Near-optimal Batch Mode Active Learning and Adaptive Submodular Optimization

Yuxin Chen YUXIN.CHEN@INF.ETHZ.CH
Andreas Krause KRAUSEA@ETHZ.CH
ETH Zurich, Universitätstrasse 6, 8092 Zürich, Switzerland

Abstract

Active learning can lead to a dramatic reduction in labeling effort. However, in many practical implementations (such as crowdsourcing, surveys, high-throughput experimental design), it is preferable to query labels for batches of examples to be labelled in parallel. While several heuristics have been proposed for batch-mode active learning, little is known about their theoretical performance.

We consider batch mode active learning and more general information-parallel stochastic optimization problems that exhibit *adaptive submodularity*, a natural diminishing returns condition. We prove that for such problems, a simple greedy strategy is competitive with the optimal batch-mode policy. In some cases, surprisingly, the use of batches incurs competitively low cost, even when compared to a fully sequential strategy. We demonstrate the effectiveness of our approach on batch-mode active learning tasks, where it outperforms the state of the art, as well as the novel problem of multi-stage influence maximization in social networks.

1. Introduction

Active learning, i.e., sequential selection of unlabeled examples for labeling, can lead to dramatic (potentially exponential) reduction in labeling effort as compared to passive learning. In many practical settings, however, fully sequential selection, where the choice of the next example depends on all previous labels, is infeasible. For example, when recruiting workers on Amazon Mechanical Turk for crowdsourcing annotation, one usually generates tasks comprising several unlabeled examples. Similarly, in high-throughput experimental design, it is often more cost-effective to perform several experiments in parallel. Such problems have been studied from the perspective of *batch-mode* active learning. While several heuristics have been proposed, little is known about their theoretical performance. More generally, in many sequential decision problems, we would like to choose multiple actions to be performed in parallel, and receive feedback only after all actions have been carried out. This feedback then informs the next batch of actions. For example, consider a viral marketing problem (Kempe et al., 2003), where we wish to spur demand for a new product by influencing a set of nodes in a social network. In such a setting it is natural to conduct a multi-stage marketing campaign, where each stage is informed by the observed effectiveness of the previous stage. Similar problems arise in resource allocation in computational sustainability (Golovin et al., 2011), and vaccination problems in epidemiology (Anshelevich et al., 2009).

In this paper, we study *information-parallel learning and decision making*. In particular, we tackle batch-mode active learning and more general stochastic optimization problems, such as influence maximization in social networks, that exhibit *adaptive submodularity* (Golovin & Krause, 2011), a natural diminishing returns condition. We prove that, for such problems, a simple BatchGreedy approach, which greedily selects examples within a batch, and assembles batches in a greedy manner, is competitive with the optimal batch-mode policy. Furthermore, we prove that surprisingly, in some natural settings, the *price of parallelism* is bounded: the use of batches incurs competitively low cost *irrespective* of the batch size, even when compared to a *fully sequential* policy. We demonstrate the effectiveness of our approach on active learning tasks, as well as adaptive influence maximization in social networks. Our approach is the first to provide both strong guarantees and compelling

Proceedings of the 30^{th} International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

empirical performance for the important practical problem of batch mode active learning, where BATCHGREEDY improves on random selection by $\approx 48\%$ more than the state of the art does on our test sets.

In summary, our main contributions are:

- We consider a general approach for information parallel learning and decision making,
- prove strong performance guarantees for a simple BATCHGREEDY algorithm,
- provide practical algorithms for batch-mode active learning and influence maximization, and
- demonstrate the empirical effectiveness of the algorithms for both applications.

2. Problem Statement and Applications

We first describe two different applications that motivate our research. Then, in Section 2.2 we introduce a formalism that captures both of them, and then prove results for this more general model in Section 3.

2.1. Motivating applications

Pool-based batch mode active learning Consider a simple model of *pool-based Bayesian active learning*. We are given a pool $\mathcal{V}$ of unlabeled examples $\mathbf{x}_1, \ldots, \mathbf{x}_n$. We use $y_1, \ldots, y_n \in \{+1, -1\}$, where y_i is the (initially unknown) label[1] of example $\mathbf{x}_i$. Our goal is to learn a classifier $h : \mathcal{V} \to \{+1, -1\}$ out of a finite set $\mathcal{H}$ of hypotheses, each corresponding to distinct labelings of the pool $\mathcal{V}$, and containing the true labeling, i.e., a hypothesis h such that $h(\mathbf{x}_i) = y_i$ for $1 \leq i \leq n$. For now let us assume that we have a uniform prior $P(h) = \frac{1}{|\mathcal{H}|}$ over the hypotheses. We later show that our results also hold for more general priors, as well as for the prior-free (non-Bayesian) setting.

Suppose we have already observed the labels $\mathbf{y}_\mathcal{A}$ for a subset $\mathcal{A} \subseteq \mathcal{V}$ of the pool. In this case, some of the hypotheses $h \in \mathcal{H}$ will be inconsistent with the observations $\mathbf{y}_\mathcal{A}$, and we use the notation $\mathcal{H}(\mathbf{y}_\mathcal{A}) = \{h \in \mathcal{H} : i \in \mathcal{A} \Rightarrow y_i = h(\mathbf{x}_i)\}$ to refer to the version space (set of hypotheses) consistent with the observation $\mathbf{y}_\mathcal{A}$. We wish to actively select a minimum number of unlabeled examples and obtain their labels $\mathbf{y}_\mathcal{A}$, such that these allow us to uniquely identify h (i.e., infer the labels of all unlabeled examples), so that $|\mathcal{H}(\mathbf{y}_\mathcal{A})| = 1$. An optimal active learning strategy is one that minimizes the expected number of labels requested, in expectation over our prior $P(h)$. Similarly, we can consider strategies for *batch-mode* active learning, which pick *batches* of k unlabeled examples at a time, then request all labels for the selected batch in parallel, and then proceed to pick the next batch given the labels obtained so far. See Figure 1 for an illustration.

Finding such an optimal policy is a formidable task. In fact, even *representing* an optimal batch policy may require exponential space. In the following, we will describe a general class of batch mode optimization problems, and present a simple greedy algorithm that is provably competitive with the optimal batch policy.

Multi-stage influence maximization in social networks Suppose we would like to stimulate demand for a novel product. The idea behind viral marketing is to utilize the social network structure connecting the potential customers: By giving the product to a subset of target people for free, these may influence their friends, potentially creating a cascade of influence motivating many more consumers to adopt the product.

This problem was formalized by Kempe et al. (2003), who show that many natural models of influence (such as the independent cascade, or linear threshold models) can be modeled stochastically. Formally, let $\mathcal{V} = \{1, \ldots, n\}$ be the set of nodes in the social network, and let $Y_s \subseteq \mathcal{V}$ be the (random) set of nodes eventually influenced if s is initially targeted. If a set $\mathcal{A}$ of nodes is initially targeted, the eventual influence is $\left|\bigcup_{s\in\mathcal{A}} Y_s\right|$ with probability[2] $P(\mathbf{Y}_\mathcal{A}) = \prod_{s\in\mathcal{A}} P(Y_s)$.

Instead of committing to all target nodes in advance, it is natural to consider conducting a multi-stage advertising campaign: In each stage certain nodes are targeted, then the effect of the campaign is observed, then the next target nodes are chosen, and so on. Implementing such a procedure may be much more practical if many nodes can be selected in each stage, to be influenced in parallel. In the following, we propose a simple greedy approach that is competitive not only with the optimal multi-stage strategy but even with an optimal *fully sequential* strategy.

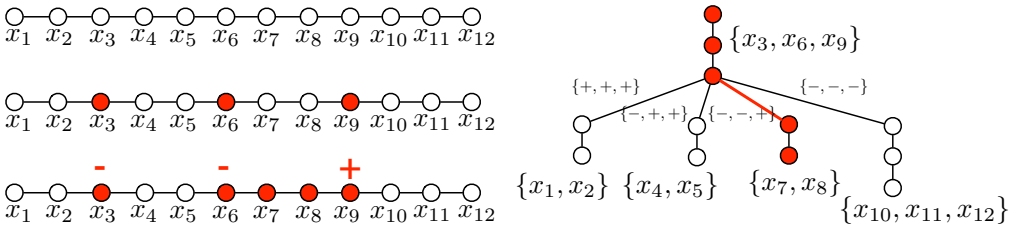
2.2. General Problem Statement

We now formalize a class of interactive optimization problems generalizing the two examples of Section 2.1.

Adaptive Submodular Optimization We wish to adaptively select items $\mathcal{A}$ out of a finite set of n items $\mathcal{V} = \{1, \ldots, n\}$ (unlabeled examples; target nodes). Each item $s \in \mathcal{V}$ is associated with a random variable Y_s, taking values in a (finite) set O of outcomes (labels; sets of nodes eventually influenced). We use $\mathbf{Y}_\mathcal{V} = [Y_1, \ldots, Y_n]$, to refer to the collection of all

[1] Note that our approach naturally extends to ≥ 2 labels.

[2] Here we focus on factorial priors. Dependencies can be modeled as well (Golovin & Krause, 2011).



(a) *1-dimensional data* (b) *The policy tree*

Figure 1. Illustration of batch mode ($k = 3$) active learning, in the simple case of one-dimensional data and binary threshold hypotheses. Figure 1(a) shows the unlabeled data (top row), first batch selected for labeling (middle row), and received labels, as well as second selected batch (bottom row). Figure 1(b) illustrates the decision tree representing the BatchGreedy policy, with the branch taken for the example of Figure 1(a).

variables, and assume that $\mathbf{Y}_\mathcal{V}$ is distributed according to a joint distribution $P(\mathbf{Y}_\mathcal{V})$. Whenever an item s is selected, the corresponding variable $Y_s = y_s$ is revealed. This information can be used to select subsequent items. We model the value associated with a set of items $\mathcal{A}$, and corresponding observations $\mathbf{y}_\mathcal{A} \subseteq \mathcal{V} \times O$ by means of an objective function[3] $f : 2^{\mathcal{V}\times O} \to \mathbb{N}$. In our active learning example, we can use $f(\mathbf{y}_\mathcal{A}) = |\mathcal{H}| - |\mathcal{H}(\mathbf{y}_\mathcal{A})|$, i.e., the number of the hypotheses eliminated through the labeled examples $\mathbf{y}_\mathcal{A}$. In our viral marketing example, we choose $f(\mathbf{y}_\mathcal{A}) = |\bigcup_{s\in\mathcal{A}} y_s|$, i.e., the number of nodes eventually influenced. Furthermore, let $\mathcal{S} \subseteq \mathcal{V} \times O$ be a set of observations. Note that while technically $\mathbf{y}_\mathcal{A}$ and $\mathcal{S}$ do not denote the same objects ($\mathcal{S}$ denotes a set of item/observation pairs, $\mathbf{y}_\mathcal{A}$ denotes the observations corresponding the item set $\mathcal{A}$), we will sometimes use these notations interchangeably to refer to observations. In both applications, f satisfies the following natural four properties[4]:

1. *Normalized:* $f(\emptyset) = 0$, i.e., we derive no utility from knowing nothing.
2. *Monotonic:* Whenever $\mathcal{S} \subseteq \mathcal{S}' \subseteq \mathcal{V} \times O$, then $f(\mathcal{S}) \leq f(\mathcal{S}')$ – adding labels never hurts.
3. *Submodular*: whenever $\mathcal{S} \subseteq \mathcal{S}' \subseteq \mathcal{V} \times O$ and $(j, y) \in \mathcal{V} \times O \setminus \mathcal{S}'$, it holds that $f(\mathcal{S} \cup \{(j, y)\}) - f(\mathcal{S}) \geq f(\mathcal{S}' \cup \{(j, y)\}) - f(\mathcal{S}')$. Thus, adding a label helps more if we have observed few labels so far, and less if we have obtained many labels.
4. *Adaptive submodular:* Consider the *conditional expected marginal gain* of item j w.r.t. to observations $\mathcal{S} \subseteq \mathcal{V} \times O$,

$$\Delta_f(j \,|\, \mathcal{S}) = \sum_y P(Y_j = y \,|\, \mathcal{S}) \Big[f(\mathcal{S} \cup \{(j, y)\}) - f(\mathcal{S}) \Big]. \tag{2.1}$$

Function f together with distribution $P(\mathbf{Y}_\mathcal{V})$ is called adaptive submodular, if, whenever $\mathcal{S} \subseteq \mathcal{S}' \subseteq \mathcal{V} \times O$ and $P(\mathcal{S}') > 0$ we have $\Delta_f(j \mid \mathcal{S}) \geq \Delta_f(j \mid \mathcal{S}')$. Thus, the gain of item j, in expectation over its unknown label, can never increase as we gather more information.

Our goal will be to find a policy π for selecting items (and associated observations) $\mathbf{y}_\mathcal{A}$, such that we achieve a certain quota of value $Q \geq 0$, i.e., $f(\mathbf{y}_\mathcal{A}) \geq Q$, while at the same time minimizing the number of items $\mathcal{A}$ used. In the active learning example, $Q = |\mathcal{H}| - 1$: achieving this quota is a necessary and sufficient condition for identifying the true hypothesis. In influence maximization, Q may be a certain fraction of the size of the social network. In the following, w.l.o.g.[5], we assume $f(\mathbf{y}_\mathcal{V}) = Q$ for all $\mathbf{y}_\mathcal{V} \in \text{supp}(P)$.

Formally, a policy $\pi : 2^{\mathcal{V}\times O} \to \mathcal{V}$ is a partial mapping from observations $\mathbf{y}_\mathcal{A} \subseteq \mathcal{V} \times O$ to the next item to be picked (or to stop, if $\mathbf{y}_\mathcal{A} \notin \text{dom}(\pi)$). Therefore, if the variables $\mathbf{Y}_\mathcal{V}$ are in state $\mathbf{Y}_\mathcal{V} = \mathbf{y}_\mathcal{V}$, the policy obtains a set of observations, which is denoted as $\mathcal{S}(\pi, \mathbf{y}_\mathcal{V}) \subseteq \mathcal{V} \times O$. We define the expected and worst-case cost of policy π as

$$\text{cost}_{ac}(\pi) = \mathbb{E}_{\mathbf{y}_\mathcal{V}}[|\mathcal{S}(\pi, \mathbf{y}_\mathcal{V})|]\,;\ \text{cost}_{wc}(\pi) = \max_{\mathbf{y}_\mathcal{V}} |\mathcal{S}(\pi, \mathbf{y}_\mathcal{V})|$$

Our goal, is to find, out of a set Π of candidate policies a feasible policy π^* with minimum cost,

$$\min_{\pi\in\Pi} \text{cost}(\pi) \text{ s.t. } f(\mathcal{S}(\pi, \mathbf{y}_\mathcal{V})) \geq Q \text{ for all } \mathbf{y}_\mathcal{V} \text{ w. } P(\mathbf{y}_\mathcal{V}) > 0.$$

Batch selection Based on this notation, we can study how different classes of policies compare in terms of their cost. On the one extreme, we have *fully sequential* policies Π_{seq}, where the choice of each item may depend on the labels of *all* previous items selected. On the other extreme, we have *constant*, or

[3] Note that if f takes rational values, we can normalize it to take integer values.

[4] As a discrete analog of convexity, in many ways (adaptive) submodularity is a minimal assumption needed to ensure (approximate) tractability.

[5] If $f(\mathbf{y}_\mathcal{V}) > Q$, we replace f with the – still submodular – function $f_Q(\mathbf{y}_\mathcal{A}) = \min(f(\mathbf{y}_\mathcal{A}), Q)$.

non-adaptive policies Π_{const} which commit to items picked in advance, before making any observations. However, fully sequential and constant policies are only two extremes on a spectrum. We are interested in policies $\Pi_{[k]}$ that sequentially pick *batches* of size k. Any policy $\pi \in \Pi_{[k]}$ starts selecting a fixed set $\mathcal{A}_1 \subseteq \mathcal{V}$ of k items. It then obtains all labels $\mathbf{y}_{\mathcal{A}_1}$. If $f(\mathbf{y}_{\mathcal{A}_1}) \geq Q$, it stops. Otherwise, if batches $\mathcal{A}_1, \ldots, \mathcal{A}_{\ell-1}$ have already been selected, it picks batch $\mathcal{A}_\ell \subseteq \mathcal{V}$ of k items, obtains the labels, and stops if $f(\mathbf{y}_{\mathcal{A}_1 \cup \cdots \cup \mathcal{A}_\ell}) \geq Q$.

Obtaining an optimal batch policy is a formidable task: There are $\binom{n}{k}$ batches of size k, and an optimal batch policy assembles such batches into a decision tree of possibly exponentially large branching factor. In the following, we describe a simple greedy algorithm, and prove that it implements a batch policy with cost competitive to that of the optimal batch policy. Moreover, we prove that under some additional conditions on the distribution $P(\mathbf{Y}_\mathcal{V})$, the greedy algorithm is even competitive with the optimal *fully sequential* policy.

3. Near-optimal Batch Selection: Greedy Algorithms and Guarantees

We consider a simple, greedy approach towards constructing batch policies. This policy, BatchGreedy, selects items *within* a batch in a greedy manner, then receives observations for all items in the batch, then selects the next batch in a greedy manner, conditional on all observations made so far, and so on. An important challenge in batch selection is the fact that the value of items (e.g., unlabeled examples) selected depends on observations (e.g., labels) obtained only *after* the entire batch is selected. In active learning for example, one wishes to select examples within a batch that are likely to be informative individually, but also diverse (minimize redundancy). BatchGreedy addresses this challenge by using a suitable notion of marginal benefit of an item, that takes into account all observations made so far, as well as items that have already been selected within the batch (but no observation has been obtained yet). Formally, we generalize the *conditional marginal benefit* (2.1) of item s by

$$\Delta_f(s \mid \mathcal{A}, \mathbf{y}_\mathcal{B}) = \mathbb{E}_{\mathbf{y}_\mathcal{V}}\left[f(\mathbf{y}_{\{s\}\cup\mathcal{A}\cup\mathcal{B}}) - f(\mathbf{y}_{\mathcal{A}\cup\mathcal{B}}) \mid \mathbf{y}_\mathcal{B}\right]. \tag{3.1}$$

Thus, $\Delta_f(s \mid \mathcal{A}, \mathbf{y}_\mathcal{B})$ reflects the expected marginal gain of item s, when items $\mathcal{B}$ have been selected and the corresponding observations $\mathbf{y}_\mathcal{B}$ have been made, and items $\mathcal{A}$ have already been selected, but *no* observations have yet been made about them. Therefore, (3.1) captures possible redundancy (diminishing gains) of candidate item s w.r.t. to labels already obtained, as well as labels that will likely be obtained within the batch. Hence it encourages diversity among the items selected in the batch.

Algorithm 1 The BatchGreedy algorithm.

Input: Quota Q. Objective f and prior $P(\mathbf{y}_\mathcal{V})$
$\mathbf{y}_\mathcal{B} \leftarrow \emptyset$
repeat
  $\mathcal{A} \leftarrow \emptyset$
  for $i = 1$ **to** k **do**
    $s \leftarrow \arg\max_{s'} \Delta_f(s' \mid \mathcal{A}, \mathbf{y}_\mathcal{B})$; $\mathcal{A} \leftarrow \mathcal{A} \cup \{s\}$
  end for
  Observe $\mathbf{y}_\mathcal{A}$ and set $\mathbf{y}_\mathcal{B} \leftarrow \mathbf{y}_\mathcal{B} \cup \mathbf{y}_\mathcal{A}$
until $f(\mathbf{y}_\mathcal{B}) \geq Q$

Using this notation, the BatchGreedy policy will greedily select the i-th element in the j-th batch

$$s_{i,j} = \arg\max_{s\in\mathcal{V}} \Delta_f(s \mid \{s_{1,j}, \ldots, s_{i-1,j}\}, \mathbf{y}_\mathcal{B}),$$

where $\mathbf{y}_\mathcal{B}$ is the set of observations (labeled examples) from batches up to $j-1$. After a batch is completed, all labels are requested and added to the observations $\mathbf{y}_\mathcal{B}$. Pseudocode is presented in Algorithm 1.

If we set the batch size k to 1, BatchGreedy reverts back to a fully sequential, greedy active learning scheme. In particular, for the active learning example from Section 2.1, this algorithm is known as *Generalized Binary Search*, studied extensively in the literature (see e.g., Dasgupta (2004)). In fact, it is known that this simple greedy algorithm is near-optimal: its cost is upper-bounded by $O(\log|\mathcal{H}|)$ times that of the optimal sequential policy. More generally, Golovin & Krause (2011) prove that this result can be generalized to any adaptive optimization problems that are *adaptive submodular*. As our first main theoretical contribution, we generalize their results, which only hold for fully sequential policies, to the batch setting.

We first show that BatchGreedy is near-optimal as compared to the optimal batch selection policy.

Theorem 1. *Let* $\mathsf{OPT}_{ac,k}$ *be the expected cost and* $\mathsf{OPT}_{wc,k}$ *be the worst-case cost of an optimal policy selecting batches of size* k*. Further let* $\delta = \min_{\mathbf{y}_\mathcal{V} \in supp(P)} P(\mathbf{y}_\mathcal{V})$*. Then for the cost of the policy* π_G *implementing* BatchGreedy *it holds that*

$$\mathrm{cost}_{ac}(\pi_G) \leq \mathsf{OPT}_{ac,k}\Big(\frac{e}{e-1}\Big)\Big(\ln Q + 1\Big), \text{ and}$$

$$\mathrm{cost}_{wc}(\pi_G) \leq \mathsf{OPT}_{wc,k}\Big(\frac{e}{e-1}\Big)\Big(\ln \frac{Q}{\delta} + 1\Big).$$

Note that the guarantee of Theorem 1 matches (up to a small constant factor) hardness results known for the fully sequential ($k = 1$) setting, which it generalizes, therefore BatchGreedy is near-optimal under

computational constraints. Further note that for the active learning application, Theorem 1 guarantees that in the non-Bayesian setting (i.e., without any prior[6]), BatchGreedy requires at most a factor of $\mathcal{O}(\ln|\mathcal{H}|)$ more batches than the optimal batch-mode policy.

As our second main theoretical result, we also prove that, perhaps surprisingly, under certain conditions BatchGreedy is not just competitive with the optimal policy that is restricted to selecting batches of examples: It is competitive with respect to an optimal fully sequential policy, which is *not* required to obey such a restriction. The additional condition needed is that the variables $Y_1,\ldots,Y_n$ are independent. This assumption is satisfied in the influence maximization application, but not in the active learning problem.

Theorem 2. *Fix $\beta>0$. Let OPT_{wc} be the worst-case cost of an optimal* sequential *policy π^*, constrained to picking a number of items which is a multiple of k. Further suppose that the variables $Y_1,\ldots,Y_n$ are independent. Then for the cost of the policy π_G implementing* BatchGreedy*, run until it achieves $f(\pi_G)\geq Q-\beta$ it holds that*

$$\mathrm{cost}_{wc}(\pi_G)\leq OPT_{wc}\ \Big(e/(e-1)\Big)^2\Big(\ln\frac{Q}{\beta}+1\Big).$$

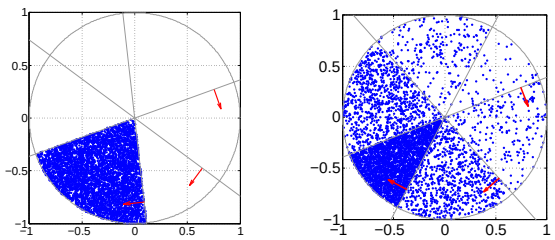
Moreover, it holds that $P\big(f(\mathcal{S}(\pi_G,\mathbf{y}_\mathcal{V}))\geq Q\big)\geq 1-\beta$.

The proofs are given in the supplementary material. The key technical insight behind the proof is that a bound on the adaptivity gap for stochastic submodular maximization of Asadpour et al. (2008), adapted and generalized to our setting, allows us to interpret the BatchGreedy policy as an approximate implementation of the fully sequential greedy policy. Note that Theorem 2, for technical reasons, is of a slightly different flavor than Theorem 1: it compares the optimal policy π^* always achieving quota Q with one achieving the quota Q only with probability $1-\beta$. By choosing $\beta<\min_{\mathbf{y}_\mathcal{V}\in\mathrm{supp}(P)}P(\mathbf{y}_\mathcal{V})$, it can be guaranteed that in fact $f(\mathcal{S}(\pi_G,\mathbf{y}_\mathcal{V}))=Q$ for all $\mathbf{y}_\mathcal{V}$ with nonzero probability. In this case, the bound on the worst-case cost of Theorem 2 is only a factor of $e/(e-1)\approx 1.58$ larger than that of Theorem 1, irrespective of the batch size.

4. Implementation Details

As is, BatchGreedy is not immediately practical for the applications from Section 2.1: Computing the marginal gains (3.1) requires computing expectations that may be intractable. In the influence maximization application, it is possible to perform Monte-Carlo sampling of the influence process to evaluate (3.1) up to arbitrarily small multiplicative error $(1+\varepsilon)$ (Kempe et al., 2003). Furthermore, with a slight generalization of the arguments of Golovin & Krause (2011), using such an approximation of (3.1) increases the cost by at most the same factor $(1+\varepsilon)$.

To obtain a practical algorithm for batch mode active learning, further challenges arise: BatchGreedy as is requires that $\mathcal{H}$ is finite, and its running time depends polynomially on $|\mathcal{H}|$. Furthermore, it requires that observations are noise free. As a practical implementation, we focus on active learning of linear separators, i.e., $h(\mathbf{x})=\mathrm{sign}(\mathbf{w}^T\mathbf{x})$. In our experiments we use a Markov-Chain Monte Carlo sampler in order to generate samples from the posterior distribution over hypotheses $P(\mathbf{w}\mid\mathbf{y}_\mathcal{S})$. In particular, we build on the hit-and-run sampler (Smith, 1984; Lovasz, 1998), which is known to lead to a provably efficient near-optimal estimation for the fully sequential active learning problem (Gonen et al., 2011). To handle noise, at each iteration, we generalize the hit-and-run sampler by sampling the



(a) $\varepsilon=0$ (b) $\varepsilon=0.2$

Figure 2. Illustration of Algorithm 2 in 2-d space. Figure 2(a) shows the sampling result in noise-free case (red arrows are constraints); Figure 2(b) shows sampling result when 20% of the observations are noisy: hypotheses that violate more constraints induce lower confidence.

Algorithm 2 Hit-and-run hypothesis sampler for linear separators and noisy observations

Input: Labeled examples $\mathbf{y}_\mathcal{B}\in\mathbb{R}^d$, N, number of mixing iterations T, noise level $\varepsilon\in[0,0.5)$.
Output: Hypotheses set $\hat{\mathcal{H}}$.
$\mathbf{w}_0\leftarrow$ a random point on d-dim. unit sphere S^d
for $i=1$ **to** N **do**
  $\theta\leftarrow$ a random direction (unit vector) in S^d
  Set $\mathcal{L}\leftarrow S^d\cap\{\mathbf{w}\mid\mathbf{w}=\mathbf{w}_{i-1}+\theta\rho,\rho\in\mathbb{R}\}$, and select $\mathbf{w}_i$ from sector $\mathcal{L}$ with $\rho\sim p(\rho)\propto(\frac{\varepsilon}{1-\varepsilon})^m$, where $m=|\{\mathbf{x}:\mathbf{x}\in\mathcal{B},(\mathbf{x},\mathrm{sign}(\mathbf{w}_i^T\mathbf{x}))\notin\mathbf{y}_\mathcal{B}\}|$
  Add every T-th sample $h_i(\mathbf{x})=\mathrm{sign}(\mathbf{w}_i^T\mathbf{x})$ to $\hat{\mathcal{H}}$
end for

[6] In the special case of uniform prior, $\delta=\frac{1}{|\mathcal{H}|}$. For general priors (with small δ), BatchGreedy can be proved to yield an $O(\log|H|)$ approximation, following analogously from Theorem 9.1 in Golovin & Krause (2011)

Algorithm 3 Approximate implementation of BATCHGREEDY for batch-mode active learning.

Input: Hypotheses $\mathcal{H}$, batch size k, noise level ε
Sample $\hat{\mathcal{H}} = \{h_1, \ldots, h_N\}$ from $\mathcal{H}$ using Algo. 2
Define $\hat{P}(\mathcal{H}) = \frac{1}{N}\sum_{\ell=1}^{N} \delta_{h_\ell}$; set $\mathbf{y}_\mathcal{B} \leftarrow \emptyset$
repeat
  $\mathcal{A} \leftarrow \emptyset$
  for $i = 1$ **to** k **do**
    $s \leftarrow \arg\min_{s'} \sum_{\ell=1}^{N} \left[\hat{P}(\mathcal{H}(\{(\mathbf{x}, h_\ell(\mathbf{x})) : \mathbf{x} \in \mathcal{A} \cup \{s'\}\}))\right]$
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{s\}$
  end for
  Observe $\mathbf{y}_\mathcal{A}$ and set $\mathbf{y}_\mathcal{B} \leftarrow \mathbf{y}_\mathcal{B} \cup \mathbf{y}_\mathcal{A}$
  Sample $\hat{\mathcal{H}} = \{h_1, \ldots, h_N\}$ (Algo. 2) using $\mathbf{y}_\mathcal{B}$
  Update approximation $\hat{P}(\mathcal{H}) = \frac{1}{N}\sum_{\ell=1}^{N} \delta_{h_\ell}$.
until $(1-\varepsilon)$ of all hypotheses in the support of $\hat{P}$ induce same labeling on the unlabeled pool

entire version space, while varying the sample density according to a likelihood function. In the case of symmetric binary channel (i.e., labels are flipped with probability ε), we sample hypotheses with probability related to the number of mistakes. This way, our method can handle data that are not linearly separable. Algorithm 2 presents details of our sampler, and our final batch-mode active learning algorithm is formalized in Algorithm 3. The time complexity of random sampling (Algorithm 2) is $\mathcal{O}(TN)$, where T is the number of mixing iterations. Once we discretized the hypothesis space with N samples, it takes $\mathcal{O}(knN)$ steps for Algorithm 3 to select a batch of k items. Hence the time complexity of Algorithm 3 selecting one batch is $\mathcal{O}(N(T + kn))$.

Furthermore, in both applications, we can use *lazy evaluations* to speed up the BATCHGREEDY algorithm (as used in Golovin & Krause (2011) for the fully sequential setting). Lazy evaluations utilize the fact that the marginal gains $\Delta_f(s \mid \mathcal{A}, \mathbf{y}_\mathcal{B})$ are monotonically decreasing in both $\mathcal{A}$ and $\mathbf{y}_\mathcal{B}$. This insight can be exploited by utilizing priority queues to accelerate selection of the next greedy choice.

5. Experimental Results

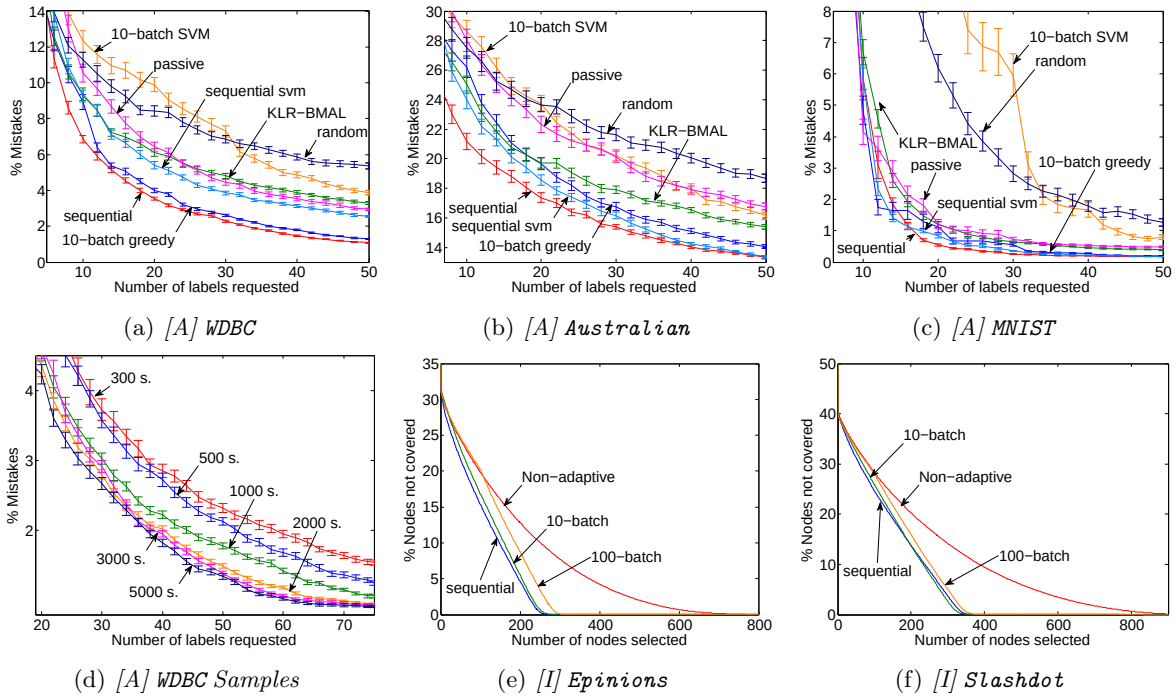
We empirically evaluate BATCHGREEDY on several data sets and on both applications discussed in Section 2.1. Our emphasis is on comparing BATCHGREEDY with baselines, as well as empirically quantifying the price of parallelism.

Batch mode active learning of linear separators One natural way to perform batch mode active learning is to select batches comprising the k most uncertain examples. As one baseline, we use "batch mode margin-based active learning" (k-batch SVM) to greedily select batches of examples, as considered in Jain et al. (2010) for large-scale active learning. In this method, we randomly chose examples until there are two distinct labels, and we train SVM classifiers based on the labeled examples at the end of each batch. The next k unlabeled examples with the smallest distances from the decision boundary $\mathbf{w}^T\mathbf{x} + b = 0$ are selected for labeling. Another baseline approach we employ is the state of the art batch mode active learning algorithm (KLR-BMAL) of Hoi et al. (2006) that selects batches of k examples that are informative w.r.t. the Fisher information matrix. To see how well the parallelization of the selection process approximates the sequential algorithm, we compare with the fully sequential active learning algorithm, where only one example is selected and observed at each iteration, as well as a "passive learning" approach, where we make no observations during the learning process (corresponding to infinite batch size). We also compare BATCHGREEDY against the sequential active learning algorithm with purely random selection.

For fair comparison, we use SVM as classifier for all competing algorithms, so the methods only differ by the set of examples chosen for labeling. We implement the KLR-BMAL algorithm using class membership probabilities inferred from the hypothesis sampler, and set the smoothing parameter δ to be 0.1 (Hoi et al., 2006). As for our sampler, we set ε to be 0.1. We normalize the data so that each feature has mean 0 and standard deviation 0.5, and place independent normal priors on each dimension. The results for all the batch mode active learning experiments are obtained from 150 random starts.

We run our the first set of experiments on two UCI data sets[7], `WDBC` (569 instances, 32-d) and `Australian` (690 instances, 16-d), using a fixed number of 5000 sampled hypotheses in each random trial. Figure 3(a) and Figure 3(b) depict the 150-trial average percentage of mistakes made by each algorithm when predicting the labels of the corresponding data set, for a batch size of $k = 10$. Figure 3(b) shows an improvement of BATCHGREEDY over both KLR-BMAL and the 10-batch SVM algorithm. On both datasets, surprisingly, BATCHGREEDY is competitive with the fully sequential greedy algorithm, with only minor differences. We also evaluate BATCHGREEDY on the `MNIST` data set. For each of the 14780 instances, we reduced the dimensionality down to 10 via PCA, and compare BATCHGREEDY with the sequential, KLR-BMAL, 10-bacth SVM, passive and random algorithms through 150 random trials. We observe that, even using 5000

[7] `http://archive.ics.uci.edu/ml/`



(a) *[A] WDBC* (b) *[A] Austrailan* (c) *[A] MNIST*

(d) *[A] WDBC Samples* (e) *[I] Epinions* (f) *[I] Slashdot*

Figure 3. Results on the [A] active learning and [I] influence maximization problems. For [A], BatchGreedy outperforms the state of the art by $\sim 48\%$ on average across the three datasets w.r.t the improvement over random selection. Note the low "price of parallelism" of BatchGreedy: Batch selection performs almost as well as fully sequential selection.

sample hypotheses for each iteration, BatchGreedy is significantly faster than KLR-BMAL, as the cost of BatchGreedy grows linearly w.r.t. the number of hypotheses and number of examples, while KLR-BMAL costs quadratically w.r.t. the number of examples. In fact, for the same settings, it takes KLR-BMAL approx. 50 seconds to select one example, compared to approx. 10 seconds for BatchGreedy.

We also study the impact of the discretization parameter M (i.e., number of hypotheses used to sample the version space), varying it from 300 to 5000, and we plot the results for each setting in Figure 3(d). For the `WDBC` data set, we can observe a statistically significant performance improvement across the 150 trials when increasing the number of sampled hypotheses used from 300 to 2000. Starting from 3000 samples, however, the advantage of introducing more samples begins to decrease dramatically. As there is a linear increase in running time as we employ more samples, we suggest to pick a moderate M to balance efficiency and accuracy.

Multi-stage influence maximization in Social Networks We also apply BatchGreedy to the multi-stage influence maximization problem described in Section 2.1. We use two data sets from the SNAP repository[8]: the `Epinions` social network (with 75879 nodes and 508837 directed edges, where members of the site can decide whether to "trust" each other) and the `Slashdot` social network (with 82168 nodes and 948464 directed edges, where users are allowed to tag each other as friends or foes). For each network, we take the subgraph induced by the top 1000 nodes with largest outdegree. We use the independent cascade model (Kempe et al., 2003). In our simulations, we assume that each person has a certain, fixed probability to influence its neighbors. We choose this probability t according to the edge density of the target network, in our case to 0.05 and 0.03, respectively.

We evaluate the performance of BatchGreedy while varying the size of batches picked at each stage. We repeat the experiments 100 times for all batch sizes (the non-adaptive method corresponds to infinite batch size). In each experiment, we initialize 100 random realizations of the target network based on the edge activation probability, and greedily select the best node in expectation. The results are summarized in Figure 3(e) and Figure 3(f). We observe that for the `Epinions` network, the sequential greedy policy covers 99% of the target network by selecting 244 nodes, while the 10-batch greedy policy, 100-batch greedy policy and non-adaptive greedy policy cost 241, 284, and 584 nodes respectively, to achieve the same coverage.

[8] http://snap.stanford.edu/

Similarly, for the `Slashdot` network those numbers are 330, 319, 343, 765. After incorporation of the first batch of labels, BatchGreedy performs surprisingly well, even competitively with the fully sequential policy.

6. Related Work

Active Learning Fully sequential active learning, where one single unlabeled example is selected to be labeled at a time, has received much attention in the machine learning community. Several heuristic approaches have been proposed that perform well in some applications (e.g., MacKay (1992); Settles (2010)), and theoretical investigations (e.g., Balcan et al. (2006); Dasgupta (2006)) prove bounds on the label complexity. While in some cases exponential reduction is possible, in other settings little reduction over passive learning can be achieved (Dasgupta, 2006). Therefore, it is interesting to study active learning as an optimization problem with the goal to efficiently identify a sufficient yet near-minimal number of informative labels. For both noise-free (Dasgupta, 2004) and noisy (Golovin et al., 2010) active leaning problems, simple greedy policies have been shown to be provably competitive with the optimal (intractable) policy. This paper builds on these approaches, and generalizes the results to batch mode active learning.

Batch mode active learning and submodularity Due to its practical importance, several approaches for batch mode active learning have been proposed. However, previous work has focused on efficiently selecting a single batch comprising examples that are both diverse and informative. Interestingly, the classical notion of submodular set functions has proven useful (Hoi et al., 2006; Guillory & Bilmes, 2011). In Hoi et al. (2006), both individual diversity and informativeness are evaluated w.r.t. the Fisher information matrix of the estimated linear separator. Cesa-Bianchi et al. (2010) and Guillory & Bilmes (2011) investigate the problem of active learning on graph structured data, and provide near-optimal solutions for such problems. In contrast to the prior work, this paper focuses on analyzing batch-mode policies, i.e., sequential construction of batches with the goal to minimize the overall number of labels needed across batches. A work closely related in spirit is that of Desautels et al. (2012), who consider batch mode bandit problems, where the goal is to trade exploration and exploitation, and performance is measured w.r.t. the cumulative regret. In spite of a different context and formalism, a greedy algorithm works provably well in that setting.

Adaptive and interactive submodular optimization For a general introduction to adaptive submodular optimization, see Golovin & Krause (2011). Asadpour et al. (2008) study a special case of the adaptive submodular maximization problem where the random variables must be independent (which is not the case in active learning). Guillory & Bilmes (2010) consider a different formalism for interactive submodular maximization and its applications, analyzing the worst-case cost of fully sequential policies. In contrast to these previous approaches, which focus on the fully sequential case, in this paper we analyze the batch-mode setting, for which we provide approximation bounds as well as practical algorithms.

Influence maximization This problem was first introduced by Domingos & Richardson (2001), and Kempe et al. (2003) prove that the problem of (non-adaptively) finding the optimal k individuals in the network to target requires submodular maximization. Recently, by using adaptive submodularity, Golovin & Krause (2011) generalize the problem to the fully sequential setting where observations are made after each selection. Our results generalize theirs and address the multi-stage setting, which, to our knowledge, is the first attempt to address this natural variant of the influence maximization problem.

7. Conclusions

We presented a general framework for batch mode active learning and stochastic optimization. We analyzed BatchGreedy, an intuitive adaptive greedy approach, and proved its competitiveness with the optimal batch-mode policy. For some problem instances (e.g., multi-stage influence maximization) we proved that, perhaps surprisingly, using batches only incurs a bounded increase of cost as compared to allowing fully sequential selection. In addition to new theoretical results, we empirically demonstrate the effectiveness of BatchGreedy on two real-world applications: Batch mode active learning of linear separators (where BatchGreedy outperforms the state of the art), and multi-stage influence maximization (where we observe a surprisingly small increase in cost compared to the fully sequential strategy). A natural question for future work is to understand more generally for which problems the price of parallelism, i.e., the increase in cost by restricting to information-parallel decisions, is bounded. We believe that our results provide an important step in characterizing the (approximate) tractability of practical active learning and optimization problems.

Acknowledgments. We would like to thank Gábor Bartók for helpful comments. This research was supported in part by SNSF grant 200021 137971, NSF IIS-0953413, DARPA MSEE FA8650-11-1-7156 and ERC StG 307036.

References

Anshelevich, Elliot, Chakrabarty, Deeparnab, Hate, Ameya, and Swamy, Chaitanya. Approximation algorithms for the firefighter problem: Cuts over time and submodularity. In *Algorithms and Computation*. Springer Berlin / Heidelberg, 2009.

Asadpour, Arash, Nazerzadeh, Hamid, and Saberi, Amin. Stochastic submodular maximization. In *WINE*, pp. 477–489, Berlin, Heidelberg, 2008. Springer-Verlag.

Balcan, Maria Florina, Beygelzimer, Alina, and Langford, John. Agnostic active learning. In *ICML*, pp. 65–72, 2006.

Cesa-Bianchi, Nicolò, Gentile, Claudio, Vitale, Fabio, and Zappella, Giovanni. Active learning on trees and graphs. In *COLT*, pp. 320–332, 2010.

Dasgupta, Sanjoy. Analysis of a greedy active learning strategy. In *NIPS*, 2004.

Dasgupta, Sanjoy. Coarse sample complexity bounds for active learning. In Weiss, Y., Schölkopf, B., and Platt, J. (eds.), *Advances in Neural Information Processing Systems 18*, pp. 235–242. MIT Press, Cambridge, MA, 2006.

Desautels, Thomas, Krause, Andreas, and Burdick, Joel. Parallelizing exploration-exploitation tradeoffs with gaussian process bandit optimization. In *ICML*, 2012.

Domingos, Pedro and Richardson, Matt. Mining the network value of customers. In *KDD*, pp. 57–66, 2001.

Golovin, Daniel and Krause, Andreas. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research (JAIR)*, 42:427–486, 2011.

Golovin, Daniel, Krause, Andreas, and Ray, Debajyoti. Near-optimal bayesian active learning with noisy observations. In *NIPS*, December 2010.

Golovin, Daniel, Krause, Andreas, Gardner, Beth, Converse, Sarah, and Morey, Steve. Dynamic resource allocation in conservation planning. In *AAAI*, 2011.

Gonen, Alon, Sabato, Sivan, and Shalev-Shwartz, Shai. Active learning halfspaces under margin assumptions. *CoRR*, abs/1112.1556v3, 2011.

Guillory, Andrew and Bilmes, Jeff. Interactive submodular set cover. In *ICML*, 2010.

Guillory, Andrew and Bilmes, Jeff. Active semi-supervised learning using submodular functions. In *UAI*, 2011.

Hoi, Steven C. H., Jin, Rong, Zhu, Jianke, and Lyu, Michael R. Batch mode active learning and its application to medical image classification. In *ICML*, 2006.

Jain, Prateek, Vijayanarasimhan, Sudheendra, and Grauman, Kristen. Hashing hyperplane queries to near points with applications to large-scale active learning. In *Advances in Neural Information Processing Systems 23*, pp. 928–936. 2010.

Kempe, David, Kleinberg, Jon, and Tardos, Éva. Maximizing the spread of influence through a social network. In *KDD*, pp. 137–146, 2003.

Lovasz, Laszlo. Hit-and-run mixes fast. *Math. Prog*, 86:443–461, 1998.

MacKay, David J.C. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

Settles, Burr. Active learning literature survey. Technical Report 1648, University of Wisconsin-Madison, 2010.

Smith, Robert L. Efficient monte carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 1984.