Communication-Efficient Distributed Online Learning with Kernels

Michael Kamp[1] ✉, Sebastian Bothe[1], Mario Boley[2], and Michael Mock[1]

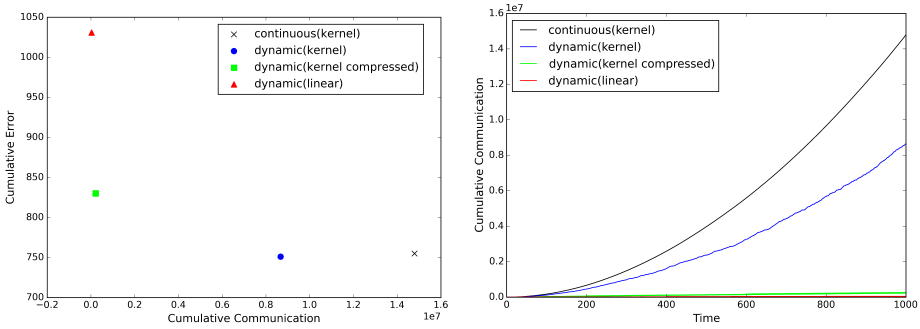
[1] Fraunhofer IAIS, Sankt Augustin, Germany
<name>.<surname>@iais.fraunhofer.de
[2] Saarland University, mboley@mmci.uni-saarland.de

Abstract. We propose an efficient distributed online learning protocol for low-latency real-time services. It extends a previously presented protocol to kernelized online learners that represent their models by a support vector expansion. While such learners often achieve higher predictive performance than their linear counterparts, communicating the support vector expansions becomes inefficient for large numbers of support vectors. The proposed extension allows for a larger class of online learning algorithms—including those alleviating the problem above through model compression. In addition, we characterize the quality of the proposed protocol by introducing a novel criterion that requires the communication to be bounded by the loss suffered.

1 Introduction

We consider the problem of distributed online learning for low-latency real-time services [4, 10]. In this scenario, a learning system of $m \in \mathbb{N}$ connected local learners provides a real-time prediction service on multiple dynamic data streams. In particular, we are interested in generic distributed online learning protocols that treat concrete learning algorithms as a black-box. The goal of such a protocol is to provide, in a communication efficient way, a service quality similar to a serial setting in which all examples are processed at a central location. While such an optimal predictive performance can be trivially achieved by centralizing all data, the required continuous communication usually exceeds practical limits (e.g., bandwidth constraints [1], latency [21, 8], or battery power [16, 5]). Similarly, communication limits can be satisfied trivially by letting all local learners work in isolation. However, this usually comes with a loss of service quality that increases with the number of local learners.

In previous work, we presented a protocol that effectively reduces communication while providing strict loss bounds for a class of algorithms that perform loss-proportional convex updates of linear models [10]. That is, algorithms that update linear models in the direction of a convex set with a magnitude proportional to the instantaneous loss (e.g., Stochastic Gradient Descent [2], or Passive Aggressive [3]). The protocol is able to cease communication as soon as no loss is suffered anymore. However, for most realistic problems this cannot be achieved



(a) (b)

Fig. 1. (a) Trade-off between cumulative error and cumulative communication, and (b) cumulative communication over time of a distributed learning system using the proposed protocol. The learning task is classifying instances from the UCI SUSY dataset with 4 learners, each processing 1000 instances. Parameters of the learners are optimized on a separate set of 200 instances per learner.

by linear models. Thus, a more complex hypothesis class is desirable that enables the learners to achieve zero loss and thus reach quiescence.

Kernelized online learning algorithms can provide such an extended hypothesis class, but practical versions of these algorithms do not perform loss-proportional convex updates (e.g., [12, 15, 20]). Therefore, in this paper we extend the class of algorithms to *approximately* loss-proportional convex updates (Sec. 2). This relaxation is particularly crucial for kernelized online learners for streams that represent the model by its support vector expansion. These learners use this relaxation in order to reduce the number of support vectors, since otherwise a monotonically increasing model size would render them infeasible in streaming settings.

Also, for the first time we characterize the quality of the proposed protocol by introducing a novel criterion for efficient protocols that requires a strict loss bound and ties the loss to the allowed amount of communication. In particular, the criterion implies that the communication vanishes whenever the loss approaches zero. We bound the loss and communication of the proposed protocol and show for which class of learning algorithms it fulfills the efficiency criterion (Sec. 3). While the strict loss bound required in our criterion can be achieved by periodically communicating protocols [14, 4], their communication never vanishes, independent of their loss, which is also required for efficiency. By communicating only when it significantly improves the service quality, our protocol achieves similar service quality as any periodically communicating protocol while communicating less by a factor depending on its in-place loss.

To further amplify this advantage, we apply methods from serial kernelized in-stream learning approaches. These approaches reduce the number of support

vectors, e.g., by truncating individual support vectors with small weights [12], or by projecting a single support vector on the span of the remaining ones [15, 20].

We illustrate the impact of the choice of the hypothesis class on the predictive performance and communication as well as the impact of model compression on an example dataset in Fig. 1. In this example, we predicted the class of instances drawn from the SUSY dataset from the UCI machine learning repository [13]. The learning systems using linear models continuously suffer loss resulting in a large cumulative error, but since the linear models are small compared to support vector expansions, the cumulative communication is small. A continuously synchronizing protocol using support vector expansions has a significantly smaller loss at the cost of very high communication, since each synchronization requires to send models with a growing number of support vectors. Using the proposed dynamic protocol, this amount of communication can be reduced without losing in prediction quality. In addition, when using model compression the communication can be further reduced to an amount similar to the linear model, but at the cost of prediction quality.

We further discuss the behavior of our protocol with respect to the trade-off between predictive performance and communication, and point out the strengths and weaknesses of the protocol in Sec. 4.

2 Distributed Online Learning with Kernels

In this section, we provide preliminaries and describe the protocol, extend it from linear function spaces to kernel Hilbert spaces, and provide an effectiveness criterion for distributed online learning. For that, we consider **distributed online learning protocols** $\Pi = (\mathcal{A}, \sigma)$ that run an online learning algorithm $\mathcal{A}$ on a distributed system of $m \in \mathbb{N}$ local learners and exchange information between these learners using a synchronization operator σ.

Preliminaries: The **online learning algorithm** $\mathcal{A} = (\mathcal{H}, \varphi, \ell)$ run at each **local learner** $i \in [m]$ maintains a **local model** $f^i \in \mathcal{H}$ from a function space $\mathcal{H}$ using an update rule φ and a loss function ℓ. That is, at each time point $t \in \mathbb{N}$, each learner i observes an individual input $\left(x_t^i, y_t^i\right)$ drawn independently from a time-variant distribution $P_t \colon X \times Y \to [0, 1]$ over an input space $X \times Y$. Based on this input and the local model, the local learner provides a service whose quality is measured by the **loss function** $\ell \colon \mathcal{H} \times X \times Y \to \mathbb{R}_+$. After providing the service, the local learner updates its local model using the **update rule** $\varphi \colon \mathcal{H} \times X \times Y \to \mathcal{H}$ in order to minimize the cumulative loss. The **synchronization operator** $\sigma \colon \mathcal{H}^m \to \mathcal{H}^m$ transfers the current **model configuration** $\mathbf{f} = \left(f^1, \ldots, f^m\right)$ of m local models to the synchronized configuration $\sigma(\mathbf{f})$. In the following, we recapitulate the dynamic protocol presented in [10] as well as two baseline protocols, i.e., a continuously and a periodic protocol.

Given an online learning algorithm $\mathcal{A}$, the **periodic protocol** $\mathcal{P} = (\mathcal{A}, \sigma_b)$ synchronizes every $b \in \mathbb{N}$ time steps the current model configuration $\mathbf{f}$ by replacing all local models by their joint **average** $\overline{\mathbf{f}} = {}^1\!/_m \sum_{i=1}^m f^i$. That is, the

synchronization operator is given by

$$\sigma_b(\mathbf{f}_t) = \begin{cases} \left(\overline{\mathbf{f}}_t, \ldots, \overline{\mathbf{f}}_t\right), & \text{if } b \mid t \\ \mathbf{f}_t = (f_t^1, \ldots, f_t^m), & \text{otherwise} \end{cases} .$$

A special case of this is the **continuous protocol** $\mathcal{C} = (\mathcal{A}, \sigma_1)$ that continuously synchronizes every round, i.e., $\sigma_1(\mathbf{f}) = \left(\overline{\mathbf{f}}, \ldots, \overline{\mathbf{f}}\right)$.

The **dynamic protocol** $\mathcal{D} = (\mathcal{A}, \sigma_\Delta)$ synchronizes the local learners using a **dynamic operator** σ_Δ [10]. This operator only communicates when the **model divergence**

$$\delta(\mathbf{f}) = \frac{1}{m} \sum_{i=1}^{m} \left\| f^i - \overline{\mathbf{f}} \right\|^2 \tag{1}$$

exceeds a **divergence threshold** Δ. That is, the dynamic averaging operator is defined as

$$\sigma_\Delta(\mathbf{f}_t) = \begin{cases} (\overline{\mathbf{f}}_t, \ldots, \overline{\mathbf{f}}_t), & \text{if } \delta(\mathbf{f}_t) > \Delta \\ \mathbf{f}_t, & \text{otherwise} \end{cases} .$$

In order to decide when to communicate, each local learner $i \in [m]$ monitors the **local condition** $\|f_t^i - r_t\|^2 \leq \Delta$ for a **reference model** $r_t \in \mathcal{H}$ that is common among all learners (see [11, 19, 6, 7] for a more general description of this method). The local conditions guarantee that if none of them is violated, the divergence does not exceed the threshold Δ. The closer the reference model is to the true average of local models, the tighter are the local conditions. Generally, the first choice for the reference model is the average model from the last synchronization step. Note, however, that there are several refinements of this choice that can be used in practice to further reduce communication.

Efficiency Criterion: In the following, we introduce performance measures in order to analyze the dynamic protocol and compare it to the continuous and periodic protocols. We measure the predictive performance of a distributed online learning system until time $T \in \mathbb{N}$ by its cumulative loss

$$L(T, m) = \sum_{t=1}^{T} \sum_{i=1}^{m} \ell(f_t^i, (x_t^i, y_t^i)) .$$

Performance guarantees are typically given by a **loss bound** $\mathbf{L}(T, m)$, i.e., for all possible input sequences it holds that $L(T, m) \leq \mathbf{L}(T, m)$. These bounds can be defined with respect to a sequence of reference models, in which case they are referred to as (shifting) **regret bounds**.

We measure its performance in terms of communication by its cumulative communication

$$C(T, m) = \sum_{t=1}^{T} c(\mathbf{f}_t) ,$$

where $c : \mathcal{H}^m \to \mathbb{N}$ measures the number of bytes required by the learning protocol to synchronize models $\mathbf{f}_t = \left(f_t^1, \ldots, f_t^m\right)$ at time t.

There is a natural trade-off between communication and loss of a distributed online learning system. On the one hand, a loss similar to a serial setting can be trivially achieved by continuous synchronization. On the other hand, communication can be entirely omitted. The trade-off for these two extreme protocols can be easily determined: if the cumulative loss of an online learning algorithm $\mathcal{A}$ is bounded by $\mathbf{L}_{\mathcal{A}}(T)$, the loss of a permanently centralizing system with m local learners running $\mathcal{A}$ is bounded by $\mathbf{L}_{\mathcal{C}}(T,m) = \mathbf{L}_{\mathcal{A}}(mT)$, i.e., the loss bound of a serial online learning algorithm processing mT inputs. The protocol transmits $\mathcal{O}(m)$ messages of size up to $\mathcal{O}(T)$ in every of the T points in time. At the same time, the loss of a distributed system without any synchronization is bounded by $\mathbf{L}(T,m) = m\mathbf{L}_{\mathcal{A}}(T)$, whereas the communication is $C(T) = 0$.

The communication bound of an adaptive protocol should only depend on $\mathbf{L}_{\mathcal{A}}(T)$ and not on T, while at the same time retaining the loss bound of the serial setting. In the following definition we formalize this in order to provide a strong criterion for effectiveness of distributed online learning protocols.

Definition 1 *A distributed online learning protocol $\Pi = (\mathcal{A}, \sigma)$ processing mT inputs is **consistent** if it retains the loss bound of the serial online learning algorithm $\mathcal{A}$, i.e.,*

$$\boldsymbol{L}_{\Pi}(T,m) \in \mathcal{O}\left(\boldsymbol{L}_{\mathcal{A}}(mT)\right) \quad .$$

*The protocol is **adaptive** if its communication bound is linear in the number of local learners m and the loss bound $\boldsymbol{L}_{\mathcal{A}}(mT)$ of the serial online learning algorithm, i.e.,*

$$C_{\Pi}(T,m) \in \mathcal{O}\left(m\boldsymbol{L}_{\mathcal{A}}(mT)\right) \quad .$$

An **efficient** protocol is adaptive and consistent at the same time. In the following section we theoretically analyze the performance of the dynamic protocol with respect to this efficiency criterion.

Extension to Kernel Methods: The protocols presented above are defined for models from a Euclidean vector space. In this paper, we generalize $\mathcal{H}$ to be a **reproducing kernel Hilbert space** $\mathcal{H} = \{f : X \to \mathbb{R} | f(\cdot) = \sum_{j=1}^{\dim F} w_j \Phi_j(\cdot)\}$ with **kernel function** $k : X \times X \to \mathbb{R}$, **feature space** F, and a mapping $\Phi : X \to F$ into the feature space [18]. The kernel function corresponds to an inner product of input points mapped into feature space, i.e., $k(x,x') = \sum_{j=1}^{\dim F} \xi_j \Phi_j(x)\Phi_j(x')$ for constants $\xi_1, \xi_2, \dots \in \mathbb{R}$. Thus, we can express the model in its **support vector expansion**, or dual representation, i.e., $f(\cdot) = \sum_{x \in S} \alpha_x k(x, \cdot)$ with a set of **support vectors** $S = \{x_1, \dots, x_{|S|}\} \subset X$ and corresponding **coefficients** $\alpha_x \in \mathbb{R}$ for all $x \in S$. This implies that the linear weights $w = (w_1, w_2, \dots) \in F$ defining f are given implicitly by $w_i = \sum_{x \in S} \xi_i \alpha_x \Phi_i(x)$. In order to apply the previously defined synchronization protocols to models from a reproducing kernel Hilbert space, we determine how to calculate the average of a model configuration and its divergence. For that, let $\mathbf{f} = \left(f^1, \dots, f^m\right) \subset \mathcal{H}$ be a model configuration with corresponding weight vectors $\left(w^1, \dots, w^m\right) \subset F$, where each

model $i \in [m]$ has support vectors $S^i = \{x_1^i, \ldots, x_{|S^i|}^i\} \subset X$ and coefficients α_x^i for all $x \in S^i$. The average is given by

$$\overline{\mathbf{f}}(\cdot) = \frac{1}{m}\sum_{i=1}^{m} f^i(\cdot) = \frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{\dim F} w_j^i \Phi_j(\cdot) = \frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{\dim F}\sum_{x\in S^i} \xi_j \alpha_x^i \Phi_j(x)\Phi_j(\cdot) \ .$$

We can simplify the above equation to $\overline{\mathbf{f}}(\cdot) = \frac{1}{m}\sum_{i=1}^{m}\sum_{x\in S^i}\alpha_x^i k(x,\cdot)$. By defining the union of support vectors $\overline{S} = \bigcup_{i\in[m]} S^i = \{s_1, \ldots, s_{|\overline{S}|}\}$ and augmented coefficients $\overline{\alpha}_s^i \in \mathbb{R}$, which are given by

$$\overline{\alpha}_s^i = \begin{cases} \alpha_x^i, & \text{if } x = s \\ 0, & \text{otherwise} \end{cases} \ ,$$

the dual representation of the average directly follows.

Proposition 2 *For a model configuration $\mathbf{f} = \left(f^1, \ldots, f^m\right) \subset \mathcal{H}$, where each model $i \in [m]$ has augmented coefficients $\overline{\alpha}_s^i$ for $s \in \overline{S}$, the average $\overline{\mathbf{f}} \in \mathcal{H}$ is given by*

$$\overline{\mathbf{f}}(\cdot) = \sum_{s\in\overline{S}} \left(\frac{1}{m}\sum_{i=1}^{m}\overline{\alpha}_s^i\right) k(s,\cdot) \ ,$$

with support vectors $\overline{S}$ and coefficients $\overline{\alpha}_s = {}^1\!/_m \sum_{i=1}^{m}\overline{\alpha}_s^i$ for all $s \in \overline{S}$.

Using this definition of the average, we now define the distance between models in $\mathcal{H}$ and the divergence δ of a model configuration $\mathbf{f} \subset \mathcal{H}$. For an individual model f^i and the average $\overline{\mathbf{f}}$, the distance induced by the inner product of $\mathcal{H}$ is defined by $\left\|f^i - \overline{\mathbf{f}}\right\| = \langle f^i, f^i\rangle + \langle \overline{\mathbf{f}}, \overline{\mathbf{f}}\rangle - 2\langle f^i, \overline{\mathbf{f}}\rangle$, i.e.,

$$\left\|f^i - \overline{\mathbf{f}}\right\| = \sum_{x\in S^i}\left(\alpha_x^i\right)^2 k(x,x) + \sum_{s\in\overline{S}}\left(\overline{\alpha}_s\right)^2 k(s,s) - 2\sum_{x\in S^i}\sum_{s\in\overline{S}}\alpha_x^i\overline{\alpha}_s k(x,s) \ .$$

Using this distance, we can compute the divergence (Eq. 1) for models from a reproducing kernel Hilbert space.

3 Performance Guarantees

In order to determine the performance of the dynamic protocol, we start by extending the definition of loss-proportional convex update rules. This allows us to bound the loss for kernelized online learning algorithms that reduce their model size using a compression step.

Let $\varphi\colon \mathcal{H}\times X\times Y \to \mathcal{H}$ be a loss-proportional convex update rule, then $\widetilde{\varphi}$ is an **approximately loss-proportional convex update rule** if for all $f \in \mathcal{H}$, $x \in X$, and $y \in Y$ it holds that $\|\widetilde{\varphi}(f,x,y) - \varphi(f,x,y)\| \leq \epsilon$. With this, we can bound the distance between two models after the approximate update step.

Lemma 3 *For two models $f, g \in \mathcal{H}$ and an approximately loss-proportional convex update rule $\widetilde{\varphi}$, with $\|\widetilde{\varphi}(f,x,y) - \varphi(f,x,y)\| \leq \epsilon$ for the corresponding loss-proportional convex update rule φ, it holds that*

$$\|\widetilde{\varphi}(f,x,y) - \widetilde{\varphi}(g,x,y)\|^2 \leq \|f-g\|^2 - \gamma^2 \left(\ell(f,x,y) - \ell(g,x,y)\right)^2 + 2\epsilon^2 \ .$$

Proof. We abbreviate $\varphi(f,x,y)$ as $\varphi(f)$. Then $\|\widetilde{\varphi}(f) - \varphi(f)\| \leq \epsilon$ implies for $f, g \in \mathcal{H}$ that $\|\widetilde{\varphi}(f) - \widetilde{\varphi}(g)\|^2 \leq \|\varphi(f) - \varphi(g)\|^2 + 2\epsilon^2$. Together with the result from Lm. 4 in [10], i.e., $\|\varphi(f) - \varphi(g)\|^2 \leq \|f-g\|^2 - \gamma^2 \left(\ell(f) - \ell(g)\right)^2$, follows the result. □

Using Lm. 3, we can bound the loss of our protocol.

Theorem 4 *Let $\mathcal{A}$ be an online learning algorithm with γ-loss-proportional convex update rule φ. Let $\mathbf{d}_1, \dots \mathbf{d}_T$ and $\mathbf{p}_1, \dots, \mathbf{p}_T$ be two sequences of model configurations such that $\mathbf{d}_1 = \mathbf{p}_1$ and the first sequence is maintained by the dynamic protocol $\mathcal{D} = (\mathcal{A}, \sigma_\Delta)$ and the second by the periodic protocol $\mathcal{P} = (\mathcal{A}, \sigma_b)$. That is, for $t = 1, \dots, T$ the sequence is defined by $\mathbf{d}_{t+1} = \sigma_\Delta\left(\varphi(\mathbf{d}_t)\right)$, and $\mathbf{p}_{t+1} = \sigma_b\left(\varphi(\mathbf{p}_t)\right)$ respectively. Then it holds that*

$$L_{\mathcal{D}}(T,m) \leq L_{\mathcal{P}}(T,m) + \frac{T}{\gamma^2}(\Delta + 2\epsilon^2) \ .$$

Proof. First note that for simplicity we abbreviate $\ell(f_t, x_t, y_t)$ by $\ell(f_t)$. We combine our Lm. 3 with Lm. 3 from [10] which states that

$$\frac{1}{m}\sum_{i=1}^{m} \|\sigma_\Delta(\mathbf{d})^i - \sigma_b(\mathbf{p})^i\|^2 \leq \frac{1}{m}\sum_{i=1}^{m} \|d^i - p^i\|^2 + \Delta \ .$$

This yields for all $t \in [T]$ that

$$\sum_{i=1}^{m} \left\|d_{t+1}^i - p_{t+1}^i\right\|^2 \leq \sum_{i=1}^{m} \left\|d_t^i - p_t^i\right\|^2 - \gamma^2 \sum_{i=1}^{m} \left(\ell(d_t^i) - \ell(p_t^i)\right)^2 + \Delta + 2\epsilon^2 \ .$$

By applying this inequality recursively for $t = 1, \dots, T$ it follows that

$$\sum_{i=1}^{m} \left\|d_{t+1}^i - p_{t+1}^i\right\|^2 \leq \sum_{i=1}^{m} \left\|d_1^i - p_1^i\right\|^2 + T(\Delta + 2\epsilon^2) - \gamma^2 \sum_{t=1}^{T}\sum_{i=1}^{m} \left(\ell(d_t^i) - \ell(p_t^i)\right)^2 .$$

Using $\mathbf{d}_1 = \mathbf{p}_1$, we conclude that

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{i=1}^{m} \left(\ell(d_t^i) - \ell(p_t^i)\right)^2 \leq \frac{1}{\gamma^2}\left(T(\Delta + 2\epsilon^2) - \sum_{i=1}^{m} \left\|d_{t+1}^i - p_{t+1}^i\right\|^2\right) \leq \frac{1}{\gamma^2} T\Delta \\
&\Leftrightarrow L_{\mathcal{D}}(T)^m - L_{\mathcal{P}}(T)^m \leq \frac{1}{\gamma^2} T(\Delta + 2\epsilon^2)
\end{aligned}$$

□

By setting the communication period $b = 1$, this result also holds for the continuous protocol $\mathcal{C}$.

The result of Thm. 4 is similar to the original loss bound of the dynamic protocol but also accounts for the inaccuracy of the update rule, e.g., because of model compression. We can apply the original consistency result: if the continuous protocol is consistent, then the dynamic protocol is consistent as well. For Stochastic Gradient Descent it has been shown that the dynamic protocol is consistent for linear models [10]. From Thm. 4 follows that the dynamic protocol remains consistent for approximately loss-proportional update rules. Note that for static target distributions, consistency can be achieved by a decreasing divergence threshold and compression error, i.e., $\Delta_t = t^{-1/2}$ and $\epsilon = t^{-1/4}$.

We now provide communication bounds for the dynamic protocol. For that, assume that the m learners maintain models in their support vector expansion. Let $S_t^i \subset \mathbb{R}^d$ denote the set of support vectors of learner $i \in [m]$ at time t and α_t^i the corresponding coefficients. Let $B_x \in \mathcal{O}(d)$ be the number of bytes required to transmit one support vector and $B_\alpha \in \mathcal{O}(1)$ be the number of bytes required for the corresponding weight. Furthermore, let $I : \mathbb{N} \times [m] \to \{0, 1\}$ be an indicator function that is 1 if for learner i at time t a new support vector has been added during the update.

We assume that a designated coordinator node performs the synchronizations, i.e., all local learners transmit their models to the coordinator which in turn sends the synchronized model back to each learner. Furthermore, we assume that all protocols apply the following trivial communication reduction strategy. Let t' be the time of last synchronization. Assume the coordinator stored the support vectors of the last average model $\overline{S}_{t'}$. Whenever a learner i has to send its model to the coordinator, it sends all support vector coefficients α but only the new support vectors, i.e., only $S_t^i \setminus S_{t'}^i$. This avoids redundant communication at the cost of higher memory usage at the coordinator side. In turn, after averaging the models, the coordinator sends to learner i all support vector coefficients, but only the support vectors $\overline{S}_t \setminus S_t^i$.

We start by bounding the communication of a continuous protocol $\mathcal{C}$, i.e., one that transmits all models from each learner in each round. The trivial communication reduction technique discussed above implies that in each round, a learner transmits its full set of support vector coefficients and potentially one support vector—depending on whether a new support vector was added in this round. Thus, at time t learner i submits

$$|S_t^i| B_\alpha + I(t, i) B_x \tag{2}$$

bytes to the coordinator. The coordinator transmits to learner $i \in m$ all support vector coefficients of the average model and all its support vectors, except the support vectors S_t^i of the local model at learner i. Thus, it transmits the following amount of bytes.

$$\left|\overline{S}_t\right| B_\alpha + \left|\overline{S}_t \setminus S_t^i\right| B_x = \left|\bigcup_{j=1}^{m} S_t^j\right| B_\alpha + \left|\bigcup_{j=1}^{m} S_t^j \setminus S_t^i\right| B_x \ . \tag{3}$$

With this we can derive the following communication bound.

Proposition 5 *The communication of the continuous protocol $\mathcal{C}$ on $m \in \mathbb{N}$ learners until time $T \in \mathbb{N}$ is bound by*

$$C_{\mathcal{C}}(T,m) \leq Tm2|\overline{S}_T|B_\alpha + m|\overline{S}_T|B_x \leq m^2T^2B_\alpha + m^2TB_x \in \mathcal{O}\left(m^2T^2\right) \ .$$

Proof. The constantly synchronizing protocol transmits at each time step from each learner a set of support vector coefficients and potentially one support vector to the coordinator. The amount of bytes is given in Eq. 2. The coordinator transmits the averaged model back to each learner with an amount of bytes as given in Eq. 3. Summing up the communication over $T \in \mathbb{N}$ time points and m learners yields

$$\begin{aligned} C_{\mathcal{C}}(T,m) &= \sum_{t=1}^{T}\sum_{i=1}^{m}\left(|S_t^i|B_\alpha + I(t,i)B_x + \left|\bigcup_{j=1}^{m} S_t^j\right| B_\alpha + \left|\bigcup_{j=1}^{m} S_t^j \setminus S_t^i\right| B_x\right) \\ &= \sum_{t=1}^{T}\sum_{i=1}^{m}\left(|S_t^i|B_\alpha + \left|\overline{S}_t\right| B_\alpha + I(t,i)B_x + \left|\overline{S}_t \setminus S_t^i\right| B_x\right) \ . \end{aligned}$$

We analyze this sum separately in terms of bytes required for sending the support vectors and bytes for sending the coefficients. The amount of bytes for sending the support vectors is bounded by $m|S_T^i|B_x$, as we show in the following.

$$\begin{aligned} &\sum_{t=1}^{T}\sum_{i=1}^{m} I(t,i)B_x + \left|\overline{S}_t \setminus S_t^i\right| B_x = \underbrace{\sum_{t=1}^{T}\sum_{i=1}^{m} I(t,i)B_x}_{=|\overline{S}_T|B_x} + \sum_{t=1}^{T}\sum_{i=1}^{m}\left|\overline{S}_t \setminus S_t^i\right| B_x \\ =&|\overline{S}_T|B_x + \sum_{t=1}^{T}\sum_{i=1}^{m}\left|\left(\bigcup_{j=1}^{m} S_t^j \setminus \bigcup_{j=1}^{m} S_{t-1}^j\right) \setminus \left(S_t^i \setminus \overline{S}_{t-1}\right)\right| B_x \\ \leq&|\overline{S}_T|B_x + \sum_{t=1}^{T}\sum_{i=1}^{m}\sum_{\substack{j=1\\ j\neq i}}^{m} I(t,i)B_x \leq |\overline{S}_T|B_x + \sum_{t=1}^{T}\sum_{i=1}^{m}(m-1)I(t,i)B_x \\ \leq&|\overline{S}_T|B_x + (m-1)|\overline{S}_T|B_x = m|\overline{S}_T|B_x \ . \end{aligned}$$

We now bound the amount of bytes required for sending the support vector coefficients.

$$\sum_{t=1}^{T}\sum_{i=1}^{m} \underbrace{|S_t^i|}_{\leq|\overline{S}_T|} B_\alpha + \underbrace{|\overline{S}_t|}_{\leq|\overline{S}_T|} B_\alpha \leq \sum_{t=1}^{T}\sum_{i=1}^{m} 2|\overline{S}_T|B_\alpha = Tm2|\overline{S}_T|B_\alpha \ .$$

From $\left|\overline{S}_T\right| \leq mT$ and the fact that we regard $B_\alpha \in \mathcal{O}(1)$ and $B_x \in \mathcal{O}(d)$ as constants we can follow that

$$C_{\mathcal{C}}(T,m) \leq 2Tm|\overline{S}_T|B_\alpha + m|\overline{S}_T|B_x \leq m^2T^2B_\alpha + m^2TB_x \in \mathcal{O}\left(m^2T^2\right) \ .$$

□

Note that this communication bound implies that—unlike for linear models—synchronizing models in their support vector expansion requires even more communication than centralizing the input data. However, in real-time prediction applications, the latency induced by central computation can exceed the time constraints, rendering continuous synchronization a viable approach nonetheless.

Similarly, the communication of a periodic protocol $\mathcal{P}$ that communicates every $b \in \mathbb{N}$ steps (b is often referred to as mini-batch size) can be bounded by

$$C_{\mathcal{P}}(T,m) \leq \frac{T}{b} 2m|\overline{S}_T|B_\alpha + m|\overline{S}_T|B_x \leq \frac{T}{b} m^2 T B_\alpha + m^2 T B_x \in \mathcal{O}\left(\frac{1}{b} m^2 T^2\right) \ .$$

We now for the first time provide a communication bound for the dynamic protocol $\mathcal{D}$. For that, we first bound the number of synchronization steps and then analyze the amount of communication per synchronization.

Proposition 6 *Let $\mathcal{A} = (\mathcal{H}, \widetilde{\varphi}, \ell)$ be an online learning algorithm with an approximately loss-proportional convex update rule $\widetilde{\varphi}$ for which holds that $\|f - \widetilde{\varphi}(f,x,y)\| \leq \eta \ell(f,x,y)$. The number of synchronizations $V_{\mathcal{D}}(T)$ of the dynamic protocol $\mathcal{D}$ running $\mathcal{A}$ in parallel on m nodes until time $T \in \mathbb{N}$ with divergence threshold Δ is bounded by*

$$V_{\mathcal{D}}(T) \leq \frac{\eta}{\sqrt{\Delta}} L_{\mathcal{D}}(T,m) \ .$$

where $L_{\mathcal{D}}(T,m)$ denotes the cumulative loss of $\mathcal{D}$.

Proof. For this proof, we abbreviate $\ell(f_t^i, x_t^i, y_t^i)$ as $\ell(f_t^i)$ and $\widetilde{\varphi}(f_t^i, x_t^i, y_t^i)$ as $\widetilde{\varphi}(f_t^i)$. The dynamic protocol synchronizes if a local condition $\|f_t^i - r_t\|^2 \leq \Delta$ is violated. Now assume that at $t = 1$ all models are initialized with $f_1^1 = \cdots = f_1^m$ and $r_1 = \overline{\mathbf{f}}_1$, i.e., for all local learners i it holds that $\|f_1^i - r_1\| = 0$. A violation, i.e., $\|f_t^i - r_t\| > \sqrt{\Delta}$, occurs if one local model drifts away from r_t by more than $\sqrt{\Delta}$. After a violation, a synchronization is performed and $r_t = \overline{\mathbf{f}}_t$, hence $\|f_t^i - r_t\| = 0$ and the situation is again similar to the initial setup for $t = 1$. In the worst case, a local learner drifts continuously in one direction until a violation occurs. Hence, we can bound the number of violations $V_i(T)$ at a single learner i by the sum of its drifts divided by $\sqrt{\Delta}$:

$$V_l(T) \leq \frac{1}{\sqrt{\Delta}} \sum_{t=1}^{T} \|f_t^i - f_{t+1}^i\| = \frac{1}{\sqrt{\Delta}} \sum_{t=1}^{T} \underbrace{\|f_t^i - \widetilde{\varphi}(f_t^i)\|}_{\leq \eta \ell(f_t^i)} \leq \frac{1}{\sqrt{\Delta}} \sum_{t=1}^{T} \eta \ell(f_t^i) \ .$$

With this, we can bound the amount of points in time $t \in [T]$ where at least one learner l has a violation, i.e., $V(T)$. In the worst case, all violations at all local learners occur at different time points, so that we can upper bound $V(T)$ by the sum of local violations $V_i(T)$ which is again upper bounded by the cumulative sum of drifts of all local models:

$$V(T) \leq \sum_{i=1}^{m} V_i(T) \leq \frac{1}{\sqrt{\Delta}} \sum_{t=1}^{T} \sum_{i=1}^{m} \eta \ell(f_t^i) = \frac{\eta}{\sqrt{\Delta}} L_{\mathcal{D}}(T,m) \ .$$

□

In the following theorem we bound the overall communication by combining this bound on the number of synchronizations with an analysis of the amount of bytes transfered per synchronization.

Theorem 7 *Let $\mathcal{A} = (\mathcal{H}, \widetilde{\varphi}, \ell)$ be an online learning algorithm with approximately loss-proportional update rule $\widetilde{\varphi}$ and $\|f - \widetilde{\varphi}(f, x, y)\| \leq \eta \ell(f, x, y)$. The amount of communication $C_{\mathcal{D}}(T, m)$ of the dynamic protocol $\mathcal{D}$ running $\mathcal{A}$ in parallel on m nodes until time $T \in \mathbb{N}$ with divergence threshold Δ is bounded by*

$$C_{\mathcal{D}}(T, m) \leq \frac{\eta}{\sqrt{\Delta}} L_{\mathcal{D}}(T, m) \left(2m \left|\overline{S}_T\right| B_\alpha\right) + m \left|\overline{S}_T\right| B_x$$

Proof. Assume that at time T, the dynamic protocol performs a synchronization. Then, similar to the argument for the continuous protocol, the support vector set at time T is similar for all learners and independent of the number of synchronization steps before. In particular, it is the same if a synchronization was performed in every time step. Thus, again the amount of bytes required for sending the support vectors is bounded by $m \left|\overline{S}_T\right| B_x$. Let $\theta \colon \mathbb{N} \to \{0, 1\}$ be an indicator function such that $\theta(t) = 1$ if at time t the dynamic protocol performed a synchronization and $\theta(t) = 0$ otherwise. Then, the amount of bytes required to send all the support vector coefficients until time T is

$$\sum_{t=1}^{T} \theta(t) \sum_{i=1}^{m} \left(\left|S_t^i\right| + \left|\overline{S}_t\right|\right) B_\alpha \leq \underbrace{\sum_{t=1}^{T} \theta(t)}_{=V_{\mathcal{D}}(T)} \sum_{l=1}^{m} 2|\overline{S}_T| B_\alpha \leq \underbrace{\frac{\eta}{\sqrt{\Delta}} L_{\mathcal{D}}(T, m)}_{\text{Prop. 6}} \left(2m|\overline{S}_T| B_\alpha\right)$$

Together with the amount of bytes required for exchanging all support vectors this yields $C_{\mathcal{D}}(T, m) \leq \frac{\eta}{\sqrt{\Delta}} L_{\mathcal{D}}(T, m) \left(2m|\overline{S}_T| B_\alpha\right) + m \left|\overline{S}_T\right| B_x$. □

Note that the loss bounds for online learning algorithms are typically sublinear in T, e.g., optimal regret bounds for static target distributions are in $\mathcal{O}(\sqrt{T})$. In these cases, the dynamic protocol has an amount of communication in $\mathcal{O}(m^2 T \sqrt{T})$ which is smaller than $\mathcal{O}(m^2 T^2)$ of the continuously and periodic protocols by a factor of $\sqrt{T}$.

In the original case of linear models instead, the dynamic protocol only transmits m weight vectors of fixed size per synchronization. In this case the amount of communication per synchronization is bounded by a constant. If for an online learning algorithm $\mathcal{A}$ and the periodic protocol $\mathcal{P}$ it holds that $\mathbf{L}_{\mathcal{P}}(T, m) \leq \mathbf{L}_{\mathcal{A}}(mT)$, then by Thm. 4 it also holds that $\mathbf{L}_{\mathcal{D}}(T, m) \leq \mathbf{L}_{\mathcal{A}}(mT)$. This implies that the dynamic protocol is adaptive. In the following corollary, we show that for linear models, the dynamic protocol is adaptive when using the Stochastic Gradient Descent algorithm.

Corollary 8 *The dynamic protocol $\mathcal{D} = (SGD, \sigma_\Delta)$ using Stochastic Gradient Descent SGD with linear models is adaptive, i.e.,*

$$C_{\mathcal{D}}(T, m) \in \mathcal{O}\left(m \boldsymbol{L}_{SGD}(mT)\right)$$

Proof. The amount of synchronizations of the dynamic protocol is bounded by $V(T)$ (see Prop. 6). In each synchronization, each learner transmits one linear model, i.e., one weight vector of fixed size to the coordinator. The coordinator submits one averaged weight vector back to each learner. Thus, the amount of communication per synchronization is bounded by $\mathbf{c}_m \in \mathbb{N}$, where $\mathbf{c}_m \in \mathcal{O}(m)$. Then, the total communication is bounded by

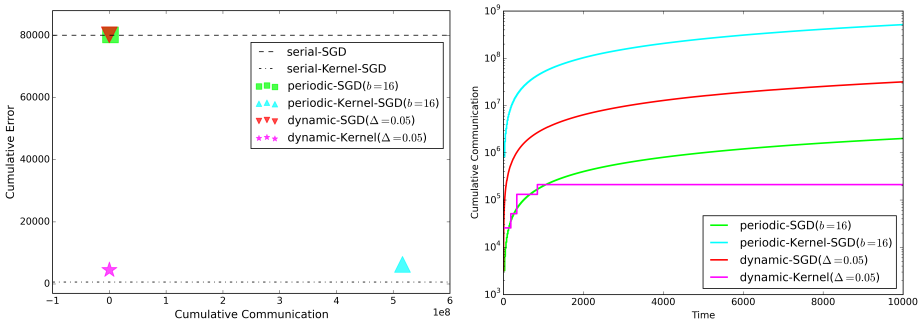
$$C_{\mathcal{D}}(T,m) \leq \mathbf{c}_m \frac{\eta}{\sqrt{\Delta}} L_{\mathcal{D}}(T,m) \in \mathcal{O}\left(mL_{\mathcal{D}}(T,m)\right) \quad .$$

The dynamic protocol retains the loss bound of Stochastic Gradient Descent [10], i.e., $L_{\mathcal{D}}(T,m) \leq \mathbf{L}_{\text{SGD}}(mT)$ □

Unfortunately, from Thm. 4 also follows that the dynamic protocol applied to kernelized online learning algorithms that do not bound the size of their models does not comply to the strict notion of adaptivity as given in Def. 1. That is, because the model size and thus the size of each message to and from the coordinator can grow with T. Nonetheless, the theorem guarantees that if the learners do not suffer loss anymore, the dynamic protocol reaches quiescence.

In order to make the dynamic protocol adaptive in the strict sense of Def. 1, the model size has to be bounded. For kernelized online learning in streams, several **model compression** techniques have been proposed [12, 15, 20]. These techniques typically guarantee that the compression error is bounded, i.e., for the **compressed model** $\widetilde{f}$ it holds that $\left\|f - \widetilde{f}\right\| \leq \epsilon$. From this directly follows that if the base algorithm uses a loss-proportional convex update rule φ, the compressed version is an approximately loss-proportional convex update rule $\widetilde{\varphi}$.

One approach to compressing the support vector expansion is to project a new support vector on the span of the remaining ones and thus avoid adding it to the support set. Another one is to truncate support vectors with small coefficients. For the projection approach (e.g., described in [15]) the error bound is independent of the learning algorithm. However, there is no bound on the number of support vectors. Thus, even though the model size is reduced in practice, there is no formal bound on the model size. For the truncation approach, however, [12] have shown that an error bound as well as a bound on the number of support vectors can be achieved when using Stochastic Gradient Descent. Specifically, for a fixed model size of τ support vectors, they have shown that the compression error is bound by $\left\|f - \widetilde{f}\right\| \leq \epsilon \in \mathcal{O}\left(\frac{1}{\lambda}(1-\lambda)^{\tau}\right)$, where $\lambda \in \mathbb{R}$ is the learning rate of the Stochastic Gradient Descent algorithm (SGD). Therefore, we can follow that the dynamic protocol with SGD using kernel models compressed by truncation is adaptive. Specifically for SGD, [4] have shown that periodic synchronizations retain the serial loss bound of SGD. It is consistent in this setting, because the dynamic protocol in turn retains the loss bounds of any periodic protocol. Since it is both consistent and adaptive, the dynamic protocol is efficient.



(a) (b)

Fig. 2. (a) Trade-off between cumulative error and cumulative communication and (b) cumulative communication over time of the dynamic protocol versus a periodic protocol. 32 learners perform a stock price prediction task using SGD (learning rate η and regularization parameter λ optimized over 200 instances, with $\eta = 10^{-10}$, $\lambda = 1.0$ for the periodic protocol, and $\eta = 1.0$, $\lambda = 0.01$ for the dynamic protocol) updates, either with linear models or with non-linear models (Gaussian kernel with number of support vectors limited to 50 using the truncation approach of [12]).

4 Discussion

The dynamic protocol, extended to kernel methods, yields for the first time a theoretically efficient tool to learn non-linear models for distributed real-time services, in settings where communication is a major bottleneck. For that, it can employ online kernel methods together with model compression techniques, which reduce, or bound the number of support vectors. The efficiency of the protocol is characterized by a novel criterion that ties a tight loss bound to the required amount of communication—a criterion which is not satisfied by the state of the art of periodically communicating protocols.

While we provided a theoretical analysis, the advantage of the dynamic protocol in combination with kernel methods can also be shown in practice: Fig. 2 shows the results of an experiment on financial data [9], where 32 learners predicted the stock price of a target stock. We can see that for this difficult learning task linear models perform poorly compared to non-linear models using a Gaussian kernel function. Simultaneously, the communication required to periodically synchronize these non-linear models is larger than for linear models by more than two orders of magnitude. Using the dynamic protocol with kernel models we could reduce the error by an order of magnitude compared to using linear models (a reduction by a factor of 18). At the same time, the communication is reduced by more than three orders of magnitude compared to the static protocol (by a factor of 2433), which is yet an order of magnitude smaller than the communication when using linear models (by a factor of 10). Moreover, within less than 2000 rounds, the dynamic protocol reaches quiescence, as it is implied by the efficiency criterion.

A limit of the employed notion of efficiency is that it only takes into account the sum of messages but not the peak communication. In large data centers, where the distributed learning system is run next to other processes, the main bottleneck is the overall amount of transmitted bytes and a high peak in communication can often be handled by the communication infrastructure or evened out by a load balancer. In smaller systems, however, high peak communication can become a serious problem for the infrastructure and it remains an open problem how it can be reduced. Note that the frequency of synchronizations in a short time interval can actually be bounded by a trivial modification of the dynamic protocol: local conditions are only checked after a mini-batch of examples have been observed. Thus, the peak communication is upper bounded in the same way as with a periodic protocol, while still dynamically reducing the overall amount of communication.

When analyzing the reason for practical efficiency, model compression has proven to be a crucial factor, since storing and evaluating models with large numbers of support vectors can become infeasible—even in serial settings. In a distributed setting, transmitting large models furthermore induce high communication costs, which is aggravated by averaging local models, because the synchronized model consists of the union of all local support vectors. For the model truncation approach of [12], we have shown that the efficiency criterion is satisfied, but other model compression approaches might be favorable in certain scenarios. Thus, an interesting direction for future research is to study the relationship between loss and model size of those model compression techniques in order to extend the results on efficiency.

Also, alternative approaches to ensuring constant model size could be investigated, e.g., a finite dimensional approximation of the feature map $\Phi\colon X \to \mathcal{H}$ of a reproducing kernel Hilbert space $\mathcal{H}$, such as Random Fourier Features [17]. It remains an open problem how tight loss bounds combined with communication bounds can be derived in these settings.

Finding the right divergence threshold for the dynamic protocol, i.e., one that suits the desired trade-off between service quality and communication, is in practice a neither intuitive nor trivial task. The threshold can be selected using a small data sample, but the communication for a given threshold can vary over time and is also influenced by other parameters of the learner. Thus, another direction for future research is to investigate an adaptive divergence threshold. This could allow for a more direct selection of the desired trade-off between service quality (i.e., predictive performance) and communication.

Acknowledgments This research has been supported by the EU FP7-ICT-2013-11 under grant 619491 (FERARI).

Bibliography

[1] Barroso, L.A., Clidaras, J., Hölzle, U.: The datacenter as a computer: An introduction to the design of warehouse-scale machines. Synthesis lectures on computer architecture 8(3), 1–154 (2013)

[2] Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge university press (2004)

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

[4] Dekel, O., Gilad-Bachrach, R., Shamir, O., Xiao, L.: Optimal distributed online prediction using mini-batches. The Journal of Machine Learning Research 13, 165–202 (2012)

[5] Deligiannakis, A., Kotidis, Y., Roussopoulos, N.: Bandwidth-constrained queries in sensor networks. The VLDB Journal 17(3), 443–467 (2008)

[6] Gabel, M., Keren, D., Schuster, A.: Communication-efficient distributed variance monitoring and outlier detection for multivariate time series. In: Proceedings of the 28th International Parallel and Distributed Processing Symposium (IPDPS). pp. 37–47. IEEE (2014)

[7] Giatrakos, N., Deligiannakis, A., Garofalakis, M., Sharfman, I., Schuster, A.: Prediction-based geometric monitoring over distributed data streams. In: Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. pp. 265–276 (2012)

[8] Heires, K.: Budgeting for latency: If i shave a microsecond, will i see a 10x profit. Securities Industry News 22(1) (2010)

[9] Kamp, M., Boley, M., Gärtner, T.: Beating human analysts in nowcasting corporate earnings by using publicly available stock price and correlation features. In: 2013 IEEE 13th International Conference on Data Mining Workshops. pp. 384–390 (2013)

[10] Kamp, M., Boley, M., Keren, D., Schuster, A., Sharfman, I.: Communication-efficient distributed online prediction by dynamic model synchronization. In: Machine Learning and Knowledge Discovery in Databases, pp. 623–639. Springer (2014)

[11] Keren, D., Sharfman, I., Schuster, A., Livne, A.: Shape sensitive geometric monitoring. IEEE Transactions on Knowledge and Data Engineering 24(8), 1520–1535 (2012)

[12] Kivinen, J., Smola, A.J., Williamson, R.C.: Online learning with kernels. IEEE Transactions on Signal Processing 52(8), 2165–2176 (2004)

[13] Lichman, M.: UCI machine learning repository (2013), http://archive.ics.uci.edu/ml

[14] Mcdonald, R., Mohri, M., Silberman, N., Walker, D., Mann, G.S.: Efficient large-scale distributed training of conditional maximum entropy models. In: Advances in Neural Information Processing Systems. pp. 1231–1239 (2009)

[15] Orabona, F., Keshet, J., Caputo, B.: Bounded kernel-based online learning. The Journal of Machine Learning Research 10, 2643–2666 (2009)

[16] Predd, J.B., Kulkarni, S.R., Poor, H.V.: Distributed learning in wireless sensor networks. John Wiley & Sons: Chichester, UK (2007)
[17] Rahimi, A., Recht, B.: Random features for large-scale kernel machines. In: Advances in Neural Information Processing Systems (NIPS). pp. 1177–1184 (2007)
[18] Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT press (2001)
[19] Sharfman, I., Schuster, A., Keren, D.: A geometric approach to monitoring threshold functions over distributed data streams. Transactions on Database Systems (TODS) 32(4) (2007)
[20] Wang, Z., Vucetic, S.: Online passive-aggressive algorithms on a budget. In: International Conference on Artificial Intelligence and Statistics. pp. 908–915 (2010)
[21] Yuan, S., Wang, J., Zhao, X.: Real-time bidding for online advertising: measurement and analysis. In: Proceedings of the Seventh International Workshop on Data Mining for Online Advertising. p. 3 (2013)