Tradeoffs for Space, Time, Data and Risk in Unsupervised Learning

Mario Lucic
ETH Zürich
lucic@inf.ethz.ch

Mesrob I. Ohannessian
Microsoft Research - INRIA
mesrob@gmail.com

Amin Karbasi
Yale University
amin.karbasi@gmail.com

Andreas Krause
ETH Zürich
krausea@ethz.ch

Abstract

Faced with massive *data*, is it possible to trade off (statistical) *risk*, and (computational) *space* and *time*? This challenge lies at the heart of large-scale machine learning. Using k-means clustering as a prototypical unsupervised learning problem, we show how we can strategically summarize the data (control space) in order to trade off risk and time when data is generated by a probabilistic model. Our summarization is based on *coreset* constructions from computational geometry. We also develop an algorithm, TRAM, to navigate the space/time/data/risk tradeoff in practice. In particular, we show that for a fixed risk (or data size), as the data size increases (resp. risk increases) the running time of TRAM *decreases*. Our extensive experiments on real data sets demonstrate the existence and practical utility of such tradeoffs, not only for k-means but also for Gaussian Mixture Models.

1 Introduction

The computational and statistical performance of any learning algorithm for a given data set can be described in terms of three parameters: risk, running time, and space usage. The massive growth in datasets, coupled with limited resources in terms of time and space, raises new challenging questions on the accuracy of learning that can be achieved. At the heart of this challenge is to identify the relationships between *risk* ϵ, and the resources we have available, namely, *time* t, *space* s, and *data* n. Most of classical learning theory centers around the question of how risk scales with dataset (or sample) size: How much data n is needed in order to achieve a certain level of risk ϵ (i.e., what is the sample complexity of a given learning task)? In contrast, and from a practical point of view, increasing the data size is a source of computational complexity which typically translates into higher running time t. From this perspective, large data is considered a nuisance rather than a resource for achieving lower risk. As a result, most practical algorithms accumulate data until they exhaust either the time or space constraints and drop the data afterwards.

Related Work. An alternative direction is to investigate *computational and statistical tradeoffs*: using data as a computational resource when available beyond the sample complexity of the learning task. Pioneering this effort, [1] and [2] showed tradeoffs in the realizable PAC learning model. Exploring these tradeoffs has gained much recent attention due to emerging problems in big data. For instance, [3], [4] and [5] showed the existence of such tradeoffs for learning linear classifiers as the data size increases. These tradeoffs are generally achieved by leveraging the fact that as we accumulate more data, the desired risk ϵ becomes easier to reach, thus computationally cheaper but less accurate algorithms can be employed. This idea of *algorithmic weakening* was explored more systematically by [6] using convex relaxations.

Our Contributions. Existing approaches in computational and statistical tradeoffs consider only three of the four parameters: for a desired level of risk ϵ they identify tradeoffs between running time t and data size n. Our primary goal in this paper is to study how *summarization* (i.e., controlling

space) can help navigate the tradeoff between time, data size and risk. In other words, we present a *weakening mechanism*, akin to [6], albeit in a different direction. Instead of weakening *learning algorithms*, we consider weakening the *data representation*. As more data becomes available, more representative elements can be extracted, without incurring much computational cost. Our approach is based on novel computational geometric techniques, called *coresets* [7], where a small amount of most relevant data is extracted from the dataset, while performing the computation on this extracted data guarantees an approximate solution to the original problem. To the best of our knowledge, this paper is a first effort in introducing a methodological data-summarization approach for studying and navigating space/time/data/risk tradeoffs. As a prototypical unsupervised learning problem, we focus on k-means clustering, also known as *vector quantization*, due to its simplicity and practical importance. In this problem, a set of k centers is sought to minimize the expected (squared) distance between data points and the closest center. Finding the optimal centers is NP-hard, but good approximation algorithms are known, e.g., Lloyd's algorithm [8]. We show how coreset constructions for k-means [9, 10, 7, 11, 12] can be used to strategically summarize the data: in order to achieve a fixed precision, the running time can be made to *decrease* as the data set grows, by carefully controlling space usage. We also provide a practical algorithm TRAM that uses existing algorithms for solving k-means (e.g., Lloyd's algorithm, or k-means++) in order to realize this tradeoff in practice. We demonstrate the effectiveness of our summarization strategy on several synthetic and real data sets. We should highlight that k-means clustering is a *non-convex* problem, thus prior computational-statistical tradeoff strategies that heavily relied on convexity cannot be applied in this setting. While we focus on k-means, coresets are available for many other unsupervised learning tasks [12], and we believe that our approach can be applied much more generally. In particular, we empirically demonstrate how such tradeoffs can be achieved for Gaussian Mixture Models (GMMs).

2 The Statistical k-Means Problem

Typically, k-means is viewed as a (combinatorial) optimization problem. We focus instead on the statistical variant. In particular, we assume that an underlying distribution generates i.i.d. samples, and we seek centers that generalize well. More formally, let $\mathbf{P}$ be an *unknown* distribution on $\mathbb{R}^d$ where we assume that it is supported on a ball of radius B at the origin, i.e., for $X \sim \mathbf{P}$ we have $\mathbf{P}(\|X\|_2 \leq B) = 1$. In k-means clustering, any data point $x \in \mathbb{R}^d$ is associated with the closest among a set of k *centers* $c = \{c_1, \cdots, c_k\}$, where $c_i \in \mathbb{R}^d$. We judge the quality of this association by a *risk* defined as $R(c) = \mathbf{E}_{X\sim\mathbf{P}}[\mathrm{d}^2(c, X)]$ between c and a sample X from $\mathbf{P}$, where $\mathrm{d}^2(c, X) = \min_{i=1}^k ||c_i - X||_2^2$. Let $\mathcal{C}$ be the set of all k centers in the ball of radius B at the origin. The *optimal centers* are those that minimize this risk: $c^\star = \arg\min_{c\in\mathcal{C}} R(c)$. Since $\mathbf{P}$ is unknown, we seek centers for a dataset of n samples $X_1, \ldots, X_n$ drawn i.i.d. from $\mathbf{P}$. Any choice of a sequence of functions $\tilde{c}_n$, from $\mathbb{R}^{d\times n} \to \mathbb{R}^{d\times k}$ is called a k-means *procedure*. Out of all such choices, of particular importance is the one that minimizes the *empirical risk*, to obtain the *empirically optimal centers*:

$$R_n(c) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{d}^2(c, X_i), \quad \hat{c}_n = \arg\min_{c\in\mathcal{C}} R_n(c). \tag{1}$$

3 Data Summarization

Data summarization refers to a procedure that takes a data set of size n and replaces it with a smaller set of size s_{proc}, which (approximately) suffices for solving the learning task at hand. This summarization may simply be a truncation without any consideration to the inherent structure of the data (a simple method that is often practiced), or it may be a combination of truncation and strategic sampling that adapts to structure in the data. We denote the truncation size by m_{proc}. One of the main advantages of having summarized data, apart from saving space, is the substantial reduction in running time. For this reason, truncation must be allowed, as otherwise the running time of *any* learning algorithm would grow with the data size. We now formally present these two strategies.

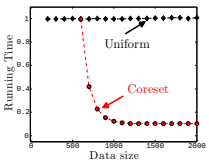
Uniform Subsampling This is the simplest form of data summarization: start with a data set of size n, preserve only the first $s_{\mathsf{subs}} \leq n$ points, and then solve the learning problem by minimizing the empirical risk. In the k-means problem, this amounts to $\tilde{c}_{\mathsf{subs}} = \arg\min_{c\in\mathcal{C}} R_{s_{\mathsf{subs}}}(c)$ where $R_{s_{\mathsf{subs}}}(c) = \frac{1}{s_{\mathsf{subs}}}\sum_{i=1}^{s_{\mathsf{subs}}} \mathrm{d}^2(c, X_i)$. For the uniform subsampler the summarization and truncation sizes are identical, $s_{\mathsf{subs}} = m_{\mathsf{subs}}$. Larger values of s_{subs} promote lower statistical risk but are more expensive to compute. Conversely, computation on a smaller set may be fast but results in higher risk. The uniform subsampler can tune s_{subs} to balance risk with running time.

Tradeoff	Space	Time	Data	Risk
Data-Time	Tune	Objective	Vary	Fixed
Risk-Time	Tune	Objective	Fixed	Vary
Space-Risk	Vary	Tune	Fixed	Objective
Data-Risk	Tune	Fixed	Vary	Objective
Space-Time	Vary	Objective	Tune	Fixed

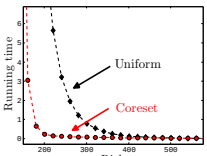
(a) Tradeoffs

Cost	k ↗	n ↗	s ↗
$\epsilon_{\text{modeling}}$	↘		
$\epsilon_{\text{estimation}}$	↗	↘	
$\epsilon_{\text{summarization}}$	↗	→	↘
t	↗	↗	↗

(b) Trends



(c) Data-Time



(d) Risk-Time

Figure 1: (a) Examples of Space-Time-Data-Risk-Tradeoffs. Each one is realized by trading two parameters (green and gray) by constraining (red) and tuning (blue) the remaining ones. (b) Effect of increasing k, n and s on the various errors and running time t. (c) Coreset (red) data-time tradeoffs versus subsampler (black). The plots represent best running time for fixed allowed risk when varying data size, as predicted by our theory (Section 5). (d) Risk-time tradeoff, i.e., best achievable running time for fixed data size when varying the allowed risk. [Time units normalized to the median subsampler time.]

Strategic Sampling *Coresets* are data summaries that are constructed via adaptive sampling, in the spirit of importance sampling. As with the uniform subsampler, we start with data of size n, then truncate it to m_{core} points. Now, instead of using the truncation as is, we perform strategic sampling to propose a set of s_{core} representative points $(Y_j)_{j=1,\cdots,s_{\text{core}}}$, each associated with a non-negative weight w_j, and we solve the learning problem not on the empirical risk, but on a *weighted* variant. In the k-means problem, this amounts to $\tilde{c}_{\text{core}} = \arg\min_{c\in\mathcal{C}} R^w_{s_{\text{core}}}(c)$ where $R^w_{s_{\text{core}}}(c) = \sum_{j=1}^{s_{\text{core}}} w_j \mathrm{d}^2(c, Y_j)$. Coresets strive to be a more faithful/concise representation of the data than uniform samples. The hallmark property of coresets is their ability to approximate the empirical risk, defined in (1), optimized over the starting m_{core} data points.

Definition 1. *A coreset construction is a $(1+\eta)$-approximation, with η a function of the coreset size s_{core}, if the centers $\tilde{c}_{\text{core}}$ satisfy $R_{m_{\text{core}}}(\tilde{c}_{\text{core}}) \leq (1+\eta(s_{\text{core}}))R_{m_{\text{core}}}(\hat{c}_{m_{\text{core}}})$.* [1]

It is worth noting that coresets have the advantage of admitting *streaming* and *parallel* constructions [10], which makes them particularly suited for massive datasets.

4 Space-Time-Data-Risk Tradeoff

Our goal now is to give a precise definition of tradeoffs: how data summarization may lead to trading off representation space, running time, data size, and statistical risk. Let $\tilde{c}_{\text{proc}}(n, m_{\text{proc}}, s_{\text{proc}})$, or $\tilde{c}_{\text{proc}}$ for short, denote a k-means procedure based on data summarization, such as uniform subsampling or coreset summarization. Recall that such a procedure starts with n data points, truncates them to m_{proc} points, summarizes these to s_{proc} (possibly weighted) representative points, and optimizes the (possibly weighted) empirical risk to obtain the set of centers $\tilde{c}_{\text{proc}}$. The *running time*, which we denote by t_{proc}, may be further decomposed into: summarization time $t^{\text{sum}}_{\text{proc}}$ and the time t_{solver} for empirical risk optimization. The former depends on the particular procedure, but the latter can be a generic solver across procedures. We assume that the act of truncation (for both the uniform subsampler and the coreset procedure) has no computational cost. The *statistical risk* of the procedure, which we denote by R_{proc}, is the expected risk, where the expectation is taken with respect to the sample. That is, $R_{\text{proc}} = \mathbf{E}[R(\tilde{c}_{\text{proc}})]$. We can decompose it as follows:

$$R_{\text{proc}} \quad \leq \quad \underbrace{R(c^\star)}_{\epsilon_{\text{model}}} + \underbrace{\mathbf{E}[R(\hat{c}_{m_{\text{proc}}})] - R(c^\star)}_{\epsilon_{\text{est}}} + \underbrace{|\mathbf{E}[R(\tilde{c}_{\text{proc}})] - \mathbf{E}[R(\hat{c}_{m_{\text{proc}}})]|}_{\epsilon_{\text{sum}}}, \tag{2}$$

where ϵ_{model} and ϵ_{est} are the usual *modeling* and *estimation* errors, and ϵ_{sum} is the *summarization* error. The latter represents , respectively. the added risk due to *approximate* data summarization. For coreset procedures, it will depend on the approximation factor $\eta(s_{\text{core}})$.

How to trade off The four dimensions *space, time, data, and risk* put forth in this paper can now be represented by the four parameters $(s_{\text{proc}}, t_{\text{proc}}, m_{\text{proc}}, R_{\text{proc}})$. We can obtain a variety of tradeoffs by constraining some dimensions and optimizing others. We call a subset of the dimensions *feasible* for a procedure, if there exist values of the others that lead to attainable tuples. By exploring the feasible landscape, one can harness various trends. For example, based on the risk decomposition stated above, as we decrease s_{proc}, the risk R_{proc} increases due to the increase in ϵ_{sum}. In contrast, solving the optimization becomes computationally cheaper with smaller s_{proc}. These interactions, illustrated schematically in Figure 1(b) give rise to various tradeoffs. Some of these are listed in Figure 1(a).

[1] Coresets conventionally require approximating the risk at *all* c: for $\varepsilon \in (0,1), \forall c \in \mathcal{C}$, $|R^w_{s_{\text{core}}}(c)/R_m(c) - 1| \leq \varepsilon$. This implies a $(1+\eta)$-approximation with $\eta = 2\varepsilon/(1-\varepsilon)$.

In this paper, we are mainly interested in *(a) data-time tradeoffs*: for R_{proc} fixed below some $\epsilon_{\mathsf{total}}$, can t_{proc} decrease as n increases? and *(b) risk-time tradeoffs*: for some fixed n, can t_{proc} decrease as R_{proc} increases? These two tradeoffs are listed respectively in the first and second rows of the table in Figure 1(a). Data summarization gives us a natural framework to answer those questions: we could achieve such gains by optimizing summarization space s_{proc}. This captures the weakening-through-data-summarization mechanism that we advocate in this paper. Formally, given a data size n and risk $\epsilon_{\mathsf{total}}$, the *optimal running time* function is:

$$t^{\star}_{\mathsf{proc}}(n, \epsilon_{\mathsf{total}}) = \min_{m_{\mathsf{proc}}, s_{\mathsf{proc}}} t_{\mathsf{proc}}(n, m_{\mathsf{proc}}, s_{\mathsf{proc}}), \quad \text{s.t. } R_{\mathsf{proc}}(m_{\mathsf{proc}}, s_{\mathsf{proc}}) \leq \epsilon_{\mathsf{total}}, m_{\mathsf{proc}} \leq n. \tag{3}$$

Note: for fixed $\epsilon_{\mathsf{total}}$ and as n increases, the optimal running time $t^{\star}_{\mathsf{proc}}$ is non-increasing by construction. Similarly, for fixed n and as $\epsilon_{\mathsf{total}}$ increases, the optimal running time $t^{\star}_{\mathsf{proc}}$ is non-increasing.

Definition 2. *We say that a k-means procedure offers a (non-trivial)* data-time tradeoff *if, for a given desired total risk $\epsilon_{\mathsf{total}}$, the running time $t^{\star}_{\mathsf{proc}}(\cdot, \epsilon_{\mathsf{total}})$ is decreasing for some range of n. We say that the procedure offers a (non-trivial)* risk-time tradeoff *if, for a given data size n, $t^{\star}_{\mathsf{proc}}(n, \cdot)$ is decreasing for some range of $\epsilon_{\mathsf{total}}$.*

5 Analysis

We have thus far motivated and laid out a clear paradigm of tradeoffs via data summarization. But are such tradeoffs even possible? In this section, we show that the answer is *yes*. We focus in particular on showing that nontrivial data-time tradeoffs (Definition 2) do indeed exist.

For the uniform subsampler the data-time tradeoff is necessarily trivial. To see this, let $n_{\mathrm{f}}(\epsilon_{\mathsf{total}})$ be the smallest data size n when $\epsilon_{\mathsf{total}}$ becomes feasible, i.e. $\epsilon_{\mathsf{model}} + \epsilon_{\mathsf{est}}(n) \leq \epsilon_{\mathsf{total}}$. Then for all $n > n_{\mathrm{f}}(\epsilon_{\mathsf{total}})$, the uniform subsampler has no incentive to use more than $m_{\mathsf{subs}} = n_{\mathrm{f}}(\epsilon_{\mathsf{total}})$ samples, since otherwise its running time would be greater (for unneeded risk reduction). This means that $t^{\star}_{\mathsf{subs}}(\cdot, \epsilon_{\mathsf{total}})$ is undefined for $n < n_{\mathrm{f}}(\epsilon_{\mathsf{total}})$, and is flat beyond that.

The more interesting question is thus: Can coreset procedures give non-trivial data-time tradeoffs that improve on the uniform subsampler? Our main result answers in the affirmative. Informally:

Main Result (Existence of Tradeoffs)**.** *Let the following conditions hold for a coreset procedure:*

(a) The summarization is time-efficient (its running time is negligible relative to that of the solver).

(b) The summarization is sample-efficient (the approximation factor vs. summarization size decays no slower than the estimation error vs. sample size).

(c) The estimation error decays fast ($\sim$ power law).

(d) The solver is slow (at least super-linear).

Then, for small enough risks, the procedure admits a non-trivial tradeoff, and its optimal running time dominates that of the uniform subsampler for large enough sample sizes. Existing bounds and coreset constructions do satisfy these conditions.

The statement of this problem can be formalized rigorously. The conditions are rather natural. For example, if summarization is much more time-intensive than solving (if Conditon (a) fails), then we can't expect to benefit from summarization. To verify these conditions, we use as a concrete construction the one given in [13], where $\eta(s_{\mathsf{core}})$ behaves roughly like $\mathcal{O}(1/\sqrt{s_{\mathsf{core}}})$, therefore satisfies Condition (b), since generally $\epsilon_{\mathsf{est}}(m) = \mathcal{O}(1/\sqrt{m})$. Thanks to the latter, Condition (c) is also satisfied. Lastly even the most optimistic running times of heuristic algorithms (of the Lloyd variety) are super-linear, and thus Condition (d) follows.

6 Data-driven Tradeoff Navigation

So far we demonstrated tradeoffs in k-means by considering analytical models. In practice, however, even if a tradeoff exists, it is a priori unclear how to harness it: one would seemingly need a "tuning oracle" to adjust the procedure to yield an optimal tradeoff, by selecting optimal truncation and summarization sizes. An exhaustive search for such an adjustment is useful for illustration, but it defeats the purpose of the endeavor, which is to yield a practical algorithm whose running time decreases with more data. In this section, we address this challenge by proposing a *TRadeoff nAvigation algorithM (*TRAM*)*. It uses a limited amount of additional validation data to explore the summarization landscape, and leads to a summarization that exhibits acceptable

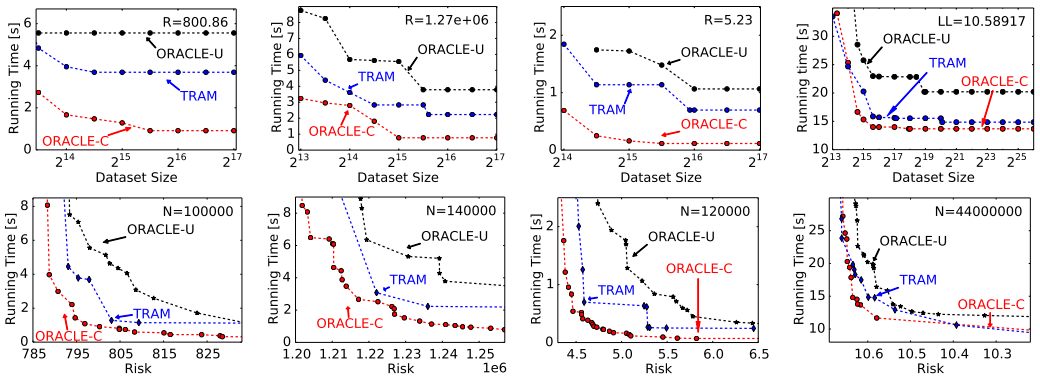


Figure 2: Results for SYNTHETIC (S), KDD2004BIO (K), CSN (C) and WEBSCOPE (W) data, column-wise. Figures in the first row show data-time tradeoffs: best running time for fixed risk tolerance and varying data sizes (cf. Figure 1(c)). Tradeoffs exist: running time decreases with increasing data size. Furthermore, the coreset procedure dominates uniform subsampling, and TRAM tracks the coreset tradeoff closely, with limited overhead. Figures in the second row risk-time tradeoffs: best running time for fixed data size and varying risk tolerance (cf. Figure 1(d)).

loss in risk ϵ_{total}, time $t^\star$, and space $s^\star$, thus effectively approximating a tuning oracle. We focus specifically on data-time tradeoffs via coreset data-summarization schemes, though the approach is potentially extensible to other tradeoffs and procedures.

A TRadeoff nAvigation algorithM (TRAM) The idea of TRAM is as follows: search for a good summarization by starting small then growing until the desired risk is achieved. The challenge is that the risk cannot be known exactly and needs to be tested using data. We therefore have a compromise: if we stop too early we miss the target, and if we stop too late we spend too much on computation. The analysis shows that the algorithm achieves a certain balance. We assume we have validation samples beyond the main body of data. In practice the data itself is partitioned to provide these samples. More specifically, we use a $a[i] = 4ib\log(1/\delta)/{\epsilon_{\text{total}}}^2$ of these points at iteration i. We also assume $t_{\text{sum}}(m) = \mathcal{O}(m)$ and $t_{\text{solver}}(s) = \mathcal{O}(s^\beta)$, with known β.

Algorithm TRadeoff nAvigation algorithM (TRAM)

1: **Input:** Data of size n; risk level ϵ_{total}; validation data of size a; accuracy parameter $\delta > 0$.
2: **Initialization:** Start with a truncation of size $m[0] < n$ and a coreset size of $s[0]$.
3: **repeat**
4: **Iteration step** i**:** Summarize the $m[i]$-truncation to a coreset of size $s[i]$, and solve for the centers $\tilde{c}[i]$. Increment: $m[i+1] \leftarrow 2m[i] \wedge n$, and $s[i+1] \leftarrow 2^{1/\beta}s[i]$. Use a portion $a[i]$ of the validation data to evaluate the empirical risk of $\tilde{c}[i]$.
5: **until** $R_{a[i]}(\tilde{c}[i]) < 1.5\epsilon_{\text{total}}$.
6: **Output:** The last set of centers $\tilde{c}[i]$.

Theorem. *Let T and J denote the running time and number of iterations of* TRAM *respectively. Under an assumption of strong feasibility with probability $1-\lambda$, given data of size n, a base risk ϵ_{total}, and parameter $\delta < \frac{1}{5}$, with probability at least $(1-\lambda)(1-5\delta)$,* TRAM*:*

▷ *runs for time $T \leq 8t^{\star 2} + \mathcal{O}\left(\log\frac{1}{\delta}\log_2^2 t^\star\right)$,*

▷ *uses $a[J] \leq \mathcal{O}\left(\log\frac{1}{\delta}\log_2 t^\star\right)$ validation points,*

▷ *and produces centers $\tilde{c}$ with risk $R(\tilde{c}) \leq 2\epsilon_{\text{total}}$.*

7 Experimental Results

We now describe our experiments, empirically establishing the existence of tradeoffs, and evaluating the performance of TRAM.

Setup Given a dataset $\mathcal{X} \subseteq \mathbb{R}^d$ and some ϵ_{total}, we wish to find the minimum computational cost of obtaining a k-means solution with risk less than or equal to ϵ_{total}. We simulate various dataset sizes by restricting individual experiments to a random subset of $\mathcal{X}$. For each pair of data size $n_i \in \mathcal{N}$ and summary size $s_j \in \mathcal{S}$ we sample n_i instances i.i.d. from $\mathcal{X}$ and summarize the sample

with a summary of size s_j and solve the problem on the summary. We repeat the latter 50 times and report the average time and risk obtained. For the uniform subsampler, s_j refers to the subsample size, and for the coresets it refers to the size of the coreset. We denote the cumulative running time of summarizing and solving the problem on the summary by $t(n_i, s_j)$ and obtained risk by $R(n_i, s_j)$. For each procedure, let $\Lambda_{\mathsf{proc}} = \{(n, t(n, s), R(n, s)) \mid n \in \mathcal{N}, s \in \mathcal{S}\}$.

We can now leverage Λ_{proc} to characterize various tradeoffs. For example, to capture the data-time tradeoff for a particular size n we find the minimum running time t' such that $\exists (m, t', R) \in \Lambda_{\mathsf{proc}}$, with $m < n$ and $R \leq \epsilon_{\mathsf{total}}$. Searching Λ_{proc} yields Pareto-optimal boundaries of two oracles: coreset-based (ORACLE-C) and uniform-sampling-based (ORACLE-U). To show that one can navigate the space/time/data/risk tradeoffs in practice using TRAM, we showcase it alongside the oracles in Figure 2. Note that constructing the oracles is computationally prohibitive as it entails a full grid search over $\mathcal{N}$ and $\mathcal{S}$. Nevertheless, the reported times assume the oracles *know* the best summarization right away.

Datasets SYNTHETIC — We generate synthetic data of $100,000$ points in $\mathbb{R}^{100}$ from a mixture of Gaussians. We choose $k = 100$ centers in $[0, 100]^{100}$ and set them as means for the k spherical Gaussian distributions with $\Sigma = 5I$. The relative magnitudes of the clusters are sampled from an exchangeable Dirichlet distribution with $\alpha = 1/20$.

KDD2004BIO — This dataset was used for the Protein Homology Prediction Task in KDD Cup 2004. It contains $145,751$ instances and 74 attributes that describe the match between two proteins. We fit k-means with $k = 150$.

CSN — The Community Seismic Network (CSN) uses smart phones with accelerometers as inexpensive seismometers for earthquake detection. [14] compiled 7 GB of acceleration data and computed 17-dimensional feature vectors. We apply k-means with $k = 200$.

YAHOO! WEBSCOPE R6A — $45,811,883$ instances in $\mathbb{R}^6$ that represent the user click log displayed on Yahoo! Front Page. For this dataset, we extend our framework from k-means to Gaussian Mixture Models. We fit a GMM with $k = 200$ components. The risk is now defined as negative log-likelihood on the hold-out data.

Parameters For the k-means clustering problem we use the coreset construction from [13], and a weighted variant of the k-means++ algorithm to solve the problem on the the subsample. In the case of GMMs, we use the coreset construction from [15] and a weighted EM for GMMs. We consider sizes summarization sizes between 100 and 20000. For TRAM, we start with summarization size and truncation size inversely proportional to the risk required. At every iteration, we double the truncation size and take 1.5-fold of the summarization size. $1/5^{\text{th}}$ of the sample is used for validation, with a δ of 0.1.

Observations The plots in the first row in Figure 2 show the Pareto-optimal boundary for a fixed risk as data size varies. There is a data-time tradeoff as predicted from theory. Furthermore, TRAM traces the solutions achieved by the coreset oracle, implying that we *can* navigate tradeoff curves without oracles. Remarkably, TRAM remains better than the uniform subsampler oracle, eventhough either oracle takes orders of magnitude more time to obtain by exhaustive search. The second row illustrates the existence of a risk-time tradeoffs also: for fixed data size, the time to guarantee a desired risk decreases as the risk increases. Solving the problem on the whole dataset is often out of the question (in the case of GMMs, it may take weeks). Summarization slashes this time down (minutes instead of weeks). However, because the coreset procedure can achieve a faster time even as it accesses a larger portion of data, it will be more likely to guarantee a desired risk, as compared to the uniform subsampler, at least for *interesting* (small) risk levels. And TRAM opimizes this.

8 Conclusions

We explored space/time/data/risk tradeoffs achievable via coreset-based data-summarization. Our theory predicts and our empirical results demonstrate the existence and utility of such tradeoffs. We further showed how such tradeoffs can be practically realized via a novel algorithm, TRAM. While our analysis focused on k-means, our insights are more generally applicable. In particular, we empirically demonstrated tradeoffs in learning Gaussian Mixture Models. Approaches that optimize cost functions related to the quantization error, such as small-variance limits of non-parametric Bayesian models [16], may also immediately benefit from our results. We thus strongly believe that our results present an important step towards understanding tradeoffs in large-scale unsupervised learning. Lastly, given promising summarization-style techniques [17, 18, 19], similar results may also be possible in supervised learning.

References

[1] Scott E Decatur, Oded Goldreich, and Dana Ron. Computational sample complexity. *SIAM Journal on Computing*, 29(3):854–879, 2000.

[2] Rocco A Servedio. Computational sample complexity and attribute-efficient learning. In *Proceedings of the 31st Annual Symposium on Theory of Computing*, pages 701–710. ACM, 1999.

[3] Léon Bottou and Olivier Bousquet. The Tradeoffs of Large-Scale Learning. In *Advances in Neural Information Processing Systems*, volume 20, pages 161–168. NIPS Foundation, 2008.

[4] Shai Shalev-Shwartz and Nathan Srebro. SVM optimization: inverse dependence on training set size. In *International Conference on Machine Learning*, pages 928–935, 2008.

[5] Aharon Birnbaum and Shai S Shwartz. Learning halfspaces with the zero-one loss: time-accuracy tradeoffs. In *Advances in Neural Information Processing Systems*, pages 935–943, 2012.

[6] Venkat Chandrasekaran and Michael I Jordan. Computational and statistical tradeoffs via convex relaxation. *Proc. Natl. Acad. Sci. U.S.A.*, 110(13):E1181–90, March 2013.

[7] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric Approximation via Coresets. *Combinatorial and computational geometry*, 52:1–30, 2005.

[8] Stuart Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[9] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. An efficient k-means clustering algorithm: Analysis and implementation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):881–892, 2002.

[10] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *STOC*, pages 291–300. ACM, 2004.

[11] Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the 23rd Annual Symposium on Computational Geometry*, pages 11–18. ACM, 2007.

[12] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for *k*-means, PCA and projective clustering. In *SODA*, 2013.

[13] Dan Feldman and Michael Langberg. A Unified Framework for Approximating and Clustering Data. In *STOC*, pages 569–578. ACM, 2011.

[14] Matthew Faulkner, Michael Olson, Rishi Chandy, Jonathan Krause, K Mani Chandy, and Andreas Krause. The next big one: Detecting earthquakes and other rare events from community-based sensors. In *IPSN*, pages 13–24, 2011.

[15] Dan Feldman, Andreas Krause, and Matthew Faulkner. Scalable training of mixture models via coresets. pages 2142–2150, 2011.

[16] Ke Jiang, Brian Kulis, and Michael I Jordan. Small-variance asymptotics for exponential family Dirichlet process mixture models. In *Advances in Neural Information Processing Systems*, pages 3167–3175, 2012.

[17] Dmitry Pavlov, Darya Chudova, and Padhraic Smyth. Towards scalable support vector machines using squashing. In *KDD*, pages 295–299, 2000.

[18] Gökhan H. Bakir, Léon Bottou, and Jason Weston. Breaking SVM Complexity with Cross-Training. In *NIPS*, 2004.

[19] Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. Core vector machines: fast SVM training on very large data sets. *JMLR*, 6:363–392, 2005.