Budgeted Nonparametric Learning from Data Streams

Ryan Gomes GOMES@VISION.CALTECH.EDU
Andreas Krause KRAUSEA@CALTECH.EDU

Abstract

We consider the problem of extracting informative exemplars from a data stream. Examples of this problem include exemplar-based clustering and nonparametric inference such as Gaussian process regression on massive data sets. We show that these problems require maximization of a submodular function that captures the informativeness of a set of exemplars, over a data stream. We develop an efficient algorithm, StreamGreedy, which is guaranteed to obtain a constant fraction of the value achieved by the optimal solution to this NP-hard optimization problem. We extensively evaluate our algorithm on large real-world data sets.

1. Introduction

Modern machine learning is increasingly confronted with the challenge of very large data sets. The unprecedented growth in text, video, and image data demands techniques that can effectively learn from large amounts of data, while still remaining computationally tractable. *Streaming* algorithms (Gaber et al., 2005; Domingos & Hulten, 2000; Guha et al., 2003; Charikar et al., 2003) represent an attractive approach to handling the data deluge. In this model the learning system has access to a small fraction of the data set at any point in time, and cannot necessarily control the order in which the examples are visited. This is particularly useful when the data set is too large to fit in primary memory, or if it is generated in real time and predictions are needed in a timely fashion.

While computational tractability is critical, powerful methods are required in order to learn useful models of complex data. Nonparametric learning methods are promising because they can construct complex decision rules by allowing the data to "speak for itself". They may use complex similarity measures that capture domain knowledge while still providing more flexibility than parametric methods. However, nonparametric techniques are difficult to apply to large datasets because they typically associate a parameter with every data point, and thus depend on all the data. Therefore, most algorithms for nonparametric learning operate in batch mode. To overcome this difficulty, nonparametric learning methods may be approximated by specifying a budget: a fixed limit on the number of examples that are used to make predictions.

In this work, we develop a framework for budgeted nonparametric learning that can operate in a streaming data environment. In particular, we study sparse Gaussian process regression and exemplar based clustering under complex, non-metric distance functions, which both meet the requirements of our framework. The unifying concept of our approach is *submodularity*, an intuitive diminishing returns property. When a nonparametric problem's objective function satisfies this property, we show that a simple algorithm, StreamGreedy, may be used to choose examples from a data stream. We use submodularity to prove strong theoretical guarantees for our algorithm. We demonstrate our approach with experiments involving sparse Gaussian Process regression and large scale exemplar-based clustering of 1.5 million images.

2. Problem statement

We consider the problem of extracting a subset $\mathcal{A} \subseteq \mathcal{V}$ of k representative items from a large data set $\mathcal{V}$ (which can, e.g., consist of vectors in $\mathbb{R}^d$ or other objects such as graphs, lists, etc.). Our goal is to maximize a set function F that quantifies the utility $F(\mathcal{A})$ of any possible subset $\mathcal{A} \subseteq \mathcal{V}$. We give examples of such utility functions in Sec. 3. Intuitively, in the clustering example, $F(\mathcal{A})$ measures, e.g., the reduction in quantization error when selecting exemplars $\mathcal{A}$ as cluster centers. In Gaussian process (GP) regression, $F(\mathcal{A})$ measures the prediction performance when selecting the active set $\mathcal{A}$. As we show below, many utility functions, such as those arising in clustering and GP

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

regression, satisfy *submodularity*, an intuitive diminishing returns property: Adding a cluster center helps more if we have selected few exemplars so far, and less if we have already selected many exemplars. Formally, a set function F is said to be *submodular*, if for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$ and $s \in \mathcal{V} \setminus \mathcal{B}$ it holds that $F(\mathcal{A} \cup \{s\}) - F(\mathcal{A}) \geq F(\mathcal{B} \cup \{s\}) - F(\mathcal{B})$. An additional natural assumption is that F is *monotonic*, i.e., $F(\mathcal{A}) \leq F(\mathcal{B})$ whenever $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$.

Since the data set $\mathcal{V}$ is large, it is not possible to store it in memory, and we hence can only access a small number of items at any given time t. Let $\mathcal{B}_1, \dots, \mathcal{B}_T, \dots$ be a sequence of subsets of $\mathcal{V}$, where $\mathcal{B}_t$ is the set of elements in $\mathcal{V}$ that are available to the algorithm at time t. Typically $|\mathcal{B}_t| = m \ll n = |\mathcal{V}|$. For example, hardware limitations may require us to read data from disk, one block $\mathcal{B}_t$ of data points at a time.

We only assume that there is a bound ρ, such that for each element $b \in \mathcal{V}$, if $b \notin \mathcal{B}_t \cup \dots \cup \mathcal{B}_{t+\ell}$, then $\ell < \rho$, i.e., we have to wait at most ρ steps until b reappears. This assumption is satisfied, for example, if $\mathcal{B}_t$ is a sliding window over the data set (in which case $\rho = n$), or $\mathcal{V}$ is partitioned into blocks, and the $\mathcal{B}_t$ cycle through these blocks (in which case ρ is $n/(\min_i |\mathcal{B}_i|)$). Our goal is to select at each time t a subset $\mathcal{A}_t \subseteq \mathcal{A}_{t-1} \cup \mathcal{B}_t$, $|\mathcal{A}_t| \leq k$, in order to maximize $F(\mathcal{A}_T)$ after some number of iterations T. Thus, at each time t we are allowed to pick any combination of k items from both the previous selection $\mathcal{A}_{t-1}$ and the available items $\mathcal{B}_t$, and we would like to maximize the final value $F(\mathcal{A}_T)$.

Our streaming assumptions mirror those in (Charikar et al., 2003), in that we assume a finite data set in which data items may be revisited although the order is not under our control. For certain submodular objectives (F_V and F_C but not F_H, see section 3) we require the additional assumption that we may access data items uniformly at random (see section 4). Note that even if $\mathcal{B}_1 = \dots = \mathcal{B}_T = \mathcal{V}$, i.e., access to the entire data set is always available, the problem of choosing a set

$$\mathcal{A}^* = \operatorname*{argmax}_{|\mathcal{A}| \leq k} F(\mathcal{A})$$

maximizing a submodular function F is an NP-hard optimization problem (Feige, 1998). Hence, we cannot expect to efficiently find the optimal solution in general. The setting where $\mathcal{B}_t \subsetneq \mathcal{V}$ is strictly more general and thus harder. In this paper, we will develop an efficient approximation algorithm with strong theoretical guarantees for this problem.

3. Examples of online budgeted learning

In this section, we discuss concrete problem instances of the streaming budgeted learning problem, and the corresponding submodular objective functions F.

Active set selection in GPs. Gaussian processes have been widely used as a powerful tool for nonparametric regression (Rasmussen & Williams, 2006; Cressie, 1991). Formally, a Gaussian process (GP) is a joint probability distribution $P(\mathcal{X}_\mathcal{V})$ over a (possibly infinite) set of random variables $\mathcal{X}_\mathcal{V}$ indexed by a set $\mathcal{V}$, with the property that every finite subset $\mathcal{X}_\mathcal{A}$ for $\mathcal{A} = \{s_1, \dots, s_k\}$, $\mathcal{A} \subseteq \mathcal{V}$ is distributed according to a multivariate normal distribution, $P(\mathcal{X}_\mathcal{A} = \mathbf{x}_\mathcal{A}) = \mathcal{N}(\mathbf{x}_\mathcal{A}; \mu_\mathcal{A}, \Sigma_{\mathcal{A}\mathcal{A}})$, where $\mu_\mathcal{A} = (\mathcal{M}(s_1), \dots, \mathcal{M}(s_k))$ is the prior mean and

$$\Sigma_{\mathcal{A}\mathcal{A}} = \begin{pmatrix} \mathcal{K}(s_1, s_1) & \dots & \mathcal{K}(s_1, s_k) \\ \vdots & & \vdots \\ \mathcal{K}(s_k, s_1) & \dots & \mathcal{K}(s_k, s_k) \end{pmatrix}$$

is the prior covariance, parameterized through the positive definite kernel function $\mathcal{K}$. In GP regression, each data point $s \in \mathcal{V}$ is interpreted as a random variable in a GP. Based on observations $\mathcal{X}_\mathcal{A} = \mathbf{x}_\mathcal{A}$ of a subset $\mathcal{A}$ of variables, the predictive distribution of a new data point $s \in \mathcal{V}$ is a normal distribution $P(\mathcal{X}_s \mid \mathcal{X}_\mathcal{A} = \mathbf{x}_\mathcal{A}) = \mathcal{N}(\mu_{s|\mathcal{A}}; \sigma^2_{s|\mathcal{A}})$, where

$$\mu_{s|\mathcal{A}} = \mu_s + \Sigma_{s\mathcal{A}} \Sigma^{-1}_{\mathcal{A}\mathcal{A}} (\mathbf{x}_\mathcal{A} - \mu_\mathcal{A}) \tag{3.1}$$

$$\sigma^2_{s|\mathcal{A}} = \sigma^2_s - \Sigma_{s\mathcal{A}} \Sigma^{-1}_{\mathcal{A}\mathcal{A}} \Sigma_{\mathcal{A}s}, \tag{3.2}$$

and $\Sigma_{s\mathcal{A}} = (\mathcal{K}(s, s_1), \dots, \mathcal{K}(s, s_k))$ and $\Sigma_{\mathcal{A}s} = \Sigma^T_{s\mathcal{A}}$. Computing the predictive distributions according to (3.1) is expensive, as it requires "inverting" (finding the Cholesky decomposition) of the kernel matrix $\Sigma_{\mathcal{A}\mathcal{A}}$, which, in general requires $\Theta(|\mathcal{A}|^3)$ floating point operations. Reducing this computational complexity (and thereby enabling GP methods for large data sets) has been subject of much research (see Rasmussen & Williams 2006).

Most approaches for efficient inference in GPs rely on choosing a small *active set* $\mathcal{A}$ of data points for making predictions. For example, the informative vector machine (IVM) uses the set $\mathcal{A}$ that maximizes the information gain

$$F_H(\mathcal{A}) = H(\mathcal{X}_\mathcal{V}) - H(\mathcal{X}_\mathcal{V} \mid \mathcal{X}_\mathcal{A}), \tag{3.3}$$

or, equivalently, the entropy $H(\mathcal{X}_\mathcal{A})$ of the random variables associated with the selected data points $\mathcal{A}$. It can be shown, that this criterion is monotonic and submodular (Seeger, 2004). While efficiently computable, the IVM criterion F_H only depends on the selected data points, and does not explicitly optimize the prediction error of the non-selected examples $\mathcal{V} \setminus \mathcal{A}$.

An alternative is to choose data points which minimize the prediction accuracy on the non-selected data: $\widehat{L}(\mathcal{A}) = \sum_{s \in \mathcal{V} \setminus \mathcal{A}} (\mathbf{x}_s - \mu_{s|\mathcal{A}})^2$. If the data points $\mathcal{V}$ are drawn from some distribution $P(s)$, then this criterion can be seen as a sample approximation to the expected variance reduction,

$$\widehat{L}(\mathcal{A}) \approx \int P(s) \int P(\mathbf{x}_s \mid \mathbf{x}_\mathcal{A})(\mathbf{x}_s - \mu_{s|\mathcal{A}})^2 ds d\mathbf{x}_s$$
$$= \int P(s) \sigma^2_{s|\mathcal{A}} d\mathbf{x}_s = L(\mathcal{A}).$$

It can be shown, that under certain assumptions on the kernel function, the *expected variance reduction*

$$F_V(\mathcal{A}) = L(\emptyset) - L(\mathcal{A}) \quad (3.4)$$

is a monotonic submodular function.

Exemplar based clustering with complex distance functions on data streams. In exemplar clustering problems, the goal is to select a set of examples from the data set that are representative of the data set as a whole. Exemplar clustering is particularly relevant in cases where choosing cluster centers that are averages of training examples (as in the k-means algorithm) is inappropriate or impossible (see Dueck & Frey 2007 for examples). The k-medoid (Kaufman & Rousseeuw, 1990) approach seeks to choose exemplars that minimize the average dissimilarity of the data items to their nearest exemplar:

$$L(\mathcal{A}) = \frac{1}{|\mathcal{V}|} \sum_{s \in \mathcal{V}} \min_{c \in \mathcal{A}} d(\mathbf{x}_s, \mathbf{x}_c). \quad (3.5)$$

This loss function can be transformed to a monotonic submodular utility function by introducing a *phantom exemplar* $\mathbf{x}_0$ which may not be removed from the active set, and defining the utility function

$$F_C(\mathcal{A}) = L(\{\mathbf{x}_0\}) - L(\mathcal{A} \cup \{\mathbf{x}_0\}). \quad (3.6)$$

This measures the decrease in the loss associated with the active set versus the loss associated with just the phantom exemplar, and maximizing this function is equivalent to minimizing (3.5). The dissimilarity function $d(\mathbf{x}, \mathbf{x}')$ need only be a positive function of $\mathbf{x}$ and $\mathbf{x}'$, making this approach potentially very powerful.

4. StreamGreedy for budgeted learning from data streams

If, at every time, full access to the entire data set $\mathcal{V}$ is available, a simple approach to selecting the subset $\mathcal{A}_T$ would be to start with the empty set, $\mathcal{A}_0 = \emptyset$, and, at iteration t, greedily select the element

$$s_t = \operatorname*{argmax}_{s \in \mathcal{V}} F(\mathcal{A}_{t-1} \cup \{s\}) \quad (4.1)$$

for $t \leq k$, and $\mathcal{A}_t = \mathcal{A}_{t-1}$ for $t > k$. Perhaps surprisingly, this simple greedy algorithm is guaranteed to obtain a near-optimal solution: Nemhauser et al. (1978) prove that for the solution $\mathcal{A}_T$, for any $T \geq k$, obtained by the greedy algorithm it holds that $F(\mathcal{A}_T) \geq (1 - 1/e) \max_{|\mathcal{A}| \leq k} F(\mathcal{A})$, i.e., it achieves at least a constant fraction of $(1 - 1/e)$ of the optimal value. In fact, no efficient algorithms can provide better approximation guarantees unless P=NP (Feige, 1998).

Unfortunately, the greedy selection rule (4.1) requires access to all elements of $\mathcal{V}$, and hence cannot be applied in the streaming setting. A natural extension to the streaming setting is the following algorithm: Initialize $\mathcal{A}_0 = \emptyset$. For $t \leq k$, set $\mathcal{A}_t \leftarrow \mathcal{A}_{t-1} \cup \{s_t\}$, where

$$s_t = \operatorname*{argmax}_{s \in \mathcal{B}_t} F(\mathcal{A}_{t-1} \cup \{s\}). \quad (4.2)$$

For $t > k$, let

$$(s', s) = \operatorname*{argmax}_{s' \in \mathcal{A}_{t-1}, s \in \mathcal{A}_{t-1} \cup \mathcal{B}_t} F(\mathcal{A}_{t-1} \setminus \{s'\} \cup \{s\}), \quad (4.3)$$

and set $\mathcal{A}_t = \mathcal{A}_{t-1} \setminus \{s'\} \cup \{s\}$, i.e., replace item s' by item s in order to greedily maximize the utility. Stop after no significant improvement (at least η for some small value $\eta > 0$) is observed after a specified number ρ of iterations. StreamGreedy is summarized in Algorithm 1.

Algorithm 1 StreamGreedy

```
Initialize active set $\mathcal{A}_0 = \emptyset$; Bound $\rho$ on wait time
for $t = 1 : k$ do
    Set $s_t = \operatorname{argmax}_{s \in \mathcal{B}_t} F(\mathcal{A}_{t-1} \cup \{s\})$
    Set $\mathcal{A}_t \leftarrow \mathcal{A}_{t-1} \cup \{s_t\}$
end for
Set $NI = 0$
while $NI \leq \rho$ do
    Set $(s', s) = \operatorname*{argmax}_{s' \in \mathcal{A}_{t-1}, s \in \mathcal{A}_{t-1} \cup \mathcal{B}_t} F(\mathcal{A}_{t-1} \setminus \{s'\} \cup \{s\})$
    Set $t \leftarrow t + 1$; $\mathcal{A}_t = \mathcal{A}_{t-1} \setminus \{s'\} \cup \{s\}$
    if $F(\mathcal{A}_t) > F(\mathcal{A}_{t-1}) + \eta$ then
        Set $NI = 0$
    else
        Set $NI = NI + 1$
    end if
end while
```

Dealing with limited access to the stream. So far, we have assumed that StreamGreedy can evaluate the objective function F for any candidate set $\mathcal{A}$. While the IVM objective $F_H(\mathcal{A})$ for active set selection in GPs (see Section 3) only requires access to the selected data points $\mathcal{A}$, evaluating the objectives F_C and F_V requires access to the entire data set $\mathcal{V}$. However, these objective functions share a key property: They additively decompose over the data set. Hence, they can be written in the form $F(\mathcal{A}) = \frac{1}{|\mathcal{V}|}\sum_{s\in\mathcal{V}} f(\mathcal{A}, \mathbf{x}_s)$ for suitable function f such that $f(\cdot, \mathbf{x}_s)$ is submodular for each input $\mathbf{x}_s$. If we assume that data points $\mathbf{x}_s$ are generated i.i.d. from a distribution and f is a measurable function of $\mathbf{x}_s$, then $f(\mathcal{A}, \mathbf{x}_s)$ are themselves a series of i.i.d. outcomes of a random variable. Moreover, the range of random variables $f(\mathcal{A}, \mathbf{x}_s)$ is bounded by some constant B (for clustering, B is the diameter of the data set; for GP regression, B is the maximum prior marginal variance). We can construct a sample approximation $\widehat{F}(\mathcal{A}) = \frac{1}{|\mathcal{W}|}\sum_{s\in\mathcal{W}} f(\mathcal{A}, \mathbf{x}_s)$ by choosing a validation set $\mathcal{W}$ uniformly at random from the stream $\mathcal{V}$. The following corollary of Hoeffding's inequality adapted from Smola et al. (1999) bounds the deviation between $\widehat{F}(\mathcal{A})$ and $F(\mathcal{A})$:

Corollary 1 (Smola et al. 1999)**.** *Let* $c = \frac{B^2 \log(\frac{2}{\delta})}{2|\mathcal{V}|\varepsilon^2}$ *and* $\delta > 0$*. Then, with probability* $1-\delta$ *for* $|\mathcal{W}| = \frac{c}{1+c}|\mathcal{V}|$*:*

$$\left|\frac{1}{|\mathcal{W}|}\widehat{F}(\mathcal{A}) - \frac{1}{|\mathcal{V}|}F(\mathcal{A})\right| < \varepsilon$$

The result relates the level of approximation to the fraction of the data set that is needed for validation. As the number of elements in the stream $|\mathcal{V}|$ increases, smaller fractions are needed to reach a given accuracy. Because this result holds for any (bounded) data distribution, it is usually pessimistic; in practice, smaller validation sets often suffice.

Furthermore, this sample based approximation only requires a constant amount of memory: When $\mathbf{x}_s$ arrives from the stream, $f(\mathcal{A}, \mathbf{x}_s)$ may be added to a sufficient statistic and $\mathbf{x}_s$ itself may be discarded.

5. Theoretical analysis

Clustering-consistent objectives. For clarity of notation, we will consider the setting where $\mathcal{B}_t = \{b_t\}$ contains only a single element $b_t \in \mathcal{V}$. The results generalize to sets $\mathcal{B}_t$ containing more elements.

We first show that for an interesting class of submodular functions, the algorithm actually converges to the optimal solution. Suppose, the data set $\mathcal{V}$ can be partitioned into a set of clusters, i.e., $\mathcal{V} = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_L$, where $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$. We call a monotonic submodular function F *clustering-consistent* for a particular clustering $\mathcal{C}_1, \ldots, \mathcal{C}_L$, if the following conditions hold:

1. $F(\mathcal{A}) = \sum_{\ell=1}^{L} F(\mathcal{A} \cap \mathcal{C}_\ell)$, i.e., F decomposes additively across clusters.

2. Whenever for two sets $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{B} = \mathcal{A} \cup \{s\} \setminus \{s'\}$, $s \in \mathcal{C}_i$, $s' \in \mathcal{C}_j$, $i \neq j$ it holds that if $|\mathcal{A} \cap \mathcal{C}_j| > 1$ and $\mathcal{A} \cap \mathcal{C}_i = \emptyset$, then $F(\mathcal{A}) \leq F(\mathcal{B})$.

Intuitively, a submodular function F is clustering-consistent, if it is always preferable to select a representative from a new cluster than having two representatives of the same cluster.

Proposition 2. *Suppose F is clustering-consistent for $\mathcal{V}$ and $k \leq L$. Then, for $T = 2\rho$ it holds for all sets $\mathcal{A}_t$, $t \geq T$ returned by* StreamGreedy *(for $\eta = 0$) that*

$$F(\mathcal{A}_t) = \max_{|\mathcal{A}|\leq k} F(\mathcal{A}).$$

The proofs can be found in the Appendix. Thus, for clustering-consistent objectives F, if the data set really consists of L clusters, and we use StreamGreedy to select a set of $k \leq L$ exemplars, then StreamGreedy converges to the optimal solution after at most two passes through the data set $\mathcal{V}$.

Of course the question is which classes of objective functions are clustering-consistent. In the following, suppose that the elements in $\mathcal{V}$ are endowed with a metric d. The following proposition gives interesting examples:

Proposition 3. *Suppose $\mathcal{V} = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_L$, $|\mathcal{C}_i| < \alpha|\mathcal{C}_j|$ for all i, j. Further suppose that*

$$\max_i \operatorname{diam}(\mathcal{C}_i) < \beta \min_{i,j} d(\mathcal{C}_i, \mathcal{C}_j)$$

for suitable constants α and β, where $d(\mathcal{C}_i, \mathcal{C}_j) = \min_{r\in\mathcal{C}_i, s\in\mathcal{C}_j} d(r,s)$ and $\operatorname{diam}(\mathcal{C}_i) = \max_{r,s\in\mathcal{C}_i} d(r,s)$. Then the following objectives from Sec. 3 are clustering-consistent with $\mathcal{V} = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_L$:

- *The clustering objective F_C, whenever $\max_{\mathbf{x}\in\mathcal{C}_i} d(\mathbf{x}, \mathbf{x}_0) \leq \min_j d(\mathcal{C}_i, \mathcal{C}_j)$ for all i, j, where $\mathbf{x}_0$ is the phantom exemplar.*

- *The entropy F_H and variance reduction*[1] *F_V for Gaussian process regression with squared exponential kernel functions with appropriate bandwidth σ^2, and where d is the Euclidean metric in $\mathbb{R}^d$.*

[1] under the condition of conditional suppressor-freeness (Das & Kempe, 2008)

Intuitively, Propositions 2 and 3 suggests that in situations where the data actually exhibits a well-separated, balanced clustering structure, and we are interested in selecting a number of exemplars k consistent with the number of clusters L in the data, we expect StreamGreedy to perform near-optimally.

General submodular objectives. However, the assumptions made by Propositions 2 and 3 are fairly strong, and likely violated by the existence of outliers, overlapping and imbalanced clusters, etc. Furthermore, when using criteria such as F_C and F_V (Sec. 3), it is not possible to evaluate $F(\mathcal{A})$ exactly, but only up to additive error ε. Perhaps surprisingly, even in such more challenging settings, the algorithm is still guaranteed to converge to a near-optimal solution:

Theorem 4. *Let $\eta > 0$. Suppose F is monotonic submodular on $\mathcal{V}$, and we have access to a function $\widehat{F}$ such that for all $\mathcal{A} \subseteq \mathcal{V}$, $|\mathcal{A}| \leq 2k$ it holds that $|\widehat{F}(\mathcal{A}) - F(\mathcal{A})| \leq \varepsilon$. Furthermore suppose F is bounded by B. Then, for $T = \rho B/\eta$ it holds for all sets $\mathcal{A}_t$, $t \geq T$ selected by* StreamGreedy *applied to $\widehat{F}$ that*

$$F(\mathcal{A}_t) \geq \frac{1}{2} \max_{|\mathcal{A}| \leq k} F(\mathcal{A}) - k(\varepsilon + \eta).$$

Thus, e.g., in the case where $b_t = s_{t \bmod n}$, i.e., if StreamGreedy sequentially cycles through the data set $\mathcal{V}$, at most B/η passes (typically it will stop far earlier) through the data set will suffice to produce a solution that obtains almost half the optimal value. The proof relies on properties of the pairwise exchange heuristic for submodular functions (Nemhauser et al., 1978). See the Appendix for details.

6. Experimental results

Exemplar based streaming clustering. Our exemplar based clustering experiments involve StreamGreedy applied to the clustering utility F_C (Eq. (3.6)) with $d(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'||^2$. The implementation can be made efficient by exploiting the fact that only a subset of the validation points (*c.f.*, Sec. 4) change cluster membership for each candidate swap. We have also implemented an adaptive stopping rule that is useful when determining an appropriate size of the validation set. Please see Appendix II for details.

Our first set of experiments uses MNIST handwritten digits with 60,000 training images and 10,000 test images.[2] The MNIST digits were preprocessed as follows: The 28 by 28 pixel images are initially represented as 784 dimensional vectors, and the mean of the training image vectors was subtracted from each image; then the resulting vectors are normalized to unit norm. PCA was performed on the normalized training vectors and the first 50 principal components coefficients were used to form feature vectors. The same normalization procedure was performed on the test images and their dimensionality was also reduced using the training PCA basis.

Fig. 1 compares the performance of our approach against batch k-means and online k-means (Dasgupta, 2009) with the number of exemplars set to K = 100. We chose the origin as the *phantom exemplar* in this experiment, since this yielded better overall quantization performance than choosing a random exemplar. To unambiguously assess convergence speed we use the entire training set of 60,000 points as the validation set. We assess convergence by plotting (3.6) against the number of swap candidates ($\sum_{t=1}^{T} |\mathcal{B}_t|$) considered. We find that our algorithm converges to a solution after examining nearly the same number of data points as online k-means, and is near its final value after a single pass through the training data. Similar convergence was observed for smaller validation sizes. The left plot in Fig. 1 shows that k-means performs better in terms of quantization loss. This is probably because StreamGreedy must choose exemplar centers from the training data, while k-means center locations are unconstrained. When the k-means' centers are replaced with the nearest training example (center plot), the advantage disappears. The right plot in Fig. 1 examines the impact of validation set size on quantization performance on the held out test set, measured as test set utility ((3.6) where $\mathcal{V}$ is the test set). It is possible to obtain good generalization performance even when using a small validation set. The y-axis indicates test performance relative to the performance attained with the full data set at the specified value of K (1.0 indicates equal performance, values less than one indicate worse performance than the full set), and the x-axis is plotted as the relative size of the validation set versus the full set. We find that as the number of centers K increases, a larger fraction of the data set is needed to approach the performance with the full set. This appears to be because as K increases, the numerical differences between $F_C(\mathcal{A}_{t-1} \setminus \{s'\} \cup \{s\})$ for alternative candidate swaps (s, s') decrease, and more samples are needed in order to stably rank the swap alternatives.

Our second set of experiments involves approximately 1.5 million *Tiny Images*[3] (Torralba et al., 2008), and

[2] MNIST was downloaded from http://yann.lecun.com/exdb/mnist/.

[3] `http://people.csail.mit.edu/torralba/tinyimages/`.

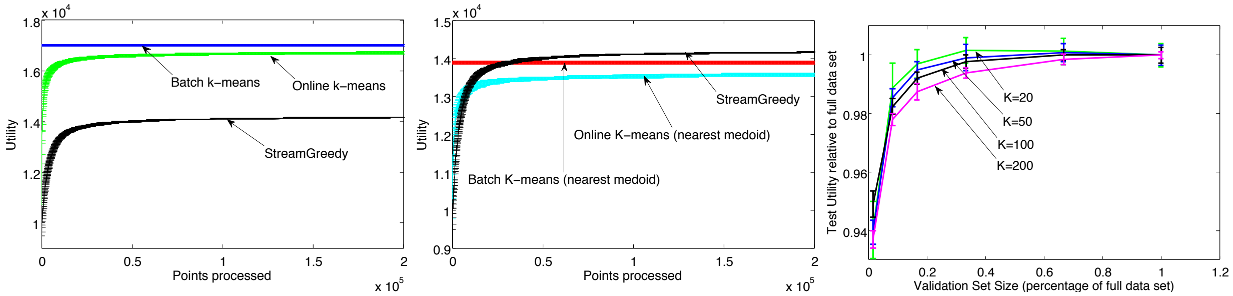

Figure 1. Left and Center: Convergence rates on MNIST data set. The y-axis represents the clustering utility evaluated on the training set. The x-axis shows the number of data items processed by StreamGreedy and online k-means. K-means' unconstrained centers yield better quantization performance. When k-means' centers are replaced with the nearest training set example, the advantage disappears (center). Right: Test performance versus validation set size. It is possible to obtain good generalization performance even using relatively small validation sets. The validation set size is varied along the x-axis. The y-axis shows test utility divided by the test utility achieved with the entire data set used for validation. As K increases, more validation data is needed to achieve full performance.

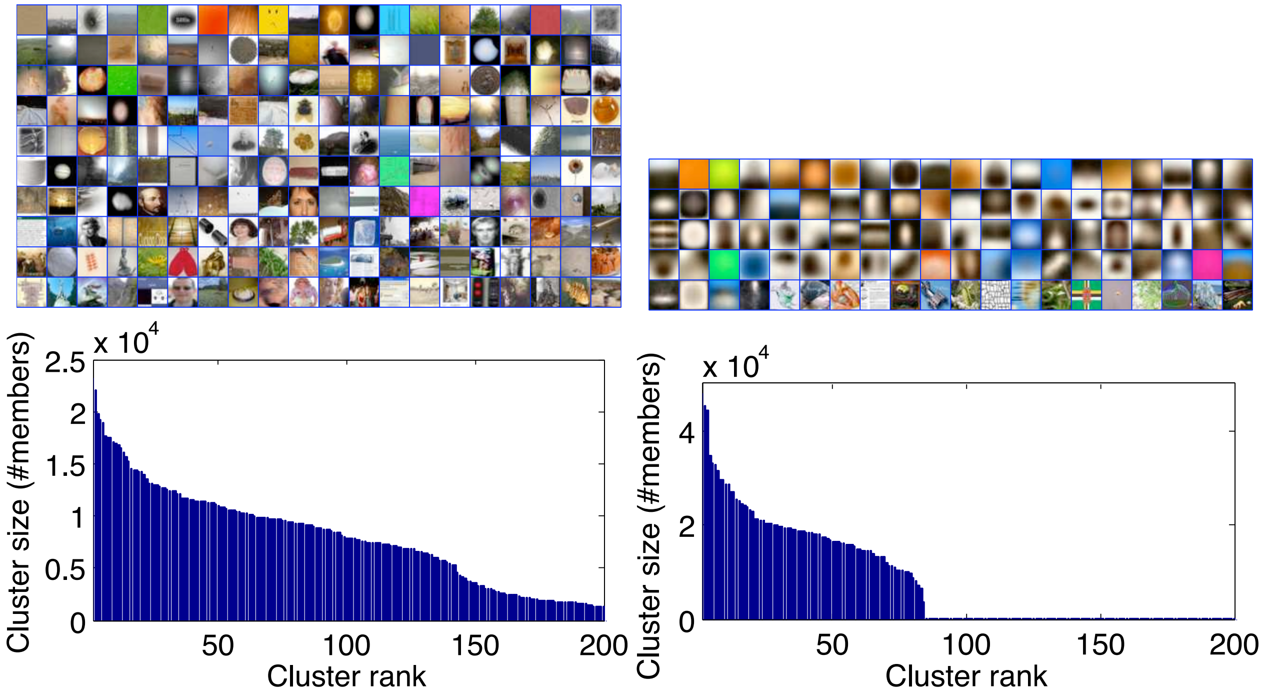

Figure 2. Tiny Image data set. Top Left: Cluster exemplars discovered by StreamGreedy, sorted according to descending size. Top Right: Cluster centers from online kmeans (singleton clusters omitted). Bottom Left: Cluster sizes (number of members) for our algorithm. Bottom Right: Cluster sizes for online k-means. Online k-means finds a poor local minima with many of the 200 clusters containing only a single member.

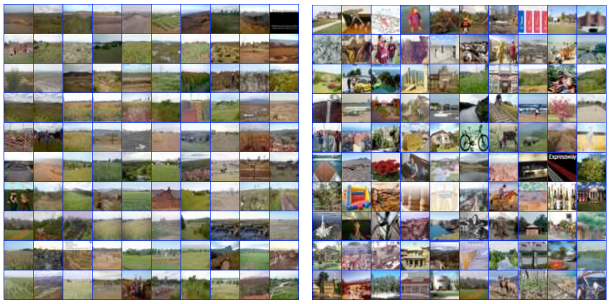

Figure 3. Examples from Tiny Image cluster 26. Left: 100 examples nearest to exemplar 26. Right: 100 randomly sampled images from cluster 26.

is designed to test our algorithm on a large scale data set. Each image in the data set was downloaded by Torralba et al. from an Internet search engine and is associated with an English noun query term. The 32 by 32 RGB pixel images are represented as 3,072 dimensional vectors. Following Torralba et al. (2008), we subtract from each vector its mean value (average of all components), then normalize it to unit norm. No dimensionality reduction is performed. We generate a random center to serve as the *phantom exemplar* for this experiment, since we find that this leads to qualitatively more interesting clusters than using the origin. [4]

Fig. 2 (left) shows $K = 200$ exemplars discovered by our algorithm. Clusters are organized primarily according to non-semantic visual characterstics such as color and basic shape owing to the simple sum of squared differences similarity measure employed (Fig. 3). We set the validation size to one-fifth of the data set. This was determined by examining the stability of $\text{argmax}_{s' \in \mathcal{A}_{t-1}, s \in \mathcal{A}_{t-1} \cup \mathcal{B}_t} F_C(\mathcal{A}_{t-1} \setminus \{s'\} \cup \{s\})$ as validation data was progressively added to the sums in F_C, which tends to stabilize well before this amount of data is considered. The algorithm was halted after 600 iterations (each considering $|\mathcal{B}_t| = 1,000$ candidate centers). This was determined based on inspection of the utility function, which converged before a single pass through the data. We compare against the online k-means algorithm with 200 centers initialized to randomly chosen images, and run through a single pass over the data. We find that online k-means converges to a suboptimal solution in which many of the clusters are empty or contain only a single member (see Fig. 2.)

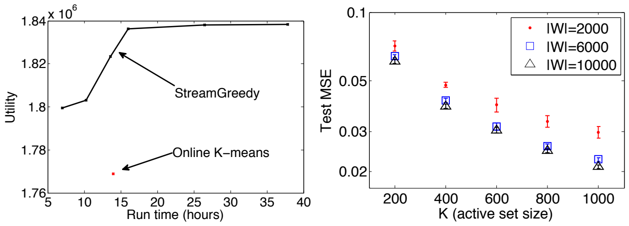

Figure 4. Left: Algorithm utility score versus run time on the Tiny Images data set. Right: Gaussian Process regression. y-axis is test set mean squared prediction error. x-axis is the size of the active set.

In Fig. 4 (left) we assess the tradeoff between run time and performance by varying the parameter $|\mathcal{B}_t| = \{500, 1000, 2000\}$ and the validation set size as $\{10\%, 20\%, 40\%\}$ of the data set. The number of centers and iterations are fixed at 200 and 600, respectively. Our Matlab StreamGreedy implementation was run on a quad-core Intel Xeon server. Performance is visualized as a point in the test utility versus run time plane, and only the Pareto optimal points are displayed for clarity. Online k-means is also shown for comparison. We find a clear saturation in performance as run time increases.

Online active set selection for GP regression. Our Gaussian Process regression experiments involve specialization of StreamGreedy for the objective function F_V in Sec. 3. The implementation can be made more efficient by using Cholesky factorization on the covariance matrix combined with rank one updates and downdates. Please see Appendix II for details. We used the KIN40K dataset[5] which consists of 9 attributes generated by a robotic arm simulator. We divide the dataset into 10,000 training and 30,000 test instances. We follow the preprocessing steps outlined in (Seeger et al., 2003) in order to compare our approach to the results in that study. We used the squared exponential kernel with automatic relevance determination (ARD) weights and learn the hyperparameters using marginal likelihood maximization (Rasmussen & Williams, 2006) on a subset of 2,000 training points, again following (Seeger et al., 2003).

Fig. 4 (right) shows the mean squared error predictive performance $\frac{1}{2}\sum_s (y_s - \mu_s)$ on the test set as a function of the size of the active set. Comparing our results to the experiments in (Seeger et al., 2003), we find that our approach outperforms the info-gain criterion for active set size $K = \{200, 400, 600\}$ at all values

[4] We find that a random phantom exemplar is unlikely to be chosen as a prototype, while one near the origin is the prototype for a significant fraction of the data.

[5] Downloaded from http://ida.first.fraunhofer.de/ anton/data.html.

of the validation set size $|\mathcal{W}| = \{2000, 6000, 10000\}$. At values $K = \{800, 1000\}$ our approach outperforms info-gain for $|\mathcal{W}| = \{6000, 10000\}$. Our performance matches Smola and Bartlett (Smola & Bartlett, 2000) at $K = \{200, 400\}$ but slightly underperforms their approach at larger values of K. We find that even for $|\mathcal{W}| = 2,000$, the algorithm is able to gain predictive ability by choosing more active examples from the data stream. The performance gap between $|\mathcal{W}| = 6,000$ and $|\mathcal{W}| = 10,000$ is quite small.

7. Related Work

Specialization of StreamGreedy to the clustering objective F_C (3.6) yields an algorithm which is similar to the Partitioning Around Medoids (PAM, Kaufman & Rousseeuw 1990) algorithm for k-medoids, and related algorithms CLARA (Kaufman & Rousseeuw, 1990) and CLARANS (Ng & Han, 2002). Like our approach, these algorithms are based on repeatedly exchanging centers for non-center data points if the swap improves the objective function. Unlike our approach, however, no performance guarantees are known for these approaches. PAM requires access to the entire data set, and every data point is exhaustively examined at each iteration, leading to an approach unsuitable for large databases. CLARA runs PAM repeatedly on subsamples of the data set, but then makes use of the entire dataset when comparing the results of each PAM run. Like our algorithm, CLARANS evaluates a random subset of candidate centers at each iteration, but then makes use of the entire data set to evaluate candidate swaps. Our approach takes advantage of the i.i.d. concentration behavior of the clustering objective in order to eliminate the need for accessing the entire data set, while still yielding a performance guarantee. Domingos & Hulten (2001) exploit the concentration behavior of the (non-exemplar) k-means objective in a similar way. While there exist online algorithms for k-medoids with strong theoretical guarantees (Charikar et al., 2003), these algorithms require the distance function d to be a metric, and the memory to grow (logarithmically) in $|\mathcal{V}|$. In contrast, our approach uses arbitrary dissimilarity functions and the memory requirements are independent of the data set size.

Specialization of StreamGreedy to sparse GP inference is an example of the *subset of datapoints* class of sparse Gaussian Process approximations (Rasmussen & Williams, 2006), in which the GP predictive distribution is conditioned on only the datapoints in the active set. Seeger et al. (2003) also use a *subset of datapoints* approach that makes use of a submodular (Seeger, 2004) utility function (the entropy of the Gaussian distribution of each site in the active set). This approach is computationally cheaper than ours in that the evaluation criterion does not require a validation set, but depends only on the current active set. Seeger et al.'s approach also fits the framework proposed by this paper, and our approach could be used to optimize this objective over data streams. Smola & Bartlett (2000) use a *subset of regressors* approach. Their criterion for greedy selection of regressors has the same complexity as our approach if we use the entire data set for validation. Our approach is cheaper when we make use of a limited validation set. Csató & Opper (2002) develop an approach for online sparse GP inference based on *projected process* approximation that also involves swapping candidate examples into an active set, but without performance guarantees. See Rasmussen & Williams (2006) for a survey of other methods for sparse Gaussian Process approximation.

StreamGreedy's structure is similar to the algorithm by Weston et al. (2005) for online learning of kernel perceptron classifiers, in that both approaches make use of a fixed budget of training examples (the active set) that are selected by evaluating a loss function defined over a limited validation set.

Nemhauser et al. (1978) analyzed the greedy algorithm and a pairwise exchange algorithm for maximizing submodular functions. As argued in Sec. 4, these algorithms do not apply to the streaming setting. Streeter & Golovin (2008) develop an online algorithm for maximizing a sequence of submodular functions over a fixed set (that needs to be accessed every iteration). Our approach, in contrast, maximizes a single submodular function on a sequence of sets, using bounded memory.

8. Conclusions

We have developed a theoretical framework for extracting informative exemplars from data streams that led to StreamGreedy, an effective algorithm with strong theoretical guarantees. We have shown that this framework can be successfully specialized to exemplar based problems and nonparametric regression with Gaussian Processes. In the case of clustering, our experiments demonstrate that our approach is capable of discovering meaningful clusters in large high-dimensional data sets, while remaining computationally tractable. Our sparse Gaussian Process regression algorithm is competitive with respect to other approaches and is capable of operating in a streaming data environment. Future work involves discovering other machine learning problems that fit the framework (including classification) and exploring alternative ways to approximately evaluate submodular functions without full access to a large data set.

Appendix I: Proofs

Proof of Proposition 2. Suppose F is clustering-consistent for clustering $\mathcal{C}_1, \dots, \mathcal{C}_L$. We prove Proposition 2 in two steps:

Let T_1 be such that at least one element $b \in \mathcal{C}_i$ has been encountered for each cluster $\mathcal{C}_i$. Then, for the solution $\mathcal{A}_{T_1}$ it holds that $|\mathcal{C}_i \cap \mathcal{A}_{T_1}| \leq 1$ for each i, i.e., $\mathcal{A}_{T_1}$ contains at most one representative of each cluster: Let t_i be the smallest index such that $b_{t_i} \in \mathcal{C}_i$ (i.e., the first iteration where a representative of cluster i appears in the stream). W.l.o.g., assume that $t_1 < t_2 < \dots < t_L \leq T_1$. For any set $\mathcal{A} \subseteq \mathcal{V}$, let

$$r(\mathcal{A}) = |\{i : \mathcal{C}_i \cap \mathcal{A} \neq \emptyset\}|$$

denote the number of clusters from which at least one representative has been selected. By definition of clustering-consistency, it can be seen that for the sequence of sets $\mathcal{A}_1, \dots, \mathcal{A}_T$ chosen by StreamGreedy it holds that $r(\mathcal{A}_1) \leq \dots \leq r(\mathcal{A}_T)$, i.e., it is never preferable to remove a single representative s of cluster $\mathcal{C}_i$ in order to have multiple representatives of some cluster $\mathcal{C}_j$. Moreover, it can be seen that

$$r(\mathcal{A}_{t_\ell}) = \min\{\ell, k\}.$$

Note that $T_1 \leq \rho$.

For the second step, note that for each $t \geq T_1$, it holds that $r(\mathcal{A}_t) = k$, i.e., k clusters will be represented, i.e., no set $\mathcal{A}_t$ will contain more than one exemplar from any cluster i. Let

$$s_i^* = \operatorname*{argmax}_{s \in \mathcal{C}_i} F(\{s\})$$

be the (w.l.o.g. unique) highest scoring representative of cluster i. Assume, w.l.o.g., that $F(\{s_1^*\}) \geq F(\{s_2^*\}) \geq \dots \geq F(\{s_L^*\})$. Due to the first condition of cluster-consistency (additive decomposition), it can be seen that

$$F(\{s_1^*, \dots, s_k^*\}) = \max_{|\mathcal{A}| \leq k} F(\mathcal{A}).$$

Let $t_i^* \geq T_1$ be the smallest integer such that $b_{t_i^*} = s_i^*$ where the element s_i^* appears in the stream. It can be seen that, for all $\ell \leq k$ and for all $t \geq t_\ell^*$ it holds that $s_\ell^* \in \mathcal{A}_t$, hence at time $T = t_k$ it must hold that $F(\mathcal{A}_T) = \max_{|\mathcal{A}| \leq k} F(\mathcal{A})$. Note that $T \leq 2\rho$.

□

Proof of Proposition 3. First consider the clustering objective $F = F_C$. Let $L_i(\mathcal{A}) = \sum_{s \in \mathcal{C}_i} \min_{c \in \mathcal{A} \cup \{\mathbf{x}_0\}} d(\mathbf{x}_s, \mathbf{x}_c)$ be the loss associated with cluster $\mathcal{C}_i$. Let $\mathcal{A} \subseteq \mathcal{C}_i$. Note that if $s \in \mathcal{C}_j$ for $j \neq i$, then $L_i(\mathcal{A} \cup \{s\}) = L_i(\mathcal{A})$, since $d(\mathbf{x}_{s'}, \mathbf{x}_0) \leq d(\mathbf{x}_{s'}, \mathbf{x}_s)$ for all $s' \in \mathcal{C}_i$. Hence, for any $\mathcal{A} \subseteq \mathcal{V}$, $F(\mathcal{A}) = \sum_{\ell=1}^{L} F(\mathcal{A} \cap \mathcal{C}_\ell)$. Now suppose $\mathcal{A} \subseteq \mathcal{C}_i$ and $s \in \mathcal{C}_i \setminus \mathcal{A}$. Then

$$F(\mathcal{A} \cup \{s\}) - F(\mathcal{A}) \leq |\mathcal{C}_i| \operatorname{diam}(\mathcal{C}_i).$$

On the other hand, if $s \in \mathcal{C}_j$, then

$$F(\{j\}) \geq |\mathcal{C}_j|(d(\mathbf{x}_0, \mathcal{C}_j) - \operatorname{diam}(\mathcal{C}_j)).$$

Hence, choosing

$$\alpha = \frac{\min_i d(\mathbf{x}_0, \mathcal{C}_i) - \max_j \operatorname{diam}(\mathcal{C}_j)}{\min_i \operatorname{diam}(\mathcal{C}_i)}$$

suffices to prove cluster-consistency of F_C. Choosing

$$\beta = \frac{\min_i d(\mathbf{x}_0, \mathcal{C}_i)}{\min_{i,j} d(\mathcal{C}_i, \mathcal{C}_j)}$$

suffices to ensure that $\alpha > 0$.

Now let us consider active set selection on GP regression. Under the squared exponential kernel with bandwidth h,

$$\mathcal{K}(s, s') = \eta^2 \exp(-d(s, s')^2/h^2),$$

for any $\varepsilon > 0$, there is a constant c, such that for two sets $\mathcal{A}, \mathcal{B}$ with $d(\mathcal{A}, \mathcal{B}) > ch$, it holds that $|H(\mathcal{X}_\mathcal{A}, \mathcal{X}_\mathcal{B}) - H(\mathcal{X}_\mathcal{A}) - H(\mathcal{X}_\mathcal{B})| < \varepsilon$, and similarly $|\sigma_s^2 - \sigma_{s|\mathcal{A}}^2| \leq \varepsilon$, whenever $s \in \mathcal{B}$. This proves the additive decomposition property (up to arbitrarily small error ε; Proposition 2 can be generalized to accommodate this arbitrarily small error). Let $L_i(\mathcal{A}) = \sum_{s \in \mathcal{C}_i} [\sigma_s^2 - \sigma_{s|\mathcal{A}}^2]$. Now, there exists a constant c' such that if $\operatorname{diam}(\mathcal{C}_i) < c'h$ then for any $s \in \mathcal{C}_i$ and $\gamma < 1$ it holds that $L_i(\{s\}) < \gamma |\mathcal{C}_i| \eta^2$, and thus

$$F(\{s\}) > (1 - \gamma)|\mathcal{C}_i|.$$

Similarly, for any $\mathcal{A} \subseteq \mathcal{C}_i$ and $s \in \mathcal{C}_i \setminus \mathcal{A}$,

$$F(\mathcal{A} \cup \{s\}) - F(\mathcal{A}) < \gamma |\mathcal{C}_i|,$$

proving cluster-consistency for appropriate choice of α. A similar reasoning can be used to prove cluster-consistency of the IVM objective F_H. □

Proof of Theorem 4. StreamGreedy can be interpreted as an instance of the pairwise exchange heuristic for submodular functions, which iteratively replaces a selected element by a non-selected element until no further improvement in score is possible, with the difference that the choice of candidate elements for replacement is determined by the data stream. The proof of Theorem 4 is thus analogous to the analysis of the pairwise exchange heuristic for submodular functions by Nemhauser et al. (1978), exploiting the key insight that the ordering in which pairwise exchanges are performed is immaterial for the performance guarantee of the pairwise exchange heuristic. The proof below also accommodates for the fact that F is only evaluated up to small additive error ε (by means of $\widehat{F}$), and improvement of at least η is required for each exchange.

Let T be index such that $\mathcal{A}_t = \mathcal{A}_T$ for all sets $\mathcal{A}_t$, $t \geq T$, chosen by StreamGreedy. Such a T must exist, since $F(\mathcal{A}_{t+1}) \geq F(\mathcal{A}_t)$ for all t, and $\mathcal{A}_{t+1} \neq \mathcal{A}_t$ only if $F(\mathcal{A}_{t+1}) \geq F(\mathcal{A}_t) + \eta$, and $F(\mathcal{V})$ is bounded. Construct an ordering $s_1, \ldots, s_k$ such that $s_i \in \text{argmax}_{s \in \mathcal{A}_T} F(\{s_1, \ldots, s_{i-1}, s\})$. Also let $\mathcal{A}^* = \{r_1, \ldots, r_k\}$ such that $F(\mathcal{A}^*) = \max_{|\mathcal{A}| \leq k} F(\mathcal{A})$. Let $\mathcal{S} = \{s_1, \ldots, s_{k-1}\}$, and $\delta_i = F(\{s_1, \ldots, s_i\}) - F(\{s_1, \ldots, s_{i-1}\})$. Note that $\delta_i \leq \delta_{i-1}$, due to submodularity and the fact that $s_1, \ldots, s_k$ are in greedy order. Now, due to submodularity and monotonicity of F it holds that

$$\begin{aligned}
F(\mathcal{A}^*) &\leq F(\mathcal{A}^* \cup \mathcal{S}) \\
&\leq F(\mathcal{S}) + \sum_{i=1}^{k} [F(\mathcal{S} \cup \{r_i\}) - F(\mathcal{S})] \\
&\leq F(\mathcal{S}) + k\,(\delta_k + \varepsilon + \eta) \qquad (8.1) \\
&\leq F(\mathcal{S}) + \sum_{i=1}^{k} \delta_i \qquad (8.2) \\
&= F(\mathcal{S}) + F(\mathcal{S} \cup \{s_k\}) \\
&\leq 2F(\mathcal{A}_T)
\end{aligned}$$

where inequality (8.1) follows from the fact that StreamGreedy did not replace s_k by any element r_i from the optimal solution $\mathcal{A}^*$. Note that after ρ iterations, $F(\mathcal{A}_t)$ must increase by at least η, or StreamGreedy will stop. Hence, $T \leq \rho F(\mathcal{V})/\eta \leq \rho B/\eta$. □

Appendix II: Implementation Details

Clustering

When determining the swap to perform at iteration $t > K$, we maintain the following quantities from iteration $t - 1$:

- The distance of each validation point $i \in \mathcal{W}$ to its cluster exemplar:

$$m_i = \min_{c \in A_{t-1} \cup \{\mathbf{x}_0\}} d(\mathbf{x}_i, \mathbf{x}_c) \qquad (8.3)$$

- The identity of each validation point's exemplar:

$$z_i = \arg \min_{c \in A_{t-1} \cup \{\mathbf{x}_0\}} d(\mathbf{x}_i, \mathbf{x}_c) \qquad (8.4)$$

- The distance of each validation point to its second nearest exemplar

$$o_i = \min_{c \in A_{t-1} \cup \{\mathbf{x}_0\} \setminus z_i} d(\mathbf{x}_i, \mathbf{x}_c) \qquad (8.5)$$

We then compute the dissimilarity of each candidate in $\mathcal{B}_t$ to the points in the validation set $\mathcal{W}$ which requires $O(|\mathcal{B}_t||\mathcal{W}|\text{cost}\{d\})$ operations (where $\text{cost}\{d\}$ is the cost associated with computing the dissimilarity $d(\mathbf{x}, \mathbf{x}'))$. We then score each of the $K|B_t|$ potential swaps (indexed by $s \in \mathcal{B}_t$ and $s' \in \mathcal{A}_{t-1}$) by computing

$$\begin{aligned}
&L(\mathcal{A}_{t-1} \setminus \{s'\} \cup \{s, \mathbf{x}_0\}) \\
&\qquad = L(\mathcal{A}_{t-1} \setminus \{s'\} \cup \{s, \mathbf{x}_0\}) - L(\mathcal{A}_{t-1} \cup \{s, \mathbf{x}_0\}) \\
&\qquad + L(\mathcal{A}_{t-1} \cup \{s, \mathbf{x}_0\}) - L(\mathcal{A}_{t-1} \cup \{\mathbf{x}_0\}) \\
&\qquad + L(\mathcal{A}_{t-1} \cup \{\mathbf{x}_0\}).
\end{aligned}$$

This can be done in $O(|\mathcal{W}|)$ operations since the decrease in loss due to adding center s

$$\begin{aligned}
L(\mathcal{A}_{t-1} \cup \{s, \mathbf{x}_0\}) &- L(\mathcal{A}_{t-1} \cup \{\mathbf{x}_0\}) \\
&= \sum_{i: d(\mathbf{x}_i, \mathbf{x}_s) < m_i} d(\mathbf{x}_i, \mathbf{x}_s) - m_i
\end{aligned}$$

and the increase in loss associated with removing center s'

$$\begin{aligned}
L(\mathcal{A}_{t-1} \setminus \{s'\} \cup \{s, \mathbf{x}_0\}) &- L(\mathcal{A}_{t-1} \cup \{s, \mathbf{x}_0\}) \\
&= \sum_{i: z_i = s' \wedge [d(\mathbf{x}_i, \mathbf{x}_{s'}) < d(\mathbf{x}_i, \mathbf{x}_s)]} o_i - m_i
\end{aligned}$$

require $O(|\mathcal{W}|)$. The third term $L(\mathcal{A}_{t-1} \cup \{\mathbf{x}_0\}) = \sum_{i \in \mathcal{W}} m_i$ doesn't depend on s or s'. The total cost for iteration t is $O(K|\mathcal{B}_t||\mathcal{W}| + |\mathcal{B}_t||\mathcal{W}|\text{cost}\{d\})$.

GP Regression

At each iteration t we retain from iteration $t - 1$ the Cholesky decomposition of $\Sigma_{\mathcal{A}_{t-1}, \mathcal{A}_{t-1}} = R_{t-1}^T R_{t-1}$, where R_{t-1} is an upper right triangular matrix, as well as the output of the kernel function $\mathcal{K}$ evaluated between points in $\mathcal{W}$ and $\mathcal{A}_{t-1}$. We compute $\mathcal{K}$ between each member of $\mathcal{B}_t$ and $\mathcal{W}$ as well as between $\mathcal{B}_t$ and $\mathcal{A}_{t-1}$ in $O((|\mathcal{W}| + K)|\mathcal{B}_t|\text{cost}(\mathcal{K}))$.

For each candidate swap indexed by $s \in \mathcal{B}_t$ and $s' \in \mathcal{A}_{t-1}$, we compute $R_{t-1}^{(s,s')}$ which is the Cholesky decomposition of $\Sigma_{\mathcal{A}_{t-1}\setminus s'\cup\{s\},\mathcal{A}_{t-1}\setminus s'\cup\{s\}}$ with a downdate of element s' and update of element s performed in $O(K^2)$ operations. The prediction weight vector $\Sigma^{-1}_{\mathcal{A}_{t-1}\setminus s'\cup\{s\},\mathcal{A}_{t-1}\setminus s'\cup\{s\}}\mathbf{x}_{\mathcal{A}_{t-1}\setminus s'\cup\{s\}}$ can be computed in $O(K^2)$ operations using two forward substitutions (matrix right division $R_{t-1}^{(s,s')}\setminus([R_{t-1}^{(s,s')}]^T\setminus\mathbf{x}_{\mathcal{A}_{t-1}\setminus s'\cup\{s\}})$ in Matlab). The candidate swaps are scored according to $\sum_{i\in\mathcal{W}}(\mathbf{x}_i - \mu_{i|\mathcal{A}_{t-1}\setminus s'\cup\{s\}})^2$ in $O(K|\mathcal{W}|)$.

The total cost for iteration t is $O((|\mathcal{W}| + K)|\mathcal{B}_t|\text{cost}(\mathcal{K}) + K^3|\mathcal{B}_t| + K^2|\mathcal{B}_t||\mathcal{W}|)$. We are exploring ways to reduce this cost that involve identifying and evaluating only a fraction of the $K|\mathcal{B}_t|$ possible swaps, while still maintaining convergence guarantees.

Adaptive Stopping Rule

We have implemented an adaptive stopping rule based on updating a sufficient statistic $\widehat{F}(\mathcal{A}_{t-1}\cup\{s\}\setminus\{s'\}) = \sum_{i\in\mathcal{W}} f(\mathcal{A}_{t-1}\cup\{s\}\setminus\{s'\},\mathbf{x}_i)$ for each swap candidate (indexed by $s \in \mathcal{B}_t$ and $s' \in \mathcal{A}_{t-1}$). Each time a data point $\mathbf{x}_i$ arrives from the steam, $f(\mathcal{A}_{t-1}\cup\{s\}\setminus\{s'\},\mathbf{x}_i)$ is added to its corresponding sufficient statistic. We define an algorithm failure probablity $\hat{\delta}$, and use Lemma 1 with $\delta = \frac{\hat{\delta}}{A}$ where A is the maximum number of times the bound will be used during the course of the algorithm. This establishes confidence bands $\varepsilon_{s,s'}$ around each statistic depending on the number of samples summarized so far by the sufficient statistics, as well as confidence band $\varepsilon_{\mathcal{A}_{t-1}}$ on $\widehat{F}(\mathcal{A}_{t-1})$ (the current utility). We halt when one of two conditions are met:

- There exists a swap (s, s') such that $\widehat{F}(\mathcal{A}_{t-1}\cup\{s\}\setminus\{s'\}) - \varepsilon_{s,s'} > \widehat{F}(\mathcal{A}_{t-1}) + \varepsilon_{\mathcal{A}_{t-1}}$. We then perform swap (s, s') and move on to the next iteration $t+1$.
- For all swaps (s, s'), $\widehat{F}(\mathcal{A}_{t-1}\cup\{s\}\setminus\{s'\}) + \varepsilon_{s,s'} < \widehat{F}(\mathcal{A}_{t-1}) - \varepsilon_{\mathcal{A}_{t-1}}$. No swap is performed and we move on to the next iteration $t+1$.

This setup is similar to Hoeffding Racing (Maron & Moore, 1994). We have experimented with this rule on the Tiny Images data set. We find that in the early iterations, it can cut down the number of validation samples used by a factor between 3 and 10. However, as the algorithm proceeds, the difference in utility between swap candidates becomes smaller and eventually the entire data set is used. We observe that this approach is pessimistic: $\arg\max_{s,s'} \widehat{F}(\mathcal{A}_{t-1}\cup\{s\}\setminus\{s'\})$ stablizes with many fewer samples than required by the stopping rule.

8.1. Acknowledgements

We thank Pietro Perona, Piotr Dollar, Kristin Branson, and the anonymous reviewers for their helpful comments. This research was partially supported by ONR grants N00014-09-1-1044 and N00014-06-1-0734, a gift from Microsoft Corporation, and an Okawa Foundation Research Grant.

References

Charikar, M., O'Callaghan, L., and Panigrahy, R. Better streaming algorithms for clustering problems. In *STOC*, pp. 30–39, 2003.

Cressie, N. A. C. *Statistics for Spatial Data*. Wiley, 1991.

Csató, Lehel and Opper, Manfred. Sparse on-line gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

Das, A. and Kempe, D. Algorithms for subset selection in linear regression. In *STOC*, 2008.

Dasgupta, S. Lecture notes on online clustering. Technical report, http://www-cse.ucsd.edu/∼dasgupta/291/lec6.pdf, 2009.

Domingos, P. and Hulten, G. Mining high-speed data streams. In *KDD*, 2000.

Domingos, P. and Hulten, G. A general method for scaling up machine learning algorithms and its application to clustering. In *ICML*, 2001.

Dueck, D. and Frey, B. J. Non-metric affinity propagation for unsupervised image categorization. In *ICCV*, pp. 1–8, 2007.

Feige, U. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634 – 652, July 1998.

Gaber, Mohamed Medhat, Zaslavsky, Arkady, and Krishnaswamy, Shonali. Mining data streams: a review. *SIGMOD Record*, 34(2):18–26, June 2005.

Guha, Meyerson, Mishra, Motwani, and O'Callaghan. Clustering data streams: Theory and practice. *IEEE TKDE*, 15, 2003.

Kaufman, L. and Rousseeuw, P. J. *Finding Groups in Data: an Introduction to Cluster Analysis*. Wiley, 1990.

Maron, Oded and Moore, Andrew W. Hoeffding races: Accelerating model selection search for classification and function approximation. In *NIPS*, 1994.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. An analysis of approximations for maximizing submodular set functions. *Math. Programming*, 14(1):265–294, December 1978.

Ng, R. T. and Han, J. CLARANS: A method for clustering objects for spatial data mining. *IEEE Trans. Knowl. Data Eng*, 14(5):1003–1016, 2002.

Rasmussen, C. E. and Williams, C. K.I. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, 2006.

Seeger, M. Greedy forward selection in the informative vector machine. Technical report, University of California at Berkeley, 2004.

Seeger, M., Williams, C. K. I., and Lawrence, N. D. Fast forward selection to speed up sparse gaussian process regression. In *AISTATS*, 2003.

Smola, Alex J. and Bartlett, Peter L. Sparse greedy gaussian process regression. In *NIPS*, pp. 619–625. MIT Press, 2000.

Smola, Alex J., Mangasarian, Olvi L., and Scholkopf, Bernhard. Sparse kernel feature analysis, 1999.

Streeter, Matthew and Golovin, Daniel. An online algorithm for maximizing submodular functions. In *NIPS*, pp. 1577–1584, 2008.

Torralba, A., Fergus, R., and Freeman, W. T. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE PAMI*, 30(11):1958–1970, November 2008.

Weston, Jason, Bordes, Antoine, and Bottou, Léon. Online (and offline) on an even tighter budget. In *AISTATS*, pp. 413–420, 2005.