, the online algorithm observes an instance $\mathbf{x}_t$, which is drawn from some predefined instance domain $\mathcal{X}$. The algorithm predicts the binary label associated with that instance and is then given the correct label $y_t \in \{-1, +1\}$. At this point, the algorithm may use the new example $(\mathbf{x}_t, y_t)$ to improve its prediction mechanism for future rounds. We make no assumptions on the way in which the sequence of examples is generated. The goal of the algorithm is to correctly predict as many labels as possible.

The predictions of the online algorithm are determined by a function which is stored in its internal memory and is updated from round to round. We refer to this function as the *hypothesis* of the algorithm and denote the hypothesis used on round

t by f_t. Our focus in this paper is on margin based hypotheses, namely, f_t is a function from $\mathcal{X}$ to $\mathbb{R}$ where $\mathrm{sign}(f_t(\mathbf{x}_t))$ constitutes the actual binary prediction and $|f_t(\mathbf{x}_t)|$ is the confidence in this prediction. The term $yf(\mathbf{x})$ is called the *margin* of the prediction and is positive whenever y and $\mathrm{sign}(f(\mathbf{x}))$ agree. We can evaluate the performance of an hypothesis on a given example $(\mathbf{x}, y)$ in one of two ways. First, we can check whether the hypothesis makes a prediction mistake, namely determine whether $y = \mathrm{sign}(f(\mathbf{x}))$ or not. Throughout this paper, we use M to denote the number of prediction mistakes made by an online algorithm on a sequence of examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$. The second way we can evaluate the predictions of an hypothesis is by using the *hinge-loss* function, defined as

$$\ell\big(f; (\mathbf{x}, y)\big) \;=\; \begin{cases} 0 & \text{if } \; yf(\mathbf{x}) \geq 1 \\ 1 - yf(\mathbf{x}) & \text{otherwise} \end{cases} \quad . \tag{2.1}$$

The hinge-loss penalizes an hypothesis for any margin less than 1. Additionally, if $y \neq \mathrm{sign}(f(\mathbf{x}))$ then $\ell(f, (\mathbf{x}, y)) \geq 1$ and therefore the *cumulative hinge-loss* suffered over a sequence of examples upper bounds M. The algorithms discussed in this paper use kernel-based hypotheses, namely, they are defined with respect to a symmetric positive semidefinite kernel operator $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A kernel-based hypothesis takes the form

$$f(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i K(\mathbf{x}_i, \mathbf{x}) \quad , \tag{2.2}$$

where $\mathbf{x}_1, \ldots, \mathbf{x}_k$ are members of $\mathcal{X}$ and $\alpha_1, \ldots, \alpha_k$ are real valued weights. To facilitate the derivation of our algorithms and their analysis, we associate a Reproducing Kernel Hilbert Space (RKHS) with K in the standard way common to all kernel-based learning methods. First, we define the inner product between the functions $f(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i K(\mathbf{x}_i, \mathbf{x})$ and $g(\mathbf{x}) = \sum_{j=1}^{l} \beta_j K(\mathbf{z}_j, \mathbf{x})$ to be

$$\langle f, g \rangle = \sum_{i=1}^{k} \sum_{j=1}^{l} \alpha_i \beta_j K(\mathbf{x}_i, \mathbf{z}_j) \quad .$$

This inner-product naturally induces a norm defined by $\|f\| = \langle f, f \rangle^{1/2}$ and a metric $\|f - g\| = (\langle f, f \rangle - 2\langle f, g \rangle + \langle g, g \rangle)^{1/2}$. Next, we let $\mathcal{H}_K$ denote the closure of the set of all hypotheses of the form given in Eq. (2.2), with respect to this metric. These definitions play an important role in the analysis of our algorithms.

Online kernel methods typically restrict themselves to hypotheses that are defined by a subset of the examples observed on previous rounds. That is, the hypothesis used on round t takes the form

$$f_t(\mathbf{x}) = \sum_{i \in I_t} \alpha_i K(\mathbf{x}_i, \mathbf{x}) \quad , \tag{2.3}$$

where I_t is a subset of $\{1, \ldots, (t-1)\}$ and $\mathbf{x}_i$ is the instance observed on round i. As stated above, I_t is called the active set, and we say that example $(\mathbf{x}_i, y_i)$ is *active* on round t if $i \in I_t$.

Perhaps the most well known online algorithm for binary classification is the Perceptron [10]. Stated as a kernel method, the hypotheses generated by the Perceptron take the form $f_t(\mathbf{x}) = \sum_{i \in I_t} y_i K(\mathbf{x}_i, \mathbf{x})$. Namely, the weight assigned to each active

example is either $+1$ or -1, depending on the label of that example. The Perceptron initializes I_1 to be the empty set, which implicitly sets f_1 to be the zero function. It then updates its hypothesis only on rounds where a prediction mistake is made. Concretely, if on round t the margin $y_t f_t(\mathbf{x}_t)$ is non-positive then the index t is inserted into the active set. As a consequence, the size of the active set on round t equals the number of prediction mistakes made on previous rounds. A relative mistake bound can be proven for the Perceptron algorithm. The bound holds for any sequence of examples, and compares the number of mistakes made by the Perceptron with the cumulative hinge-loss of any fixed hypothesis $g \in \mathcal{H}_K$, even one defined with prior knowledge of the sequence.

THEOREM 2.1. *Let K be a kernel and let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples such that $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t. Let g be an arbitrary function in $\mathcal{H}_K$ and define $\ell_t^\star = \ell\big(g; (\mathbf{x}_t, y_t)\big)$. Then the number of prediction mistakes made by the Perceptron on this sequence is bounded by*

$$M \;\leq\; \|g\|^2 + 2\sum_{t=1}^{T} \ell_t^\star \quad .$$

The proof of this theorem is given in the next section, and serves as the basis of the analysis in this paper. Although the Perceptron is guaranteed to be competitive with any fixed hypothesis $g \in \mathcal{H}_K$, the fact that its active set grows with every mistake may pose a serious computational problem, as already noted in the introduction. In fact, this problem is common to most kernel-based online methods which do not explicitly monitor the size of I_t.

On the limitation of algorithms on a memory budget:. Our goal is to derive and analyze an online prediction algorithm which resolves the problems discussed above by enforcing a *fixed* bound on the size of the active set. Formally, let B be a positive integer which we refer to as the *budget parameter*. We would like to devise an algorithm which enforces the constraint $|I_t| \leq B$ on every round t. Furthermore, we would like to prove a relative mistake bound for this algorithm along the lines of Thm. 2.1. Regretfully, it turns out that this goal cannot be reached without making additional assumptions. We show this inherent limitation by presenting a simple counterexample. That is, for any kernel-based algorithm that uses a prediction function of the form given in Eq. (2.3), and which adheres to the constraint $|I_t| \leq B$, we can find a kernel K, an hypothesis $g \in \mathcal{H}_K$ and an arbitrarily long sequence of examples such that the algorithm makes a prediction mistake on every single round while g suffers no loss at all. Our counterexample is constructed as follows. We choose $\mathcal{X}$ to be the set of $B+1$ standard unit vectors in $\mathbb{R}^{B+1}$, namely $\mathcal{X} = \{e_i\}_{i=1}^{B+1}$ where e_i is the vector with 1 in its i'th coordinate and zeros elsewhere. The kernel function K is set to be the standard dot product in $\mathbb{R}^{B+1}$, thus $K(\mathbf{x}, \mathbf{x}') = \mathbf{x} \cdot \mathbf{x}'$. On every round t, the online hypothesis, f_t, is a linear combination of at most B vectors from $\mathcal{X}$. Since $|\mathcal{X}| = B+1$, there exists a vector $\mathbf{x}_t \in \mathcal{X}$ which is not currently active. Furthermore, by construction, $\mathbf{x}_t$ is orthogonal to all of the active vectors, and therefore $f_t(\mathbf{x}_t) = 0$. Assume without loss of generality that the online algorithm we are using predicts y_t to be -1 when $f_t(\mathbf{x}) = 0$. If on every round we were to present the online algorithm with the example $(\mathbf{x}_t, +1)$ then the online algorithm would make a prediction mistake on every round. On the other hand, the hypothesis $\bar{g} = \sum_{i=1}^{B+1} e_i$ is a member of H_K and attains a hinge-loss of 0 on every round. We have found a sequence of examples and a fixed hypothesis (which is indeed defined by more than B vectors from $\mathcal{X}$) that attains a cumulative loss of zero on this sequence, while the number of mistakes made

by our online algorithm equals the number of rounds. Clearly, a general theorem along the lines of Thm. 2.1 cannot be proven.

One way to resolve the problem illustrated above is to limit the set of competing hypotheses to a subset of $\mathcal{H}_K$ in a way that would naturally exclude $\bar{g}$ in the example above. In this paper, we limit the set of competitors to hypotheses with a bounded norm. Formally, we wish to devise an online algorithm which is competitive with every hypothesis $g \in \mathcal{H}_K$ for which $\|g\| \leq U$, where U is a predefined positive constant. The counterexample above indicates that we cannot prove a relative mistake bound with $U \geq \sqrt{B+1}$, since the norm of $\bar{g}$ in our counterexample is $\sqrt{B+1}$. In this paper we come close to this upper bound by proving that our algorithms can compete with any hypothesis whose norm is bounded from above by $\frac{1}{4}\sqrt{(B+1)/\log(B+1)}$. Limiting the set of competing hypotheses to a ball of norm U about the origin of $\mathcal{H}_K$ is one possible way to overcome the problem exposed by our counterexample. It seems plausible that other restrictions on the general problem setting, such as specific choices of the instance domain $\mathcal{X}$ or the kernel function K, could resolve this problem equally well.

As mentioned in the previous section, Cesa-Bianchi and Gentile devised a randomized online classification algorithm on a budget [1]. Their analysis shows that their algorithm is competitive with any hypothesis whose norm is bounded from above by $O(\sqrt{B+1})$. However, in contrast to our analysis that bounds the actual number of prediction mistakes (M), the analysis of Cesa-Bianchi and Gentile bounds the *expected* number of mistakes ($\mathbb{E}[M]$), where expectation is taken over the internal randomization of their algorithm. Namely, the actual performance of their randomized algorithm varies from run to run. We illustrate this phenomenon empirically in Sec. 8.

3. The Remove-Oldest Perceptron. The Perceptron algorithm and its mistake bound (Thm. 2.1) serve as our starting point. Therefore, it is important to understand the proof of Thm. 2.1 before proceeding. The key to proving Thm. 2.1 is the observation that the hypothesis of the Perceptron is drawn towards good hypotheses in $\mathcal{H}_K$. Specifically, whenever the Perceptron makes a prediction mistake, its hypothesis moves closer to every hypothesis $g \in \mathcal{H}_K$ which attains a margin of at least $\frac{1}{2}$ on the current example. This fact is formally stated and proven below.

LEMMA 3.1. *Let $(\mathbf{x}, y)$ be an example, where $\mathbf{x} \in \mathcal{X}$, $K(\mathbf{x},\mathbf{x}) \leq 1$, and $y \in \{-1,+1\}$. Let $f \in \mathcal{H}_K$ be a function such that $yf(\mathbf{x}) \leq 0$, and define $f' = f + yK(\mathbf{x},\cdot)$. Then for any function $g \in \mathcal{H}_K$ it holds that*

$$\|f-g\|^2 - \|f'-g\|^2 \;\geq\; 2yg(\mathbf{x}) - 1 \quad .$$

Proof. Using the definition of f' we can write,

$$\begin{aligned}
&\|f-g\|^2 - \|f'-g\|^2 \\
&= \|f-g\|^2 - \|(f-g) + yK(\mathbf{x},\cdot)\|^2 \\
&= \|f-g\|^2 - \|f-g\|^2 - 2y\langle (f-g), K(\mathbf{x},\cdot)\rangle - K(\mathbf{x},\mathbf{x}) \\
&= -2y\langle f, K(\mathbf{x},\cdot)\rangle + 2y\langle g, K(\mathbf{x},\cdot)\rangle - K(\mathbf{x},\mathbf{x}) \quad .
\end{aligned}$$

Using the reproducing property of $\mathcal{H}_K$, we know that $\langle f, K(\mathbf{x},\cdot)\rangle = f(\mathbf{x})$ and that $\langle g, K(\mathbf{x},\cdot)\rangle = g(\mathbf{x})$, and thus we get

$$\|f-g\|^2 - \|f'-g\|^2 \;=\; -2yf(\mathbf{x}) + 2yg(\mathbf{x}) - K(\mathbf{x},\mathbf{x}) \quad . \tag{3.1}$$

Using our assumption that $yf(\mathbf{x}) \leq 0$, it follows that $-2yf(\mathbf{x}) \geq 0$. Additionally, recall that we made the assumption that $K(\mathbf{x}, \mathbf{x}) \leq 1$. Plugging these facts back into Eq. (3.1) gives

$$\|f - g\|^2 - \|f' - g\|^2 \;\geq\; 2yg(\mathbf{x}) - 1 \quad . \tag{3.2}$$

This concludes the proof. □

The term $\|f_t - g\|^2 - \|f_{t+1} - g\|^2$ measures how much the hypothesis of the Perceptron gets closer to g, as a result of the update on round t. This term plays an important role in our paper and we therefore denote it by Δ_t. It is worth noting that Δ_t also plays an important role in the analysis of other online algorithms [8, 6, 2]. The proof of Thm. 2.1 is a simple corollary of Lemma 3.1.

Proof. [Proof of Thm. 2.1] We prove the theorem by bounding $\sum_{t=1}^{T} \Delta_t$ from above and from below. First note that $\sum_t \Delta_t$ is a telescopic sum, which reduces to

$$\sum_{t=1}^{T} \Delta_t \;=\; \|f_1 - g\|^2 - \|f_{T+1} - g\|^2 \quad .$$

Using the facts that $\|f_{T+1} - g\|^2 \geq 0$ and that f_1 is the zero function, we can upper bound $\sum_t \Delta_t$ by $\|g\|^2$. Next we show a lower bound on $\sum_t \Delta_t$. For rounds on which the Perceptron makes a correct prediction, we have that $f_{t+1} = f_t$ and thus $\Delta_t = 0$. For rounds on which the Perceptron makes a mistake, Lemma 3.1 tells us that $\Delta_t \geq 2y_t g(\mathbf{x}_t) - 1$. The definition of the hinge-loss in Eq. (2.1) implies that $\ell_t^\star \geq 1 - y_t g(\mathbf{x}_t)$ and therefore $2y_t g(\mathbf{x}_t) \geq 2 - 2\ell_t^\star$. Therefore, we have that

$$\|f_t - g\|^2 - \|f_{t+1} - g\|^2 \geq \; 1 - 2\ell_t^\star \quad .$$

Recalling that M denotes the total number of prediction mistakes made on the entire sequence of examples, we obtain that

$$\sum_{t=1}^{T} \Delta_t \;\geq\; M - 2 \sum_{t : y_t f_t(\mathbf{x}_t) \leq 0} \ell_t^\star \quad .$$

Since the hinge-loss is non-negative, it holds that

$$\sum_{t=1}^{T} \Delta_t \;\geq\; M - 2 \sum_{t=1}^{T} \ell_t^\star \quad . \tag{3.3}$$

Comparing this lower bound with the upper bound $\sum_t \Delta_t \leq \|g\|^2$ and rearranging terms proves the theorem. □

We now present the *Remove-Oldest* Perceptron, a simple modification of the kernel Perceptron, which conforms with a fixed budget constraint. As long as the active set is smaller than the budget parameter B, the Remove-Oldest Perceptron behaves exactly like the standard kernel Perceptron. The active set therefore grows with every mistake and eventually contains B examples. Once the active set reaches B examples, the online update is performed in two steps. Whenever the algorithm makes a mistake, it first adds an example to the active set by performing the standard Perceptron update, and then it reduces the size of the active set back to B by removing the oldest active example. More formally, for all $1 \leq t \leq T$, let I'_t define the active set obtained on round t after applying the standard Perceptron update. That is,

$$I'_t \;=\; \begin{cases} I_t & \text{if } y_t f_t(\mathbf{x}_t) > 0 \\ I_t \cup \{t\} & \text{if } y_t f_t(\mathbf{x}_t) \leq 0 \end{cases} \quad . \tag{3.4}$$

Also, let f'_t denote the hypothesis defined by I'_t, namely,

$$f'_t \;=\; \sum_{t \in I'_t} y_t K(\mathbf{x}_t, \cdot) \quad . \tag{3.5}$$

Now, define I_{t+1} to be

$$I_{t+1} \;=\; \begin{cases} I'_t \setminus \{r_t\} & \text{if } |I'_t| = B+1 \\ I'_t & \text{if } |I'_t| \leq B \end{cases} \quad , \tag{3.6}$$

where $r_t = \min I'_t$. Besides being an interesting algorithm, the Remove-Oldest Perceptron is an important intermediate step towards the Forgetron algorithm.

We are unable to prove a mistake bound for the Remove-Oldest Perceptron, however we are able to quantify the damage due to the second step of the update, defined in Eq. (3.6). Assume that the algorithm is run for T rounds. Let J denote the set of rounds on which a prediction mistake is made, namely $J = \{1 \leq t \leq T : y_t f_t(\mathbf{x}_t) \leq 0\}$ and $M = |J|$. Note that I_{T+1}, the active set at the end of T rounds, is a subset of J. To analyze the Remove-Oldest Perceptron, we again assume that g is an arbitrary function in $\mathcal{H}_K$ and define $\Delta_t = \|f_t - g\|^2 - \|f_{t+1} - g\|^2$. As in the proof of Thm. 2.1, the sum $\sum_{t=1}^{T} \Delta_t$ can be upper bounded by $\|g\|^2$ and we concentrate on bounding it from below. Using the notation f'_t, defined in Eq. (3.5), we can rewrite Δ_t as follows,

$$\Delta_t = \|f_t - g\|^2 - \|f'_t - g\|^2 + \|f'_t - g\|^2 - \|f_{t+1} - g\|^2 \quad .$$

For brevity, let us define

$$\alpha_t = \|f_t - g\|^2 - \|f'_t - g\|^2 \qquad \text{and} \qquad \gamma_t = \|f'_t - g\|^2 - \|f_{t+1} - g\|^2 \quad ,$$

and thus $\sum_t \Delta_t = \sum_t \alpha_t + \sum_t \gamma_t$. Since $\Delta_t \neq 0$ only for $t \in J$, we can rewrite

$$\sum_{t=1}^{T} \Delta_t \;=\; \sum_{t \in J} (\alpha_t + \gamma_t) \;=\; \sum_{t \in I_{T+1}} \alpha_t \;+\; \sum_{t \in J \setminus I_{T+1}} \alpha_t \;+\; \sum_{t \in J} \gamma_t \quad . \tag{3.7}$$

The summands in $\sum_{t \in J} \gamma_t$ which are equal to zero can be omitted from the sum. Specifically, note that γ_t is nonzero only on rounds for which $|I_t| = B$, and therefore,

$$\sum_{t \in J} \gamma_t \;=\; \sum_{t \in J \,:\, |I_t| = B} \gamma_t \quad . \tag{3.8}$$

The set $J \setminus I_{T+1}$ consists of the indices of the examples that were inserted into the active set and later removed from it. Another way to write this set, using the notation r_t defined above, is $\{r_t \;:\; t \in J \;\wedge\; |I_t| = B\}$, since active examples are removed precisely on rounds on which a mistake occurs and the active set is full. Therefore, it holds that

$$\sum_{t \in J \setminus I_{T+1}} \alpha_t \;=\; \sum_{t \in J \,:\, |I_t| = B} \alpha_{r_t} \quad . \tag{3.9}$$

Using Eq. (3.8) and Eq. (3.9), we can rewrite Eq. (3.7) as

$$\sum_{t=1}^{T} \Delta_t \;=\; \sum_{t \in I_{T+1}} \alpha_t \;+\; \sum_{t \in J \,:\, |I_t| = B} (\alpha_{r_t} + \gamma_t) \quad . \tag{3.10}$$

We have rewritten $\sum_t \Delta_t$ as the sum of two terms. Next, we lower-bound each term individually. The first term deals with examples that were added to the active set and never removed. For these examples, only the effect of the standard Perceptron update (α_t) must be taken into account. The second term deals with examples that were added and then later removed from the active set. Decomposing $\sum \Delta_t$ in this way, and dealing with the two terms separately, is an important technique which we reuse in our main formal result, namely the proof of Thm. 6.2.

We first consider the first term on the right-hand side of Eq. (3.10). For every $t \in I_{T+1}$ we can use Lemma 3.1 to bound $\alpha_t \geq 2y_t g(\mathbf{x}_t) - 1$. Using the definition of the hinge-loss in Eq. (2.1), we know that $2y_t g(\mathbf{x}_t) - 1 \geq 1 - 2\ell_t^\star$ and therefore,

$$\alpha_t \;\geq\; 1 - 2\ell_t^\star \quad . \tag{3.11}$$

Moving on to the second term on the right-hand side of Eq. (3.10), we note that for every round t on which an example was removed from the active set, α_{r_t} measures the benefit of initially adding the example r_t to the active set, whereas γ_t measures the damage caused by removing this example later on. We will actually analyze a more general case where instead of entirely removing example r_t on round t, we may only decrease its weight. In other words, instead of subtracting $y_{r_t} K(\mathbf{x}_{r_t}, \cdot)$ from the current hypothesis, we subtract $\lambda y_{r_t} K(\mathbf{x}_{r_t}, \cdot)$ for some $0 < \lambda \leq 1$. For the purpose of lower-bounding Eq. (3.10), we can simply assume that $\lambda = 1$. However, the more general form of our analysis will prove useful later on, as we make further progress toward an algorithm with a budget constraint and a mistake bound.

Next, we show that our lower bound on $\alpha_{r_t} + \gamma_t$ is influenced by two factors: the parameter λ, which determines what portion of the example $\mathbf{x}_{r_t}$ is removed, and the term $y_{r_t} f'_t(\mathbf{x}_{r_t})$, which is the margin attained by the current hypothesis on the example being removed. More precisely, we show that the lower bound on $\alpha_{r_t} + \gamma_t$ is similar to the lower bound in Eq. (3.11) minus the additional penalty

$$\Psi(\lambda, \mu) \;=\; \lambda^2 + 2\lambda - 2\lambda\mu \quad , \tag{3.12}$$

where μ is an abbreviation for $y_{r_t} f'_t(\mathbf{x}_{r_t})$.

LEMMA 3.2. *Let f, f', and g be arbitrary functions in $\mathcal{H}_K$ and let $(\mathbf{x}, y)$ be an example such that $\mathbf{x} \in \mathcal{X}$, $K(\mathbf{x}, \mathbf{x}) \leq 1$ and $y \in \{-1, +1\}$, and define $\ell^\star = \ell(g; (\mathbf{x}, y))$. Assume that $yf(\mathbf{x}) \leq 0$. Then for any $\lambda \in (0, 1]$ it holds that*

$$\begin{aligned}\Big(\|f - g\|^2 - \|(f + yK(\mathbf{x}, \cdot)) - g\|^2\Big) + \Big(\|f' - g\|^2 - \|(f' - \lambda y K(\mathbf{x}, \cdot)) - g\|^2\Big) \\ \geq\; 1 - 2\ell^\star - \Psi(\lambda,\, yf'(\mathbf{x})) \quad .\end{aligned}$$

Proof. We rewrite $\|f' - g\|^2 - \|(f' - \lambda y K(\mathbf{x}, \cdot)) - g\|^2$ as,

$$\begin{aligned}&\|f' - g\|^2 - \|f' - \lambda y K(\mathbf{x}, \cdot) - g\|^2 \\ &=\; \|f' - g\|^2 - \|f' - g\|^2 + 2\lambda y \langle f' - g, K(\mathbf{x}, \cdot)\rangle - \lambda^2 \|K(\mathbf{x}, \cdot)\|^2 \\ &=\; 2\lambda y \langle f', K(\mathbf{x}, \cdot)\rangle - 2\lambda y \langle g, K(\mathbf{x}, \cdot)\rangle - \lambda^2 \|K(\mathbf{x}, \cdot)\|^2\end{aligned}$$

Using the reproducing property of $\mathcal{H}_K$, it holds that $\langle f', K(\mathbf{x}, \cdot)\rangle = f'(\mathbf{x})$, $\langle g, K(\mathbf{x}, \cdot)\rangle = g(\mathbf{x})$ and $\|K(\mathbf{x}, \cdot)\|^2 = K(\mathbf{x}, \mathbf{x})$. Plugging these equalities into the above, and using our assumption that $K(\mathbf{x}, \mathbf{x}) \leq 1$, we have

$$\|f' - g\|^2 - \|f' - \lambda y K(\mathbf{x}, \cdot) - g\|^2 \;\geq\; 2\lambda y f'(\mathbf{x}) - 2\lambda y g(\mathbf{x}) - \lambda^2 \quad . \tag{3.13}$$

Using Lemma 3.1 and denoting $f' = f + yK(\mathbf{x}, \cdot)$ we get the bound

$$\|f - g\|^2 - \|f' - g\|^2 \;\geq\; 2yg(\mathbf{x}) - 1 \quad . \tag{3.14}$$

For brevity, let us denote the term on the left-hand side of the statement of the lemma by δ. Summing Eq. (3.14) with Eq. (3.13), we have

$$\delta \;\geq\; 1 - 2(1-\lambda)(1 - yg(\mathbf{x})) - \Big(2\lambda + \lambda^2 - 2\lambda y f'(\mathbf{x})\Big) \quad .$$

Using the definition of the hinge loss, it holds that $\ell^\star \geq 1 - yg(\mathbf{x})$, and since $(1-\lambda) \geq 0$, we get

$$\delta \;\geq\; 1 - 2(1-\lambda)\ell^\star - \Big(2\lambda + \lambda^2 - 2\lambda y f'(\mathbf{x})\Big) \quad .$$

Finally, we neglect the non-negative term $2\lambda\ell^\star$, and the lemma is proven. □

Using Lemma 3.2 with λ set to 1, and f' set to f'_t, we get

$$\alpha_{r_t} + \gamma_t \;\geq\; 1 - 2\ell^\star_{r_t} - \Psi(1,\, y_{r_t} f'_t(\mathbf{x}_{r_t})) \quad . \tag{3.15}$$

Combining Eq. (3.10) with Eq. (3.11) and Eq. (3.15), we obtain the lower bound

$$\sum_{t=1}^{T} \Delta_t \;\geq\; M - \;2\sum_{t \in J} \ell^\star_t \;- \sum_{t \in J\,:\,|I_t| = B} \Psi(1,\, y_{r_t} f'_t(\mathbf{x}_{r_t})) \quad .$$

Comparing this bound to the upper bound $\sum_t \Delta_t \leq \|g\|^2$, and using the fact that the hinge loss is always non-negative, yields the following corollary.

COROLLARY 3.3. *Let K be a symmetric positive semidefinite kernel and let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples such that $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t. Let g be an arbitrary function in $\mathcal{H}_K$, define $\ell^\star_t = \ell\big(g; (\mathbf{x}_t, y_t)\big)$ and let Ψ be as define in Eq. (3.12). Then the number of prediction mistakes made by the Remove-Oldest-Perceptron on this sequence is bounded by*

$$M \;\leq\; \|g\|^2 + 2\sum_{t=1}^{T} \ell^\star_t \;+ \sum_{t \in J\,:\,|I_t| = B} \Psi(1,\, y_{r_t} f'_t(\mathbf{x}_{r_t})) \quad .$$

This corollary does not constitute a relative mistake bound since we cannot provide any guarantee on the value of $\Psi(1,\, y_{r_t} f'_t(\mathbf{x}_{r_t}))$. The magnitude of this term depends on how well the classifier f'_t classifies the example being removed, $\mathbf{x}_{r_t}$. Referring back to the definition of Ψ in Eq. (3.12), we get that $\Psi(1,\, y_{r_t} f'_t(\mathbf{x}_{r_t})) = 3 - 2y_{r_t} f'_t(\mathbf{x}_{r_t})$. Therefore, every time $y_{r_t} f'_t(\mathbf{x}_{r_t}) \geq \frac{3}{2}$, the bound in Corollary 3.3 is actually strengthened, whereas every time $y_{r_t} f'_t(\mathbf{x}_{r_t}) < \frac{3}{2}$ it is weakened. Clearly, the term $y_{r_t} f'_t(\mathbf{x}_{r_t})$ plays an important role in determining whether or not $\mathbf{x}_{r_t}$ can be safely removed from the active set on round t. In order to obtain a concrete mistake bound, we must modify the remove-oldest Perceptron in a way which controls the damage caused by the removal step. Lemma 3.2, in its general form ($0 < \lambda \leq 1$), helps us gain this control. Namely, we can control the magnitude of the term $\Psi(\lambda,\, y_{r_t} f'_t(\mathbf{x}_{r_t}))$ by ensuring that $|f'_{r_t}(\mathbf{x}_t)|$ is sufficiently small, and by setting λ to a value smaller than 1. Both tasks can be achieved by repeatedly shrinking the online hypothesis on every update. The details of this modification are discussed in the next section.

4. Repeatedly Shrinking the Perceptron Hypothesis. In the previous section, we discussed the damage caused by removing the oldest active example from the active set. The key to controlling the extent of this damage is to ensure that the example being removed has a sufficiently small influence on the current hypothesis. One way to achieve this is by shrinking the norm of the online hypothesis following each update. Namely, on each round t where an update is performed, the online hypothesis is multiplied by a scalar $0 < \phi_t \leq 1$ (the concrete value of ϕ_t is specified in the next section). To study the effect of the shrinking step on the accuracy of the online algorithm, let us momentarily forget about the removal step introduced in the previous section and focus only on the shrinking step. The two techniques, removal and shrinking, are combined in the next section.

To facilitate the analysis of the shrinking technique, we introduce a new online algorithm, the *Shrinking* Perceptron. This algorithm is a variation of the standard kernel-based Perceptron and constructs an online hypothesis which is a *weighted* combination of functions in $\mathcal{H}_K$,

$$f_t \;=\; \sum_{i \in I_t} y_i \sigma_{i,t} K(\mathbf{x}_i, \cdot) \;\;,$$

where $\sigma_{i,t} \in [0,1]$. The update procedure starts with the standard Perceptron update. Specifically, if a correct prediction is made then $f_{t+1} = f_t$. Otherwise, I_{t+1} is set to $I_t \cup \{t\}$ and $\sigma_{t,t}$ is set to 1. We use the notation f'_t to denote the intermediate hypothesis which results from this update, namely,

$$f'_t(\mathbf{x}) \;=\; f_t(\mathbf{x}) + y_t \sigma_{t,t} K(\mathbf{x}_t, \mathbf{x}) \;\;. \tag{4.1}$$

The second step of the update is the shrinking step, which sets f_{t+1} to be $\phi_t f'_t$, where ϕ_t is a shrinking coefficient in $(0,1]$. Setting $\sigma_{i,t+1} = \phi_t \sigma_{i,t}$ for all $1 \leq i \leq t$, we can write

$$f_{t+1} \;=\; \sum_{i \in I_{t+1}} y_i \sigma_{i,t+1} K(\mathbf{x}_i, \cdot) \;\;.$$

The recursive definition of each weight $\sigma_{i,t}$ can be unraveled to give the following explicit form,

$$\sigma_{i,t} \;=\; \prod_{j \in I_{t-1} \,\wedge\, j \geq i} \phi_j \;\;.$$

By choosing sufficiently small shrinking coefficients ϕ_t, we can make the weights $\sigma_{i,t}$ decrease as fast as we like. If these weights indeed decrease rapidly enough, the contribution of older active examples to the online hypothesis becomes negligible. This demonstrates the potential of shrinking as a means of controlling the effect of old active examples on the current hypothesis. However, this benefit comes at a price. Repeatedly shrinking the norm of the Perceptron hypothesis takes a toll on the accuracy of the online algorithm. A good choice of ϕ_t should balance the need to attenuate the influence of older active examples with the damage caused by the shrinking step. In the remainder of this section, we prove a bound on the damage caused by the shrinking step.

To remind the reader, our goal, as stated in Sec. 2, is to find an algorithm which is competitive with any $g \in \mathcal{H}_K$ whose norm $\|g\|$ is bounded above by U, where $U = \frac{1}{4}\sqrt{(B+1)/\log(B+1)}$. The term $\Delta_t = \|f_t - g\|^2 - \|f_{t+1} - g\|^2$, which played

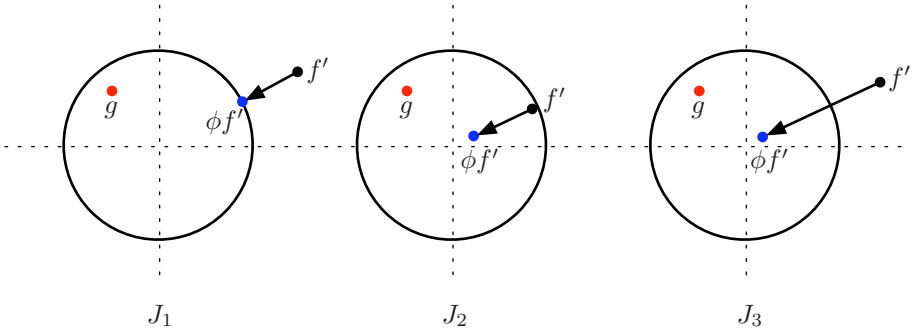

FIG. 4.1. *A geometrical interpretation of the three hypothesis-shrinking cases.*

a major role in the proof of Thm. 2.1, again appears in our analysis. We now show how this term is affected by the shrinking step. As before $\Delta_t = 0$ on rounds where a correct prediction was made, and we can focus on rounds where $\Delta_t \neq 0$. As before, we denote the set of indices t for which $\Delta_t > 0$ by J. Using the notation f'_t defined above, we can rewrite Δ_t as

$$\Delta_t = \|f_t - g\|^2 - \|f'_t - g\|^2 + \|f'_t - g\|^2 - \|f_{t+1} - g\|^2 \quad .$$

For brevity, define

$$\alpha_t = \|f_t - g\|^2 - \|f'_t - g\|^2 \qquad \text{and} \qquad \beta_t = \|f'_t - g\|^2 - \|f_{t+1} - g\|^2 \quad , \tag{4.2}$$

and so $\sum_t \Delta_t = \sum_{t\in J} \alpha_t + \sum_{t\in J} \beta_t$. For each t, α_t measures the progress made by the Perceptron update on round t, while β_t measures the damage caused by the shrinking step which follows the Perceptron update. Our first task is to lower-bound $\sum_{t\in J} \beta_t$. In order to do so, we partition the set J into the following three subsets,

$$\begin{aligned} J_1 &= \{t \in J \ : \ \phi_t \|f'_t\| \geq U\} \\ J_2 &= \{t \in J \ : \ \|f'_t\| \leq U \ \wedge \ \phi_t \|f'_t\| < U\} \\ J_3 &= \{t \in J \ : \ \|f'_t\| > U \ \wedge \ \phi_t \|f'_t\| < U\} \quad . \end{aligned} \tag{4.3}$$

To gain some intuition, we can think of the shrinking step in geometric terms. On round t, we first apply the Perceptron update and obtain f'_t. The function f'_t is a point in the Hilbert space $\mathcal{H}_K$. Then, we perform the shrinking step which moves f'_t towards the origin of $\mathcal{H}_K$, resulting in f_{t+1}. Now let $\mathcal{B}_U \subset \mathcal{H}_K$ be a ball of radius U, centered at the origin of $\mathcal{H}_K$. The set J_1 represents those rounds where both f'_t and f_{t+1} lie outside or on the surface of $\mathcal{B}_U$. The set J_2 represents the rounds where $f'_t \in \mathcal{B}_U$ and f_{t+1} lies in the interior of $\mathcal{B}_U$. Finally, J_3 represents rounds where $f'_t \notin \mathcal{B}_U$ and the shrinking step moves f_{t+1} into the interior of $\mathcal{B}_U$. This geometric interpretation is illustrated in Fig. 4.1. We now deal with each of the three cases individually, beginning with the set J_1. The following lemma builds on our assumption that $\|g\| \leq U$.

LEMMA 4.1. *Let $U > 0$ and $0 < \phi \leq 1$ be scalars and let g and f be two functions in $\mathcal{H}_K$ such that $\|g\| \leq U \leq \phi\|f\|$. Then,*

$$\|f - g\|^2 - \|\phi f - g\|^2 \geq 0 \quad .$$

Proof. We begin by noting $\phi\|f\|^2 \geq U\|f\| \geq \|g\|\|f\|$. Using the Cauchy-Schwartz inequality, we have that $\|g\|\|f\| \geq \langle f, g\rangle$, and therefore

$$\phi\|f\|^2 \;\geq\; \langle f, g\rangle \quad . \tag{4.4}$$

The term $\|f - g\|^2 - \|\phi f - g\|^2$ can now be rewritten as,

$$\begin{aligned} \|f - g\|^2 - \|\phi f - g\|^2 &= \left(\|f\|^2 - 2\langle f, g\rangle + \|g\|^2\right) - \left(\phi^2\|f\|^2 - 2\phi\langle f, g\rangle + \|g\|^2\right) \\ &= (1 - \phi^2)\|f\|^2 - 2(1 - \phi)\langle f, g\rangle \quad . \end{aligned} \tag{4.5}$$

Since $(1 - \phi)$ is non-negative, we can plug Eq. (4.4) into the right-hand side above and get the bound,

$$\|f - g\|^2 - \|\phi f - g\|^2 \;\geq\; (1 - \phi^2)\|f\|^2 - 2(1 - \phi)\phi\|f\|^2 \;=\; (1 - \phi)^2\,\|f\|^2 \;\geq\; 0 \quad .$$

□

To recap, the geometric implication of Lemma 4.1 is that the shrinking step does not have an adverse effect on Δ_t so long as f_{t+1} remains outside the interior of $\mathcal{B}_U$.

Next, we prove a looser bound, compared to the bound provided by Lemma 4.1, which holds for all $t \in J$ and in particular for $t \in J_2$.

LEMMA 4.2. *Let g and f be two functions in $\mathcal{H}_K$. Then, for any ϕ in $(0, 1]$ the following bound holds,*

$$\|f - g\|^2 - \|\phi f - g\|^2 \geq \|g\|^2(\phi - 1) \quad .$$

Proof. As in Eq. (4.5), the left-hand side in the statement of the Lemma can be rewritten as,

$$\|f - g\|^2 - \|\phi f - g\|^2 \;=\; (1 - \phi^2)\|f\|^2 - 2(1 - \phi)\langle f, g\rangle \quad .$$

We now use the elementary fact that for any $u, v \in \mathcal{H}_K$, $\|u - v\|^2 \geq 0$ which can be rewritten as $\|u\|^2 - 2\langle u, v\rangle \geq -\|v\|^2$. Setting $u = \sqrt{1 - \phi^2}\, f$ and $v = \sqrt{\frac{1-\phi}{1+\phi}}\, g$, this inequality becomes

$$(1 - \phi^2)\|f\|^2 - 2(1 - \phi)\langle f, g\rangle \;\geq\; -\frac{1 - \phi}{1 + \phi}\|g\|^2 \quad .$$

Combining the above inequality with the fact that $1 + \phi \geq 1$ proves the bound. □

Finally, we focus on rounds from J_3.

LEMMA 4.3. *Let $U > 0$ and $0 < \phi \leq 1$ be scalars and let g and f be two functions in $\mathcal{H}_K$ such that $\|g\| \leq U$, $\|f\| > U$ and $\|\phi f\| < U$. Then,*

$$\|f - g\|^2 - \|\phi f - g\|^2 \;\geq\; \|g\|^2 \left(\frac{\phi\|f\|}{U} - 1\right) \quad .$$

Proof. Defining $\nu = U/\|f\|$, we can rewrite the left-hand side of our claim as

$$\left(\|f - g\|^2 - \|\nu f - g\|^2\right) + \left(\|\nu f - g\|^2 - \|\phi f - g\|^2\right) \quad .$$

Since $0 < \nu < 1$ and $\|\nu f\| = U$, we can use Lemma 4.1 to lower-bound the first term above by 0. Similarly, we can use Lemma 4.2 to lower-bound the second term above by $-\|g\|^2\left(1 - \frac{\phi}{\nu}\right)$. Summing the two bounds we get

$$\|f - g\|^2 - \|\phi f - g\|^2 \;\geq\; -\|g\|^2\left(1 - \frac{\phi}{\nu}\right) \;=\; \|g\|^2\left(\frac{\phi\|f\|}{U} - 1\right) \quad ,$$

which proves the lemma. □

Combining Lemmas 4.1, 4.2, and 4.3, and recalling that $\beta_t = \|f'_t - g\|^2 - \|f_{t+1} - g\|^2$, we obtain the following lower-bound

$$\sum_{t\in J} \beta_t \;\geq \|g\|^2 \left(\sum_{t\in J_2} (\phi_t - 1) \;+\; \sum_{t\in J_3} \Big(\frac{\phi_t \|f'_t\|}{U} - 1 \Big) \right) \quad .$$

This bound can be restated as follows,

$$\sum_{t\in J} \beta_t \;\geq \|g\|^2 \sum_{t\in J} (\Phi_t - 1) \quad \text{where} \quad \Phi_t \;=\; \begin{cases} 1 & \text{if } t \in J_1 \\ \phi_t & \text{if } t \in J_2 \\ \frac{\phi_t \|f'_t\|}{U} & \text{if } t \in J_3 \end{cases} \quad . \tag{4.6}$$

Using the inequality $x - 1 \geq \log(x)$, we obtain the following corollary.

COROLLARY 4.4. *Let g be a function in $\mathcal{H}_K$ such that $\|g\| \leq U$, where $U \geq 0$. Let β_t be as defined in Eq. (4.2) and Φ_t as defined in Eq. (4.6). Then it holds that*

$$\sum_{t\in J} \beta_t \;\geq\; \|g\|^2 \log \left(\prod_{t\in J} \Phi_t \right) \quad .$$

Repeating the analysis of the kernel Perceptron, we can lower bound $\sum_{t\in J} \alpha_t \geq M - 2\sum_{t=1}^{T} \ell_t^\star$ (as in Eq. (3.3)) and upper bound $\sum_{t\in J} (\alpha_t + \beta_t) \leq \|g\|^2$. Combining these two inequalities with the result from Corollary 4.4 gives

$$M \;\leq\; \|g\|^2 \left(1 - \log \Big(\prod_{t\in J} \Phi_t \Big) \right) + 2 \sum_{t=1}^{T} \ell_t^\star \quad .$$

The above mistake bound can be applied to any concrete strategy of choosing the shrinking coefficient ϕ_t. In the next section, we combine elements from the analysis of the Remove-Oldest Perceptron and the Shrinking Perceptron to derive the Forgetron algorithm, our first online algorithm on a budget for which we prove a mistake bound.

5. The Forgetron Algorithm. In this section we present the Forgetron algorithm, which combines the removal and shrinking techniques presented in the previous sections. The result is a provably correct online learning algorithm on a fixed budget. The main challenge in combining the two techniques revolves around the choice of the shrinking coefficients $\phi_1, \ldots, \phi_T$. On one hand, the shrinking step must be aggressive enough to attenuate the contribution of old active examples to the online hypothesis. On the other hand, an overly aggressive shrinking policy could damage the accuracy of our algorithm. Concretely, we show that the following choice of ϕ_t successfully balances this tradeoff:

$$\phi_t \;=\; \min \left\{ (B+1)^{-\frac{1}{2(B+1)}} \,,\, \frac{U}{\|f'_t\|} \right\} \quad , \tag{5.1}$$

where $f'_t = f_t + y_t K(\mathbf{x}_t, \cdot)$ and

$$U \;=\; \frac{1}{4} \sqrt{\frac{B+1}{\log(B+1)}} \quad . \tag{5.2}$$

Although this simple choice of ϕ_t enables us to prove a formal mistake bound, we note that it has some deficiencies, which we discuss at the end of this section. In the next section, we describe a refined mechanism for choosing ϕ_t, which overcomes these deficiencies.

The Forgetron algorithm initializes I_1 to be the empty set, which implicitly sets f_1 to be the zero function. If a prediction mistake occurs on round t, namely, $y_t f_t(\mathbf{x}_t) \leq 0$, a three step update is performed. The first step is the Perceptron update, which inserts the index t into the active set. We denote the resulting active set by I'_t and the resulting hypothesis by

$$f'_t \;=\; f_t(\mathbf{x}) + y_t K(\mathbf{x}_t, \cdot) \quad . \tag{5.3}$$

The second step is the shrinking step, which sets

$$f''_t \;=\; \phi_t f'_t \quad , \tag{5.4}$$

where $\phi_t \in (0, 1]$ is the shrinking coefficient. The last step of the update is the removal step: if the budget constraint is violated we remove the oldest element from the active set. Put more formally, if $|I'_t| > B$ we set $I_{t+1} = I'_t \setminus \{r_t\}$ where $r_t = \min I'_t$ and otherwise, if $|I'_t| \leq B$, we set $I_{t+1} = I_t$. Following the notation established in the previous section, we can rewrite f_t as

$$f_t \;=\; \sum_{i \in I_t} y_i \, \sigma_{i,t} \, K(\mathbf{x}_i, \cdot) \qquad \text{where} \qquad \sigma_{i,t} \;=\; \prod_{j \in I_{t-1} \,\wedge\, j \geq i} \phi_j \quad .$$

The pseudo-code of the Forgetron algorithm is given in Fig. 5.1.

We now turn to the analysis of the Forgetron algorithm. Recall that our goal is to prove a mistake bound similar to that of the Perceptron (see Thm. 2.1), relative to any competitor g from the set $\{g \in \mathcal{H}_K : \|g\| \leq U\}$. To gain some intuition into our proof technique, assume that g attains a zero loss on every example from the input sequence, that is, $y_t \, g(\mathbf{x}_t) \geq 1$ for all t. As in the proof of Thm. 2.1, we prove a mistake bound for the Forgetron by tracking the dynamics of $\|f_t - g\|^2$. We informally refer to $\|f_t - g\|^2$ as our instantaneous distance from the competitor g. Initially, $f_1 \equiv 0$ and therefore $\|f_1 - g\|^2 = \|g\|^2$. For rounds on which the Forgetron makes a prediction mistake, we first perform the Perceptron update and obtain f'_t. From Lemma 3.1 we know that $\|f_t - g\|^2 - \|f'_t - g\|^2 \geq 2 y_t g(\mathbf{x}_t) - 1 \geq 1$, namely, the Perceptron update moves our classifier closer to g by at least one unit. Next, we perform the shrinking and removal steps. These steps might increase the distance between our classifier and g. Suppose that we could show that the deviation caused by these two steps is at most a half. Then overall, after performing the three step update, the distance between our classifier and g decreases by at least a half. Therefore, after M prediction mistakes, the distance to g decreases by at least $\frac{1}{2}M$. Using the facts that the initial distance to g is $\|g\|^2$ and the final distance cannot be negative, we conclude that $\|g\|^2 - \frac{1}{2}M \geq 0$ which gives us a bound on M. Therefore, to obtain a mistake bound, we must bound the total amount by which the shrinking and removal steps increase our distance to g. We now formalize this intuition and prove the following relative mistake bound.

THEOREM 5.1. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples such that $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t. Assume that this sequence is presented to the Forgetron algorithm with a budget parameter $B \geq 83$ and with ϕ_t defined as in Eq. (5.1). Let g be a function in $\mathcal{H}_K$ such that $\|g\| \leq U$, where U is given by Eq. (5.2), and define*

INPUT: symmetric positive semidefinite kernel $K(\cdot,\cdot)$; budget $B > 0$

INITIALIZE: $I_1 = \emptyset$; $f_1 \equiv 0$; $U = \frac{1}{4}\sqrt{\frac{B+1}{\log(B+1)}}$

For $t = 1, 2, \ldots$
receive an instance $\mathbf{x}_t$
predict $\mathrm{sign}(f_t(\mathbf{x}_t))$
receive correct label y_t
If $y_t f_t(\mathbf{x}_t) > 0$
set $I_{t+1} = I_t$
and $\forall (i \in I_t)$ set $\sigma_{i,t+1} = \sigma_{i,t}$
Else
(1) set $I'_t = I_t \cup \{t\}$
// *define* $f'_t = f_t + y_t K(\mathbf{x}, \cdot)$
(2) set $\phi_t = \min\{\, (B+1)^{-\frac{1}{2(B+1)}} \,,\, U/\|f'_t\| \,\}$
set $\sigma_{t,t+1} = \phi_t$ and $\forall (i \in I_t)$ set $\sigma_{i,t+1} = \phi_t \, \sigma_{i,t}$
// *define* $f''_t = \phi_t f'_t$
(3) **If** $|I'_t| \leq B$
set $I_{t+1} = I'_t$
Else
define $r_t = \min I_t$
set $I_{t+1} = I'_t \setminus \{r_t\}$
define $f_{t+1} = \sum_{i \in I_{t+1}} \sigma_{i,t+1} \, y_i \, K(\mathbf{x}_i, \cdot)$

FIG. 5.1. *The basic Forgetron algorithm.*

$\ell^\star_t = \ell\big(g; (\mathbf{x}_t, y_t)\big)$. *Then, the number of prediction mistakes made by the Forgetron on this sequence is at most*

$$M \;\leq\; 2\,\|g\|^2 \;+\; 4 \sum_{t=1}^{T} \ell^\star_t \quad .$$

Before proving this theorem, we must quantify the negative effect of the shrinking and removal steps on our mistake bound. As before, let J denote the set of rounds on which the Forgetron makes a prediction mistake and for every $t \in J$ define Φ_t as in Eq. (4.6). The role played by Φ_t in our analysis below is similar to its role in the analysis of the shrinking Perceptron, in Sec. 4. Namely, Φ_t bounds the effect of the shrinking step on our mistake bound. Furthermore, let t be a round in J on which $|I_t| = B$ and let r_t denote the index of the example which is removed from the active set on that round. Recall the definition of the function Ψ in Eq. (3.12) and define

$$\Psi_t \;=\; \begin{cases} \Psi\Big(\sigma_{r_t,t+1} \;,\; y_{r_t} f''_t(\mathbf{x}_{r_t})\Big) & \text{if } t \in J \;\wedge\; |I_t| = B \\ 0 & \text{otherwise} \end{cases} \quad . \tag{5.5}$$

It should come as no surprise that the function Ψ plays a role in the analysis of the removal step of the Forgetron update, similar to the role it played in our analysis of the remove-oldest Perceptron, in Sec. 3. The following lemma formalizes the relationship between Φ_t, Ψ_t and the number of mistakes made by the Forgetron.

LEMMA 5.2. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples such that $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t and assume that this sequence is presented to the Forgetron algorithm. Let Φ_t and Ψ_t be as defined in Eq. (4.6) and Eq. (5.5) respectively. Then, the following bound holds for any $g \in \mathcal{H}_K$,*

$$M - \left(\|g\|^2 \sum_{t \in J} \log(1/\Phi_t) + \sum_{t \in J} \Psi_t \right) \;\leq\; \|g\|^2 + 2 \sum_{t \in J} \ell_t^\star \quad .$$

Proof. For each t define $\Delta_t = \|f_t - g\|^2 - \|f_{t+1} - g\|^2$. As in our previous proofs, we prove the lemma by bounding $\sum_{t=1}^{T} \Delta_t$ from above and from below. First note again that $\sum_t \Delta_t$ is a telescopic sum, which collapses to, $\|f_1 - g\|^2 - \|f_{T+1} - g\|^2$. Using the facts that $\|f_{T+1} - g\|^2 \geq 0$ and that $f_1 \equiv 0$, we obtain the upper bound

$$\sum_{t=1}^{T} \Delta_t \;\leq\; \|g\|^2 \quad . \tag{5.6}$$

Next we show a lower bound on $\sum_t \Delta_t$. On rounds where the Forgetron makes a correct prediction, we have that $f_{t+1} = f_t$ and thus

$$\sum_{t=1}^{T} \Delta_t = \sum_{t \in J} \Delta_t \quad . \tag{5.7}$$

Next, we rewrite Δ_t as a sum of three terms for rounds on which the Forgetron makes a mistake,

$$\Delta_t \;=\; \underbrace{\|f_t \text{-} g\|^2 - \|f'_t \text{-} g\|^2}_{\alpha_t} + \underbrace{\|f'_t \text{-} g\|^2 - \|f''_t \text{-} g\|^2}_{\beta_t} + \underbrace{\|f''_t \text{-} g\|^2 - \|f_{t+1} \text{-} g\|^2}_{\gamma_t} \quad , \tag{5.8}$$

where f'_t and f''_t are defined in Eq. (5.3) and Eq. (5.4) respectively. Summing over $t \in J$ and using Eq. (3.10) we get that

$$\sum_{t \in J} \Delta_t \;=\; \sum_{t \in J} \alpha_t + \sum_{t \in J} \beta_t + \sum_{t \in J} \gamma_t \;=\; \sum_{t \in I_{T+1}} \alpha_t + \sum_{t \in J : |I_t| = B} (\alpha_{r_t} + \gamma_t) + \sum_{t \in J} \beta_t \quad . \tag{5.9}$$

We now bound each of the summands in the above equation. First, we use Lemma 3.1 and Eq. (3.11) to get that

$$\sum_{t \in I_{T+1}} \alpha_t \;\geq\; \sum_{t \in I_{T+1}} (1 - 2\ell_t^\star) \quad . \tag{5.10}$$

Recall that $f'_{r_t} = f_{r_t} + y_{r_t} K(\mathbf{x}_{r_t}, \cdot)$ and in addition we can rewrite f_{t+1} as $f''_t - \sigma_{r_t, t+1} y_{r_t} K(\mathbf{x}_{r_t}, \cdot)$. Using Lemma 3.2 with $f = f_{r_t}$, $f' = f''_t$, and $\lambda = \sigma_{r_t, t+1}$ we get that for any $t \in J$ for which $|I_t| = B$ we have that

$$\alpha_{r_t} + \gamma_t \;\geq\; 1 - 2\ell_{r_t}^\star - \Psi_t \quad .$$

Combining the above with Eq. (5.10) gives

$$\sum_{t \in I_{T+1}} \alpha_t + \sum_{t \in J : |I_t| = B} (\alpha_{r_t} + \gamma_t) \;\geq\; \sum_{t \in I_{T+1}} (1 - 2\ell_t^\star) \;+\; \sum_{t \in J : |I_t| = B} \left(1 - 2\ell_{r_t}^\star - \Psi_t\right) \quad .$$

Note that for each $t \in J$ we have that either $t \in I_{T+1}$ or there exists $i \in J$ for which $|I_i| = B$ and $r_i = t$. In addition, Ψ_t is defined to be zero if on round t we do not remove any element from the active set. Therefore, we can further write

$$\sum_{t \in I_{T+1}} \alpha_t + \sum_{t \in J : |I_t| = B} (\alpha_{r_t} + \gamma_t) \;\geq\; M - 2 \sum_{t \in J} \ell_t^\star - \sum_{t \in J} \Psi_t \quad . \tag{5.11}$$

Next, we bound $\sum_t \beta_t$ using corollary 4.4,

$$\sum_{t \in J} \beta_t \;\geq\; \|g\|^2 \sum_{t \in J} \log(\Phi_t) \;=\; -\|g\|^2 \sum_{t \in J} \log(1/\Phi_t) \quad . \tag{5.12}$$

Using Eq. (5.11) and Eq. (5.12) in Eq. (5.9) yields,

$$\sum_{t \in J} \Delta_t \;\geq\; M - 2 \sum_{t \in J} \ell_t^\star - \|g\|^2 \sum_{t \in J} \log(1/\Phi_t) - \sum_{t \in J} \Psi_t \quad .$$

Combining the above with Eq. (5.6) and Eq. (5.7) gives

$$M - \left(\|g\|^2 \sum_{t \in J} \log(1/\Phi_t) + \sum_{t \in J} \Psi_t \right) \;\leq\; \|g\|^2 + 2 \sum_{t \in J} \ell_t^\star \quad .$$

This concludes the proof. □

Lemma 5.2 bounds the total damage to our mistake bound due to the shrinking and removal steps. To prove Thm. 5.1, we show that our choice of the shrinking coefficient in Eq. (5.1) ensures that the term $\left(\|g\|^2 \sum_{t \in J} \log(1/\Phi_t) + \sum_{t \in J} \Psi_t\right)$ is well-behaved. First, we prove an upper bound on $\|g\|^2 \sum_{t \in J} \log(1/\Phi_t)$, the negative effect due to the shrinking step of the Forgetron update.

LEMMA 5.3. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples presented to the Forgetron algorithm with ϕ_t defined as in Eq. (5.1). Let J denote the online iterations on which the Forgetron algorithm makes a prediction mistake and let $M = |J|$. Let g be any function in $\mathcal{H}_K$ with $\|g\| \leq U$, where U is given in Eq. (5.2), and let Φ_t be as defined in Eq. (4.6). Then*

$$\|g\|^2 \sum_{t \in J} \log(1/\Phi_t) \;\leq\; \frac{M}{32} \quad .$$

Proof. We begin the proof by showing that

$$\Phi_t \;\geq\; (B+1)^{-\frac{1}{2(B+1)}} \quad , \tag{5.13}$$

for all $t \in J$. If $t \in J_1$ then $\Phi_t = 1$, which is clearly greater than $(B+1)^{-\frac{1}{2(B+1)}}$. If $t \in J_2$ then Φ_t is defined to equal ϕ_t. It follows from the definition of J_2 in Eq. (4.3) that $U \geq \|f'_t\|$ and therefore

$$(B+1)^{-\frac{1}{2(B+1)}} \;\leq\; 1 \;\leq\; \frac{U}{\|f'_t\|} \quad .$$

Referring back to Eq. (5.1), we get that $\phi_t = (B+1)^{-\frac{1}{2(B+1)}}$ and therefore Eq. (5.13) holds in this case as well. Finally, if $t \in J_3$ then Φ_t is defined to equal $\phi \|f'_t\|/U$. From the definition of J_3 in Eq. (4.3), we have that $U < \|f'_t\|$ and therefore $\Phi_t > \phi_t$. If

$\phi_t = (B+1)^{-\frac{1}{2(B+1)}}$ then Eq. (5.13) holds trivially. Otherwise, $\phi_t = U/\|f'_t\|$, $\Phi_t > 1$ and once again Eq. (5.13) holds. We can now rewrite Eq. (5.13) as

$$\log\left(\frac{1}{\Phi_t}\right) \;\leq\; \frac{\log(B+1)}{2(B+1)} \quad .$$

Combining the above with the assumption that $\|g\|^2 \leq U^2$ and using the definition of U in Eq. (5.2) result in

$$\|g\|^2 \log(1/\Phi_t) \;\leq\; \frac{B+1}{16\log(B+1)}\,\frac{\log(B+1)}{2(B+1)} \;=\; \frac{1}{32} \quad .$$

Summing both sides of the above over all $t \in J$ proves the lemma. □

Next, we prove that our choice of ϕ_t in Eq. (5.1) guarantees an upper bound on $\sum_{t\in J}\Psi_t$, the negative effect due to the removal step of the Forgetron update.

LEMMA 5.4. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples such that $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t. Assume that this sequence is presented to the Forgetron algorithm with a budget parameter $B \geq 83$ and with ϕ_t defined as in Eq. (5.1). Let J denote the online iterations on which the Forgetron algorithm makes a prediction mistake and let $M = |J|$. Let g be a function in $\mathcal{H}_K$ such that $\|g\| \leq U$, where U is given in Eq. (5.2), and let Ψ_t be as defined in Eq. (3.12). Then,*

$$\sum_{t\in J}\Psi_t \;\leq\; \frac{15M}{32} \quad .$$

Proof. Let t be an online round in J, and recall that I_t is the active set of the Forgetron algorithm on round t and that B is the predefined memory budget. If $|I_t| < B$ then $\Psi_t = 0$. Otherwise, Ψ_t equals

$$\Psi_t = \sigma^2_{r_t,t+1} + 2\,\sigma_{r_t,t+1} - 2\,\sigma_{r_t,t+1}\,y_{r_t}\,f'_t(\mathbf{x}_{r_t}) \quad . \tag{5.14}$$

The definition of ϕ_t given in Eq. (5.1) implies that $\phi_t \leq (B+1)^{-1/(2(B+1))}$ for all $t \in J$. Since the oldest element in the active set, whose index is r_t, is scaled $B+1$ times before it is removed from the active set, we get

$$\sigma_{r_t,t+1} \;\leq\; \left((B+1)^{-\frac{1}{2(B+1)}}\right)^{B+1} \;=\; \frac{1}{\sqrt{B+1}} \quad . \tag{5.15}$$

Next, we use the Cauchy-Schwartz inequality to bound the term $-y_{r_t} f'_t(\mathbf{x}_{r_t})$ by $\|f'_t\|\,\|K(\mathbf{x}_{r_t}, \cdot)\|$. The definition of ϕ_t implies that $\|f'_t\| \leq U$, and we assumed $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t, so $-y_{r_t} f'_t(\mathbf{x}_{r_t}) \leq U$. Plugging this inequality and the inequality in Eq. (5.15) into Eq. (5.14) gives

$$\Psi_t \;\leq\; \frac{1}{B+1} + \frac{2}{\sqrt{B+1}} + \frac{2\,U}{\sqrt{B+1}} \quad .$$

Using the definition of U from Eq. (5.2), we have

$$\Psi_t \;\leq\; \frac{1}{B+1} + \frac{2}{\sqrt{B+1}} + \frac{1}{2\sqrt{\log(B+1)}} \quad . \tag{5.16}$$

The right hand side of the above inequality decreases monotonically with B and is at most $15/32$ for $B \geq 83$. Thus, $\sum_{t\in J}\Psi_t \leq \frac{15M}{32}$. □

Proof. [Proof of Thm. 5.1] From Lemma 5.2 we have

$$M - \left(\|g\|^2 \sum_{t \in J} \log(1/\Phi_t) + \sum_{t \in J} \Psi_t \right) \;\leq\; \|g\|^2 + 2 \sum_{t \in J} \ell_t^\star \quad .$$

Plugging the bounds in Lemma 5.3 and Lemma 5.4 into the above inequality gives

$$M - \left(\frac{M}{32} + \frac{15M}{32} \right) \;\leq\; \|g\|^2 + 2 \sum_{t \in J} \ell_t^\star \quad .$$

Multiplying both sides of the above inequality by 2 provides the desired mistake bound. □

We have shown that the choice of ϕ_t in Eq. (5.1) indeed results in a provably correct learning algorithm on a budget. However, this definition of ϕ_t suffers from several drawbacks. First and foremost, the resulting algorithm performs poorly in practice. In the next section, we present the self-tuned Forgetron, which uses a refined shrinking mechanism, and significantly outperforms the algorithm presented in this section (see experimental results in Sec. 8). Another problem with the definition of ϕ_t in Eq. (5.1) is that it forces $\|f_t''\|$ to be at most U. We used this property in the proof above to bound $-y_{r_t} f_t''(\mathbf{x}_{r_t})$, which in turn provided us with an upper bound on Ψ_t. In practice, it is often the case that r_t can be safely removed from the active set without any shrinking, and the norm of f_t'' can be allowed to grow beyond U. The refined shrinking mechanism of the self-tuned Forgetron uses the actual values of $\Psi_1, \ldots, \Psi_t$ to define ϕ_t, and does not explicitly use U.

6. The Self-Tuned Forgetron. In Sec. 5 we introduced the Forgetron framework and proposed a simple definition of the shrinking coefficients in Eq. (5.1). Besides constants, which do not change from round to round, the definition in Eq. (5.1) depends solely on $\|f_t'\|$. Moreover, it makes explicit use of the upper bound U. In this section we propose an improved shrinking scheme which does not rely on the knowledge of U. We name the resulting algorithm the *self-tuned Forgetron*. The main principle which we follow in the derivation of the self-tuned Forgetron is to apply the gentlest possible shrinking step. For example, if we are fortunate, and the damage from the removal step happens to be small without applying any shrinking, then our improved shrinking scheme will set $\phi_t = 1$. On such rounds, the self-tuned Forgetron algorithm update reduces back to the remove-oldest-Perceptron update discussed in Sec. 3.

Recall that in our analysis in Sec. 5, Lemma 5.3 provided an upper bound on $\|g\|^2 \sum_{t \in J} \log(1/\Phi_t)$ and Lemma 5.4 provided an upper bound on $\sum_{t \in J} \Psi_t$. Together with Lemma 5.2, these upper bounds were sufficient to prove the Forgetron mistake bound in Thm. 5.1. On every update, the self-tuned Forgetron chooses the gentlest shrinking that still ensures that the bounds in Lemma 5.3 and Lemma 5.4 still hold, and that our mistake bound remains valid. More formally, given an input sequence of examples of length T, define M_t to be the number of prediction mistakes made by our algorithm on rounds $\{1, 2, \ldots, t\}$. On round t, if an online update is invoked, the self-tuned Forgetron chooses the shrinking coefficient ϕ_t to be the largest number in $(0, 1]$ that satisfies the condition

$$\forall t, \quad \sum_{i \in J \,:\, i \leq t} \Psi_i \;\leq\; \frac{15}{32} M_t \quad , \tag{6.1}$$

is met. More concretely, define

$$Q_t \;=\; \sum_{i \in J \,:\, i<t} \Psi_i \quad .$$

Let $t \in J$ be an index of a round on which the Forgetron makes a prediction mistake and is required to remove an example from the active set ($|I_t| = B$). The t'th constraint from Eq. (6.1) can be rewritten as

$$\Psi\big(\sigma_{r_t,t}\,\phi_t \;,\; y_{r_t}\phi_t f'_t(\mathbf{x}_{r_t})\big) \;+\; Q_t \;\leq\; \frac{15}{32}\,M_t \quad .$$

The self-tuned Forgetron sets ϕ_t to be the maximal value in $(0,1]$ for which the above inequality holds, namely,

$$\phi_t = \max\left\{\phi \in (0,1] \;:\; \Psi\big(\sigma_{r_t,t}\,\phi \;,\; y_{r_t}\phi f'_t(\mathbf{x}_{r_t})\big) + Q_t \;\leq\; \frac{15}{32}M_t\right\} \quad . \tag{6.2}$$

Note that Ψ is a quadratic function in ϕ and thus the optimal value of ϕ_t can be found analytically. Simple algebraic manipulations yield that

$$\phi_t \;=\; \begin{cases} \min\left\{1, \frac{-b+\sqrt{d}}{2a}\right\} & \text{if } a>0 \;\vee\; (a<0 \;\wedge\; d>0 \;\wedge\; \frac{-b-\sqrt{d}}{2a} > 1) \\ \min\{1, -c/b\} & \text{if } a = 0 \\ 1 & \text{otherwise} \end{cases} \quad , \tag{6.3}$$

where

$$\begin{aligned} a &= \sigma^2_{r_t,t} - 2\,\sigma_{r_t,t}\,y_{r_t}f'_t(\mathbf{x}_{r_t}), \quad b = 2\,\sigma_{r_t,t}, \\ c &= Q_t - \tfrac{15}{32}M_t, \quad \text{and} \quad d = b^2 - 4ac \quad . \end{aligned} \tag{6.4}$$

The pseudo-code of the self-tuned Forgetron is given in Fig. 6.1.

By construction, the effect of the removal step is upper bounded by

$$\sum_{t \in J} \Psi_t \;\leq\; \frac{15}{32}M \quad . \tag{6.5}$$

This fact replaces Lemma 5.4. Therefore, to apply the same proof technique as in the previous section, it now suffices to show that the bound

$$\|g\|^2 \sum_t \log(1/\Phi_t) \leq \frac{1}{32}M \quad , \tag{6.6}$$

still holds. To prove the above inequality, we require the following lemma.

LEMMA 6.1. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples such that $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t and assume that this sequence is presented to the self-tuned Forgetron with a budget parameter $B \geq 83$. Let J denote the set of rounds on which the algorithm makes a prediction mistake, let ϕ_t as in Eq. (6.2) and let Φ_t be as defined in Eq. (4.6). Finally, let t be a round in J such that $\Phi_t < 1$. Then,*

$$\Phi_t\,\sigma_{r_t,t} \;\geq\; \frac{1}{\sqrt{B+1}} \quad .$$

INPUT: symmetric positive semidefinite kernel $K(\cdot,\cdot)$; budget $B > 0$

INITIALIZE: $I_1 = \emptyset$; $f_1 \equiv 0$; $Q_1 = 0$; $M_0 = 0$

For $t = 1, 2, \ldots$

receive an instance $\mathbf{x}_t$

predict $\mathrm{sign}(f_t(\mathbf{x}_t))$

receive correct label y_t

If $y_t f_t(\mathbf{x}_t) > 0$

set $I_{t+1} = I_t$, $Q_{t+1} = Q_t$, $M_t = M_{t-1}$,
and $\forall (i \in I_t)$ set $\sigma_{i,t+1} = \sigma_{i,t}$

Else

set $M_t = M_{t-1} + 1$

(1) set $I'_t = I_t \cup \{t\}$

// *define* $f'_t = f_t + y_t K(\mathbf{x}, \cdot)$

If $|I'_t| \leq B$

set $I_{t+1} = I'_t$, $Q_{t+1} = Q_t$, $\sigma_{t,t} = 1$,
and $\forall (i \in I_{t+1})$ set $\sigma_{i,t+1} = \sigma_{i,t}$

Else

(2) define $r_t = \min I_t$

define a, b, c, d as in Eq. (6.4) and set ϕ_t as in Eq. (6.3)

set $\sigma_{t,t+1} = \phi_t$ and $\forall (i \in I_t)$ set $\sigma_{i,t+1} = \phi_t \, \sigma_{i,t}$

set $Q_{t+1} = \Psi\big(\sigma_{r_t,t+1} \ , \ y_{r_t} f''_t(\mathbf{x}_{r_t})\big) + Q_t$

// *define* $f''_t = \phi_t f'_t$

(3) set $I_{t+1} = I'_t \setminus \{r_t\}$

define $f_{t+1} = \sum_{i \in I_{t+1}} \sigma_{i,t+1} \, y_i \, K(\mathbf{x}_i, \cdot)$

FIG. 6.1. *The self-tuned Forgetron algorithm.*

Proof. Define

$$\phi' \ = \ \min \left\{ 1 \ , \ \frac{U}{\|f'_t\|} \ , \ \frac{1}{\sigma_{r_t,t} \sqrt{B+1}} \right\} \ .$$

This definition implies that: (i) $\phi' \in (0, 1]$. (ii) $\phi' \|f'_t\| \leq U$ and therefore, using the Cauchy-Schwartz inequality, $\phi' f'_t(\mathbf{x}_{r_t}) \leq U$. (iii) $\sigma_{r_t,t} \, \phi' \leq 1/\sqrt{B+1}$. Therefore,

$$(\sigma_{r_t,t} \, \phi')^2 + 2\sigma_{r_t,t} \, \phi' \, (1 - y_{r_t} \phi' f'_t(\mathbf{x}_{r_t})) \ \leq \ \frac{1}{B+1} + \frac{2}{\sqrt{B+1}}(1+U) \ .$$

The left-hand side of the above equals $\Psi\big(\sigma_{r_t,t+1}, \ y_{r_t} f''_t(\mathbf{x}_{r_t})\big)$. Using the definition of U we get that

$$\Psi\big(\sigma_{r_t,t}\phi' \ , \ y_{r_t}\phi' f'_t(\mathbf{x}_{r_t})\big) \ \leq \ \frac{1}{B+1} + \frac{2}{\sqrt{B+1}} + \frac{1}{2\sqrt{\log(B+1)}} \ .$$

The right-hand side of the above inequality is at most $\frac{15}{32}$ for $B \geq 83$. In addition, the definition of the self-tuned Forgetron implies that $Q_t \leq \frac{15}{32} M_{t-1}$ for each t. Therefore,

$$\Psi\big(\sigma_{r_t,t}\phi' \ , \ y_{r_t}\phi' f'_t(\mathbf{x}_{r_t})\big) + Q_t \ \leq \ \frac{15}{32} \, M_t \ . \tag{6.7}$$

Since ϕ' is in $(0,1]$ and satisfies Eq. (6.7), and ϕ_t is the largest value which satisfies Eq. (6.7), we get that $\phi_t \geq \phi'$. By the definition of Φ_t in Eq. (4.6) we have $\Phi_t \geq \phi_t$, and therefore $\Phi_t \geq \phi'$. We have therefore reduced our problem to proving $\phi' \sigma_{r_t,t} \geq 1/\sqrt{B+1}$.

The assumption that $\Phi_t < 1$ implies that $\phi' < 1$ as well. We are left with two possibilities, either $\phi' = U/\|f'_t\|$ or $\phi' = \frac{1}{\sigma_{r_t,t}\sqrt{B+1}}$. If $\phi' = U/\|f'_t\|$ then

$$\phi_t \|f'_t\| \;\geq\; \phi' \|f'_t\| \;=\; \frac{U}{\|f'_t\|} \|f'_t\| \;=\; U \quad .$$

Therefore, $t \in J_1$, that is, the norm of the hypothesis after the shrinking step is still as large as U (see also Fig. 4.1). This immediately implies that $\Phi_t = 1$, which stands in contradiction to the assumption that $\Phi_t < 1$. We have thus shown that ϕ' must equal $1/(\sigma_{r_t,t}\sqrt{B+1})$. It therefore holds that $\phi'\sigma_{r_t,t} \geq 1/\sqrt{B+1}$ and this concludes our proof. □

Equipped with the above lemma, we can prove a mistake bound for the self-tuned Forgetron.

THEOREM 6.2. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples such that $K(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t. Assume that this sequence is presented to the self-tuned Forgetron of Fig. 6.1 with a budget parameter $B \geq 83$. Let g be a hypothesis in $\mathcal{H}_K$ such that $\|g\| \leq U$, where U is given by Eq. (5.2), and define $\ell_t^\star = \ell\big(g; (\mathbf{x}_t, y_t)\big)$. Then, the number of prediction mistakes made by the self-tuned Forgetron on this sequence is at most*

$$M \;\leq\; 2\|g\|^2 \;+\; 4\sum_{t=1}^{T} \ell_t^\star \quad .$$

Proof. We follow the proof of Thm. 5.1. The bound in Eq. (6.5) holds by construction and therefore it suffices to show that Eq. (6.6) holds. Since $\|g\| \leq U$, we know that

$$\|g\|^2 \sum_{t\in J} \log(1/\Phi_t) \;\leq\; \frac{B+1}{16\log(B+1)} \sum_{t\in J} \log(1/\Phi_t) \quad .$$

Therefore, to prove that Eq. (6.6) holds it suffices to show that

$$\sum_{t\in J} \log(1/\Phi_t) \;\leq\; \frac{\log(B+1)}{2(B+1)}\, M \quad ,$$

or equivalently that

$$\prod_{t\in J} \Phi_t \;\geq\; (B+1)^{-\frac{M}{2(B+1)}} \quad . \tag{6.8}$$

We prove the above inequality by strong induction on the number of prediction mistakes made by the self-tuned Forgetron. Once again, J denotes the online rounds on which the algorithm made a prediction mistake. First note that if $|J| < B$ then $\phi_t = 1$ for all $t \in J$, in which case the claim is trivial. Therefore, we assume that $|J| \geq B$. Assume that the claim holds for every $J' \subset J$ (which means that $|J'| < M$) and let us prove the claim for J. That is, we need to show that

$$\prod_{t\in J} \Phi_t \;\geq\; (B+1)^{-\frac{|J|}{2(B+1)}} \quad . \tag{6.9}$$

Let $j = \max J$ denote the index of the last element that was inserted into J. If $\Phi_j = 1$ then

$$\prod_{t \in J} \Phi_t \;=\; \prod_{t \in J \setminus \{j\}} \Phi_t \quad .$$

Applying the inductive assumption to the set $J' = J \setminus \{j\} \subset J$ we get that

$$\prod_{t \in J} \Phi_t \;=\; \prod_{t \in J'} \Phi_t \;\geq\; (B+1)^{-\frac{|J'|}{2(B+1)}} \;\geq\; (B+1)^{-\frac{|J|}{2(B+1)}} \quad .$$

Therefore, it is left to show that the claim holds for $\Phi_j < 1$. Recall that I'_j denotes the active set after applying the Perceptron update step and before applying the removal step on round j. Using the inductive assumption on the set $J' = J \setminus I'_j$, we have $|J'| = |J| - (B+1)$ and therefore,

$$\prod_{t \in J} \Phi_t \;=\; \prod_{t \in J'} \Phi_t \prod_{t \in I'_j} \Phi_t \;\geq\; (B+1)^{-\frac{|J|-(B+1)}{2(B+1)}} \prod_{t \in I'_j} \Phi_t \quad . \tag{6.10}$$

Recall that $r_j = \min I'_j$. Using the fact that $\Phi_j \geq \phi_j$ and the definition of $\sigma_{r_j,j}$ we get that

$$\prod_{t \in I'_j} \Phi_t \;\geq\; \Phi_j \prod_{t \in I_j} \phi_t \;=\; \Phi_j \, \sigma_{r_j,j} \quad .$$

From Lemma 6.1 we know that the right-hand side of the above is at least $1/\sqrt{B+1}$. Using this fact in Eq. (6.10) gives,

$$\prod_{t \in J} \Phi_t \;\geq\; (B+1)^{-\frac{|J|-(B+1)}{2(B+1)}} \frac{1}{\sqrt{B+1}} \;=\; (B+1)^{-\frac{|J|}{2(B+1)}} \quad .$$

This concludes our proof. □

7. A Greedy Removal Scheme. The variants of the Forgetron algorithm we discussed so far always remove the oldest element from the active set. The accompanying shrinking step controls the damage due to the removal step. Our approach stands in contrast to earlier online learning algorithms on a budget [3, 12] which focus on choosing which example to remove from the active set and do not take any measures to control the damage due to this removal. While earlier work did not provide any mistake bounds, we would like to build on the intuition conveyed in previous work to devise a greedy removal scheme that may skip the shrinking step when possible.

In this section we describe an extension of the Forgetron framework which allows removal of examples other than the oldest one. Our removal criterion is based on the analysis presented in Sec. 5. Specifically, in Lemma 5.2 we showed that the damage inflicted upon the hypothesis due to the removal step is $\Psi\big(\sigma_{r_t,t}\,\phi_t \;,\; y_{r_t}\phi_t f''_t(\mathbf{x}_{r_t})\big)$. The goal of the shrinking step is to ensure that the total damage due to the removal step is at most $\frac{15}{32}$ times the number of prediction mistakes. According to our analysis, if there exists an example $i \in I_t$ for which

$$\Psi\big(\sigma_{r_t,t} \;,\; y_{r_t} f'_t(\mathbf{x}_{r_t})\big) \;\leq\; \frac{15}{32} \quad , \tag{7.1}$$

then this example can be safely removed from the active set without any shrinking. We therefore employ the following two stage approach. If indeed there exists an index $i \in I_t$ for which Eq. (7.1) holds, then we skip the shrinking step and remove this index from the active set. Otherwise, we perform the self-tuned Forgetron update discussed in the previous section. Formally, define

$$j \;=\; \arg\min i \in I_t \Psi(\sigma_{i,t}\,,\, y_i f'_t(\mathbf{x}_i)) \quad .$$

The example to be removed is set to

$$r_t \;=\; \begin{cases} j & \text{if } \; \Psi(\sigma_{j,t}\,,\, y_j f'_t(\mathbf{x}_j)) \;\leq \frac{15}{32} \\ \min I_t & \text{otherwise} \end{cases} \quad . \tag{7.2}$$

The shrinking coefficient ϕ_t is set as before, namely,

$$\phi_t \;=\; \max\left\{\phi \in (0,1] \;:\; \Psi\big(\sigma_{r_t,t}\,\phi\;,\; y_{r_t}\phi f'_t(\mathbf{x}_{r_t})\big) \;+\; Q_t \;\leq\; \frac{15}{32}M_t\right\} \quad ,$$

where $Q_t = \sum_{i \in J : i<t} \Psi_i$.

The greedy removal scheme entertains the mistake bound proven for the self-tuned Forgetron. To see this, first note that Lemma 5.2 does not assume that $r_t = \min I_t$. In fact, the lemma holds for any choice of $r_t \in I_t$. In particular, Lemma 5.2 holds for the example chosen by the greedy removal scheme. In addition, the inequality $\sum_t \Psi_t \leq \frac{15}{32}M$ holds by construction. Thus, it again suffices to show that

$$\|g\|^2 \sum_t \log(1/\Phi_t) \leq \frac{1}{32}M \quad . \tag{7.3}$$

To prove the above inequality, note that whenever $\Phi_t < 1$ (and thus $\phi_t < 1$) we have that $r_t = \min I_t$. Therefore, the update coincides with the update of the self-tuned Forgetron and the proof of Lemma 6.1 can be repeated verbatim. Moreover, it is immediate to verify that the same reasoning used to prove Thm. 6.2 carries over to the greedy removal scheme. In summary, the bound of Thm. 6.2 also applies to the greedy removal scheme.

8. Experiments. In this section we present experimental results which demonstrate the merits of the Forgetron algorithms. Since the focus of this paper is on the online learning setting, we ran different online algorithms on various datasets and we report the online error for each experiment. The online error is the number of prediction mistakes an algorithm makes on a sequence of examples, divided by the sequence length. Throughout this section we consistently use the online error to assess the performance of the different algorithms.

We compare the performance of our algorithms with the two methods described in [3] and [1], abbreviated by CKS and CG respectively, and with the standard kernel-based Perceptron. The CKS algorithm is a variant of the kernel-based Perceptron, which uses the following heuristic to enforce a strict memory budget. When the budget is exceeded, the algorithm calculates the margin attained by removing each active example from the active set, and then applying the resulting hypothesis to the removed example. The removed example is the one which attains the maximal margin. This removal scheme is similar to the removal scheme described in Sec. 7. The CKS algorithm only guarantees that its removal scheme does not damage the accuracy of the hypothesis when the margin attained by the removed example is greater than one.

$B = p/4$

	Perceptron	F (basic)	F (slf-tuned)	F (greedy)	CKS
MNIST	6.08	35.22	11.25	9.54	17.45
USPS	7.73	40.70	14.88	12.70	18.52
ADULT	20.33	30.44	22.31	24.06	33.48
synth. (5%)	9.56	11.60	9.89	11.84	32.76
synth. (10%)	18.16	20.30	18.38	21.07	41.13

$B = p/2$

	Perceptron	F (basic)	F (slf-tuned)	F (greedy)	CKS
MNIST	6.08	27.05	8.62	7.78	9.02
USPS	7.73	31.95	11.03	9.78	10.26
ADULT	20.33	26.67	21.40	23.70	27.82
synth. (5%)	9.56	10.66	9.70	11.98	20.16
synth. (10%)	18.16	19.10	18.27	21.74	30.05

$B = p$

	Perceptron	F (basic)	F (slf-tuned)	F (greedy)	CKS
MNIST	6.08	16.07	6.08	6.08	6.08
USPS	7.73	20.29	7.73	7.73	7.73
ADULT	20.33	22.06	20.33	20.33	20.33
synth. (5%)	9.56	9.79	9.56	9.56	9.56
synth. (10%)	18.16	18.37	18.16	18.16	18.16

TABLE 8.1

Each of the three tables above corresponds to a different ratio between B, the budget parameter, and p, the size of the active set used by the Perceptron on the respective dataset. For example, the top table sets the B to be a quarter of the number of mistakes suffered by the standard Perceptron algorithm. Each entry in the table gives the average online error attained by the algorithms on each dataset.

If no such example exists in the active set, no formal guarantees are provided. We therefore anticipate that the CKS algorithm would work well when the examples form a separable dataset, but is likely to fail on more difficult, inseparable, datasets.

The CG algorithm is a randomized method for online learning on a budget. When the CG algorithm exceeds its budget, it removes a randomly chosen example from the active set. In all our experiments, we ran the CG algorithm 10 times on each dataset and report the online error averaged over the 10 different runs. A disadvantage of this average-case analysis in the online setting is that in real world online learning problems, we typically run the algorithm over the sequence of examples only once. We discuss this disadvantage in our last experiment below.

In all our experiments, we focus on the self-tuned Forgetron described in Sec. 6 and on the greedy removal Forgetron described in Sec. 7. We also conducted experiments with the basic Forgetron algorithm described in Sec. 5, however its performance was found to be significantly inferior to the other Forgetron variants. This can be attributed to the worst-case definition of the shrinking coefficients employed by the basic Forgetron. Also note that the self-tuned Forgetron and the greedy removal Forgetron are identical to the original Perceptron when the active set used by the Perceptron is less than the budget parameter, while the basic Forgetron is different due to the fixed shrinking coefficient. For clarity, we present the results of the basic Forgetron only in Table 8.1, and not in the graphs in Figures 8.1, 8.2, 8.3, and 8.4.

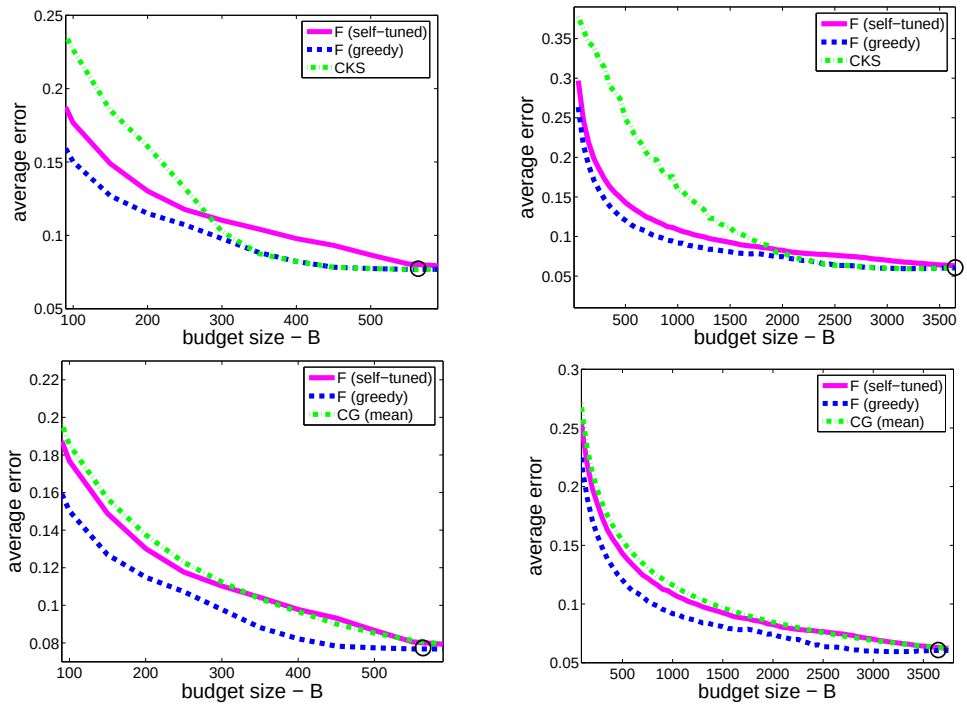

FIG. 8.1. *The average online error of different budget algorithms as a function of the budget B on the USPS dataset (left) and the MNIST dataset (right). The online error of the Perceptron and its budget requirements for each problem are marked with a circle.*

Our first experiment was performed with two standard datasets: the MNIST dataset, which consists of 60,000 training examples, and the USPS dataset, with 10,000 examples. These two datasets are well known and induce relatively easy classification problems. The instances in both datasets are handwritten images of digits, thus each image corresponds to one of the 10 digit classes. We generated 126 binary problems by splitting the 10 labels into two equal-size sets in all possible ways ($\binom{10}{5}/2 = 126$). We report the online error averaged over these 126 problems. We ran the various algorithms with different values of the budget parameter B, using a Gaussian kernel defined as $K(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\frac{1}{2}\|\mathbf{x}_1 - \mathbf{x}_2\|^2)$. The results of these experiments are summarized in Fig. 8.1 and in Table 8.1. Since the standard Perceptron does not take a budget parameter, we mark its accuracy and active set size in Fig. 8.1 with a small circle. All four algorithms perform quite well on these datasets. It is apparent that for both datasets, the greedy removal Forgetron is slightly better than the alternative methods. Comparing the performance of the self-tuned Forgetron and CKS, we note that the former performs better on small budgets while the latter is better on large budgets. It is also apparent that the average online error of the CG method is similar to the online error of the self-tuned Forgetron.

Our next experiment was performed with the census-income (adult) dataset, which consists of 199,523 examples. This dataset is highly non-balanced (only 6.21 percent of the labels are positive). We overcame this problem by randomly generating a balanced subset of this dataset. We repeated this process 10 times, generating 10 different balanced datasets. The results we report were obtained by averaging over

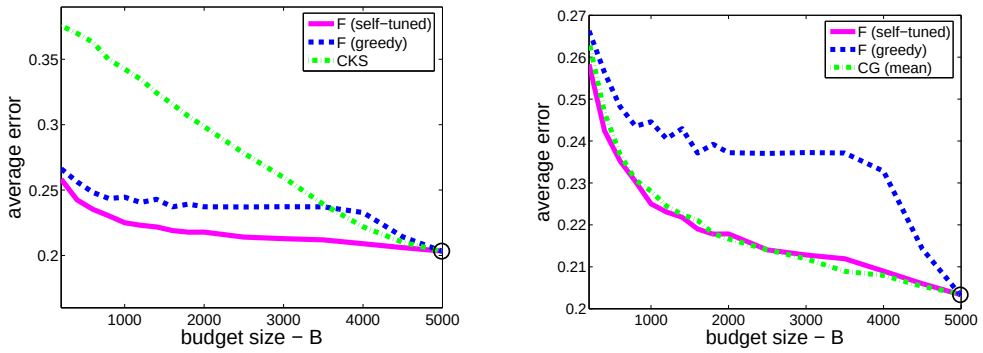

FIG. 8.2. *The average online error of different budget algorithms as a function of the budget B on the census-income (adult) dataset. The average error of the Perceptron and its budget requirements are marked with a circle.*

the 10 different selections. We first ran the Perceptron algorithm on each dataset with a Gaussian kernel. The online error of the Perceptron was approximately 21 percent. We then ran the various budget algorithms on each dataset with different values of B. The results are given in Fig. 8.2 and in Table 8.1. It is apparent that the self-tuned Forgetron and the CG method perform very well on this dataset and outperform both the greedy removal Forgetron algorithm and the CKS method. The performance of the greedy removal Forgetron is also relatively good for small budgets. The relatively poor performance of CKS on this dataset, when B takes small values, may be due to the difficulty of the classification task. As mentioned above, the analysis of CKS is based on the assumption that there always exists an example whose margin, after its removal, is greater than 1. Whenever we are unfortunate, and there is no such example in the active set, the CKS removal step may significantly damage the accuracy of the current hypothesis.

To further investigate the performance of the various algorithms, our last experiment examines the accuracy of the algorithms in the presence of label noise. Recall that the number of active examples used by the basic Perceptron algorithm grows with each prediction mistake. Therefore, we expect the Perceptron algorithm to require a large active set in the presence of noise. As in our previous experiments, we ran the various budget algorithms with a Gaussian kernel. We generated two synthetic datasets as follows. We randomly sampled 5000 positive examples from a two-dimensional Gaussian with a mean vector of $(1, 1)$ and a diagonal covariance matrix with $(0.2, 2)$ as its diagonal. We then sampled 5000 negative examples from a normal distribution with a mean vector of $(-1, -1)$ and the same covariance as before. Finally, we flipped each label with a probability of 0.1 for the first dataset and with a probability of 0.05 for the second dataset, thus introducing two noise rates. We then presented the data to each of the algorithms. We repeated this process for different values of the budget parameter B, ranging from 10 to 2000. We repeated the entire experiment 100 times and averaged the results. The average online error attained by each algorithm for each choice of B is given in Fig. 8.3 and in Table 8.1. The graphs underscore several interesting phenomena. First note that the self-tuned Forgetron and the CG method outperform both the greedy removal Forgetron and the CKS method. In fact, the self-tuned Forgetron and the CG method achieve almost the same accuracy as the vanilla Perceptron algorithm while requiring less than a fifth

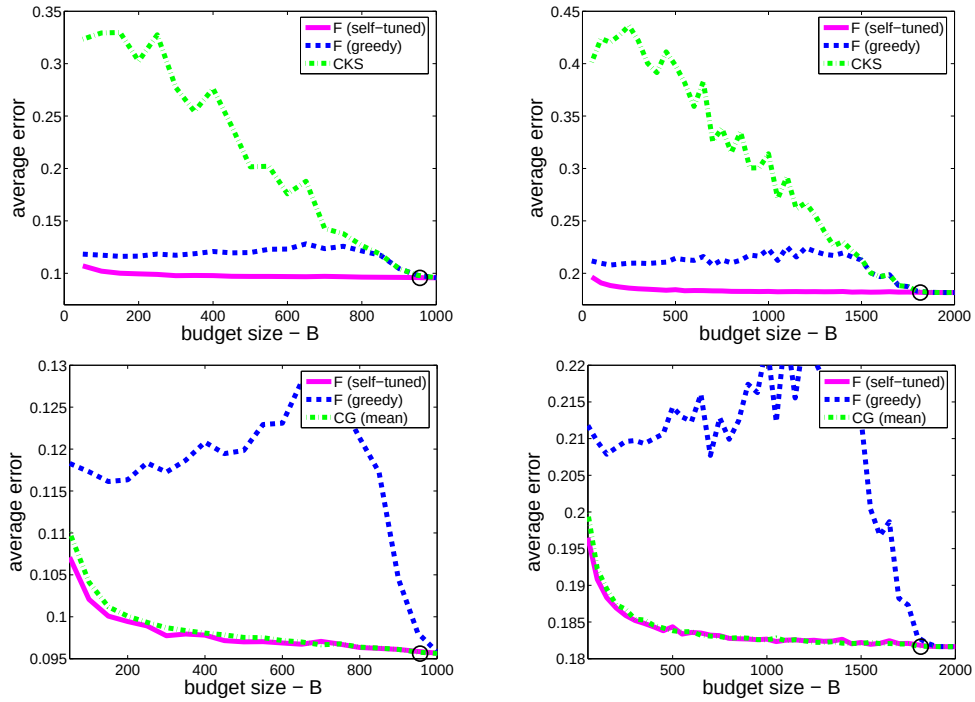

FIG. 8.3. *The average online error of different budget algorithms as a function of the budget B on synthetic datasets with 5% label noise (left) and 10% label noise (right). The average accuracy of the Perceptron and its budget requirements for each problem are marked by a circle.*

of the active set size required by the Perceptron. The ability to obtain a low error with a small budget on this dataset is not surprising as the decision boundary can be described by a small number of examples. The performance of the greedy removal Forgetron is also reasonable. The inferior performance of the CKS method on these datasets may be attributed to the following observation. A mislabeled example in the active set is likely to decrease the accuracy of the classifier. In addition, if these examples are removed from the active set, they are likely to be incorrectly classified by the resulting classifier. Alas, the removal criterion of the CKS method prefers to leave mislabeled examples in the active set. As mislabeled examples start accumulating in the active set, the damage to the classifier's accuracy becomes more pronounced. In contrast, the Forgetron algorithms demote the weight of each example in the active set on each round, thus ensuring that noisy examples do not remain active for a very long period. The removal criterion of the greedy removal Forgetron algorithm is also affected by the above argument. Indeed, we can see that the performance of the greedy removal Forgetron is rather good with small budgets, it worsens as the budget increases and finally it improves again when the budget is large. When the budget is very small, the greedy removal Forgetron cannot find an example $r_t \in I_t$ for which $\Psi \leq \frac{15}{32}$. Thus, the example removed is the oldest example in the active set (see Eq. (7.2)). As the budget increases, there are examples whose margin is greater than 1, so the greedy removal Forgetron removes them without further scaling. As with the CKS algorithm, this removal criterion prefers to leave noisy examples in the active set and we can see deterioration in the performance.

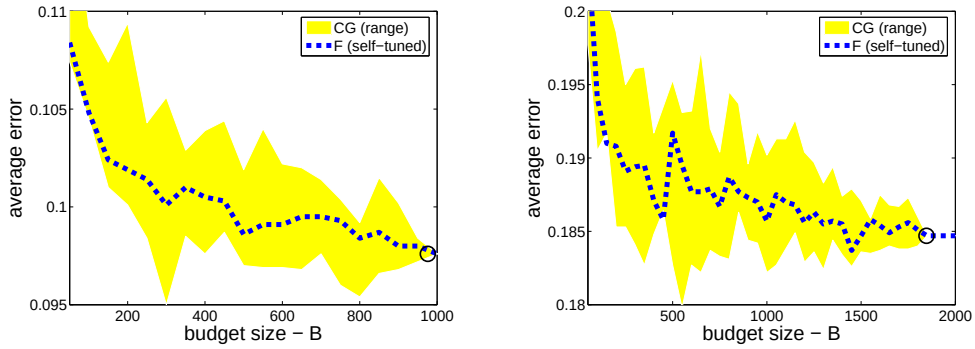

FIG. 8.4. *The online error of the self-tuned Forgetron and the CG algorithm on a single dataset with 5% label noise (left) and on a single dataset with 10% label noise. For the CG algorithm, the range of online errors over 10 runs is given.*

So far, we have calculated the average online error of the CG algorithm, and our experiments indicate that it is similar to the online error of the self-tuned Forgetron. However, the CG algorithm is a randomized method and its performance on individual runs may vary. Recall that the goal of online learning is to accurately predict a sequence of labels that is revealed incrementally as learning proceeds. Once all of the labels have been revealed, the prediction task becomes vacuous. Therefore, it only makes sense to run the online algorithm over the sequence of examples once. In our last experiment, we compared single runs of the CG algorithm with the self-tuned Forgetron. We ran the self-tuned Forgetron on a single dataset with 5% label noise and on a single dataset with 10% label noise, without averaging the results over several datasets. We also ran the CG algorithm 10 times on each of these datasets. In Fig. 8.4 we give the online error of the self-tuned Forgetron and the range of online errors attained by the CG algorithm. The self-tuned Forgetron outperforms the CG algorithm approximately half of the time, and the performance of the CG algorithm varies significantly from run to run. Therefore, when running the CG algorithm a single time on a given sequence of examples, we can only hope that we are lucky and that its performance is close to average or better. On the other hand, the deterministic self-tuned Forgetron does not suffer from this problem and consistently attains the average accuracy of the CG algorithm.

We conclude this section with a brief discussion of the time complexity of the various algorithms. Let κ denote the time required for a single evaluation of the kernel function. The implementation of the self-tuned Forgetron requires at most B kernel operations on each online round and an additional $O(B)$ operations. Therefore, its total complexity is $O(B\kappa)$ on each round. The complexity of a single round of the CG method is also $O(B\kappa)$. A direct implementation of the greedy removal Forgetron and of the CKS method requires calculating the prediction of the current hypothesis on each example in the active set. The resulting complexity is therefore $O(B^2\kappa)$. A more sophisticated implementation can decrease the number of kernel operations on each online round to be at most B. This can be done by maintaining a matrix with all the kernel evaluations for pairs $\mathbf{x}_i, \mathbf{x}_j$ where $i, j \in I_t$ and updating only a single row and a single column of this matrix on each online round. The resulting complexity of this implementation is $O(B\kappa + B^2)$. However, this implementation requires an additional storage for the $B \times B$ matrix described above.

9. Discussion. We presented a family of kernel-based online classifiers that restrict themselves to a memory of fixed size. The main idea behind our construction is to control the influence that each individual active example has on the online hypothesis. We achieve this control mechanism by repeatedly shrinking the weights that define the online hypothesis. Our shrinking step is done in a way that ensures that an active example can always be removed from the active set without significantly sacrificing classification accuracy.

Our empirical evaluation demonstrates that the gentle shrinking policy employed by the self-tuned Forgetron update significantly outperforms the aggressive shrinking policy of the basic Forgetron algorithm. Moreover, the original Perceptron algorithm, which neither performs any shrinking nor removes active examples, consistently outperforms the Forgetron variants. These observations reinforce our view of the shrinking and removal steps as a type of noise, which interferes with the online learning process. By making this noise as small as possible, we obtain online learning algorithms that approach the performance of the original Perceptron.

A nice property of this work, also shared by [1], is the way in which theory and practice go hand-in-hand. As mentioned in the introduction of this paper, previous attempts to address the task of online learning on a budget have all lacked a rigorous mathematical justification. In contrast, our algorithm and the algorithm in [1] both entertain formal worst-case guarantees. Our experiments demonstrate that the theoretically-motivated algorithms consistently outperform the heuristic approach.

Our experiments suggest that an online kernel method on a memory budget fails when its active set accumulates many noisy active examples. The basic Forgetron and the self-tuned Forgetron avoid this problem by always removing the oldest active example from the active set. This strategy ensures that a noisy active example is removed from the active set after precisely B updates. Even if an adversary creates the sequence of examples, our algorithms cannot be maneuvered into accumulating the noisy examples for a longer number of updates. The CG algorithm [1] exhibits a similar characteristic. Its randomized removal policy always gives an equal probability to removing each active example, and therefore cannot be manipulated into accumulating noisy examples. On the other hand, the more sophisticated removal strategies of the CKS algorithm [3] and the greedy Forgetron update can be exploited by an adversary. These algorithms can be tricked into maintaining noisy examples in their memory and discarding informative ones. Our experiments demonstrate that this phenomenon is exhibited even in the case of random label noise, where the input is not controlled by an adversary. This observation sheds a somewhat pessimistic light on the prospects of developing more sophisticated online kernel methods on a budget. It seems that any algorithm that applies a non-trivial removal strategy makes itself vulnerable to manipulation, and may be coerced into accumulating noise.

Several interesting open problems remain to be solved. A first challenge is to bridge the gap between the theoretical upper bound of $\sqrt{B+1}$ on the norm of the competitor and

$$U \;=\; \frac{1}{4}\sqrt{\frac{B+1}{\log(B+1)}} \;\;,$$

achieved by our algorithms. The CG algorithm of [1] managed to close this gap using a randomized algorithm and proving a bound on the *expected* number of mistakes (where expectation is taken over the internal randomization of their algorithm). The

question whether there exists a deterministic algorithm which matches the upper bound of $\sqrt{B+1}$ is open.

The intersection of machine learning and computational resource management is a fascinating research field, from both theoretical and practical standpoints. In this paper, we investigated a very simple online learning scenario, but our construction can be leveraged to solve more complex and realistic problems. For example, one could try to use our framework to devise online algorithms on a memory budget for tasks such as online regression, ranking, and sequence prediction. Another interesting problem is how to train thousands or even millions of online classifiers in parallel, where all of the classifiers share a common global memory of limited size. Rather than just limiting the number of active examples available to each classifier, we would like to dynamically allocate the global memory resource to the various classifiers in a way that would make optimal use of it.

REFERENCES

[1] N. Cesa-Bianchi and C. Gentile, *Tracking the best hyperplane with a simple budget perceptron*, in Proceedings of the Nineteenth Annual Conference on Computational Learning Theory, 2006, pp. 483–498.

[2] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, *Online passive aggressive algorithms*, Journal of Machine Learning Research, 7 (2006), pp. 551–585.

[3] K. Crammer, J. Kandola, and Y. Singer, *Online classification on a budget*, in Advances in Neural Information Processing Systems 16, 2003.

[4] O. Dekel, S. Shalev-Shwartz, and Y. Singer, *The Forgetron: A kernel-based perceptron on a fixed budget*, in Advances in Neural Information Processing Systems 18, 2005.

[5] C. Gentile, *A new approximate maximal margin classification algorithm*, Journal of Machine Learning Research, 2 (2001), pp. 213–242.

[6] D. P. Helmbold, J. Kivinen, and M. Warmuth, *Relative loss bounds for single neurons*, IEEE Transactions on Neural Networks, 10 (1999), pp. 1291–1304.

[7] J. Kivinen, A. J. Smola, and R. C. Williamson, *Online learning with kernels*, IEEE Transactions on Signal Processing, 52 (2002), pp. 2165–2176.

[8] J. Kivinen and M. Warmuth, *Exponentiated gradient versus gradient descent for linear predictors*, Information and Computation, 132 (1997), pp. 1–64.

[9] Y. Li and P. M. Long, *The relaxed online maximum margin algorithm*, in Advances in Neural Information Processing Systems 13, 1999.

[10] F. Rosenblatt, *The perceptron: A probabilistic model for information storage and organization in the brain*, Psychological Review, 65 (1958), pp. 386–407.

[11] V. N. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[12] J. Weston, A. Bordes, and L. Bottou, *Online (and offline) on an even tighter budget*, in Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, 2005, pp. 413–420.