The AAAI-17 Workshop on
Distributed Machine Learning
WS-17-08

Data Driven Resource Allocation for Distributed Learning

Travis Dick
Carnegie Mellon University
tdick@cs.cmu.edu

Mu Li
Carnegie Mellon University
muli@cs.cmu.edu

Venkata Krishna Pillutla
University of Washington
pillutla@cs.washington.edu

Colin White
Carnegie Mellon University
crwhite@cs.cmu.edu

Maria Florina Balcan
Carnegie Mellon University
ninamf@cs.cmu.edu

Alex Smola
Carnegie Mellon University
and AWS Deep Learning
alex@smola.org

Abstract

In distributed machine learning, data is dispatched to multiple machines for processing. Motivated by the fact that similar data points often belong to the same or similar classes, and more generally, classification rules of high accuracy tend to be "locally simple but globally complex" (Vapnik and Bottou 1993), we propose data dependent dispatching that takes advantage of such structure. We present an in-depth analysis of this model, providing new algorithms with provable worst-case guarantees, analysis proving existing scalable heuristics perform well in natural non worst-case conditions, and techniques for extending a dispatching rule from a small sample to the entire distribution. We overcome novel technical challenges to satisfy important conditions for accurate distributed learning, including fault tolerance and balancedness. We empirically compare our approach with baselines based on random partitioning, balanced partition trees, and locality sensitive hashing, showing that we achieve significantly higher accuracy on both synthetic and real world image and advertising datasets. We also demonstrate that our technique strongly scales with the available computing power.

Introduction

Motivation and Overview: We consider distributed learning settings where massive amounts of data are collected centrally, and for space and efficiency reasons this data must be dispatched to distributed machines in order to perform the processing needed (Li et al. 2014; Zhang, Duchi, and Wainwright 2012). The simplest approach and what past work (both theoretical and empirical) has focused on is to perform the dispatching randomly (Zhang, Duchi, and Wainwright 2012; Zhang et al. 2013). Random dispatching has the advantage that dispatching is easy, and because each machine receives data from the same distribution, it is rather clean to analyze theoretically. However, since the distributions of the data on each machine are identical, such techniques could lead to sub-optimal results in practice in terms of the accuracy of the resulting learning rule. Motivated by the fact that in practice, similar data points tend to have the same or similar classification, and more generally, classification rules of high accuracy tend to be "locally simple but globally complex" (Vapnik and Bottou 1993), we propose a new paradigm for performing *data-dependent dispatching* that takes advantage of such structure by sending similar datapoints to the same machines. For example, a *globally* accurate classification rule may be complicated, but each machine can accurately classify its *local* region with a simple classifier.

We introduce and analyze dispatching techniques that partition a set of points such that similar examples end up on the same machine/worker, while satisfying key constraints present in a real world distributed system including balancedness and fault-tolerance. Such techniques can then be used within a simple, but highly efficient distributed system that first partitions a small initial segment of data into a number of sets equal to the number of machines. Then each machine locally and independently applies a learning algorithm, with no communication between the workers during training. In other words, the learning is embarrassingly parallel. At prediction time, we use a super-fast sublinear algorithm for directing new data points to the most appropriate machine.

Our Contributions: We propose a novel scheme for partitioning data which leads to better accuracy in distributed machine learning tasks, and we give a theoretical and experimental analysis of this approach. We present new algorithms with provable worst-case guarantees, analysis proving existing scalable heuristics perform well in natural non worst-case conditions, techniques for extending a dispatching rule from a small sample to the entire distribution, and an experimental evaluation of our proposed algorithms and several baselines on both synthetic and real-world image and advertising datasets. We empirically show that our method strongly scales and that we achieve significantly higher accuracy over baselines based on random partitioning, balanced partition trees, and locality-sensitive hashing.

In our framework, a central machine starts by clustering a small sample of data into roughly equal-sized clusters, where the number of clusters is equal to the number of available machines. Next, we extend this clustering into an efficient dispatch rule that can be applied to new points. This dispatch rule is used to send the remaining training data to the appropriate machines and to direct new points at prediction time. In this way, similar datapoints wind up on the same machine. Finally, each machine independently learns a classifier using its own data (in an embarrassingly parallel manner). To perform the initial clustering used for dispatch, we use classic clustering objectives (k-means, k-median, and k-center). However, we need to add novel constraints to ensure that the clusters give a data partition that respects the constraints of real distributed learning systems:

Balancedness: We need to ensure our dispatching procedure balances the data across the different machines. If a machine receives much more data than other machines, then it will be the bottleneck of the algorithm. If any machine receives very little data, then its processing power is wasted. Thus, enforcing upper and lower bound constraints on the cluster sizes leads to a faster, more efficient setup.

Fault-Tolerance: In order to ensure that our system is robust to machine failures, we assign each point to multiple distinct clusters. This way, even if a machine fails, the data on that machine is still present on other machines. Moreover, this has the added benefit that our algorithms behave well on points near the boundaries of the clusters. We say a clustering algorithm satisfies p-replication if each point is assigned to p distinct clusters.

When designing clustering algorithms, adding balancedness and fault tolerance makes the task significantly harder. Prior work has considered upper bounds on the cluster sizes (Li 2014b; Byrka et al. 2015) and lower bounds (Ahmadian and Swamy 2016), but no prior work has shown provable guarantees with upper and lower bounds on the cluster sizes simultaneously. With upper bounds, the objective functions are nondecreasing as the number of clusters k increases, but with lower bounds we show the objective function can oscillate arbitrarily with respect to k. This makes the problem especially challenging from a combinatorial optimization perspective. Existing capacitated clustering algorithms work by rounding a fractional linear program solution, but the erratic nature of the objective function makes this task more difficult for us.

The balance constraints also introduce challenges when extending a clustering-based partitioning from a small sample to unseen data. The simple rule that assigns a new point to the cluster with the nearest center provides the best objective value on new data, but it can severely violate the balance constraints. Therefore, any balanced extension rule must take into account the distribution of data.

We overcome these challenges, presenting a variety of complementary results, which together provide strong justification for our distributed learning framework. We summarize each of our main results below.

• **Balanced fault-tolerant clustering:** We provide the first clustering algorithms with provable guarantees that simultaneously handle upper and lower bounds on the cluster sizes, as well as fault tolerance. Clustering is NP-hard and adding more constraints makes it significantly harder, as we discuss in Section . For this reason, we first devise approximation algorithms with strong worst-case guarantees, demonstrating this problem is tractable. Specifically, in Section we provide an algorithm that produces a fault-tolerant clustering that approximately optimizes the k-means objective while also roughly satisfying the given upper and lower bound constraints. The full version of the paper includes algorithms for the k-median, and k-center objectives. At a high level, our algorithm proceeds by first solving a linear program, followed by a careful balance and replication aware rounding scheme. We use a novel min-cost flow technique to finish off rounding the LP solution into a valid clustering solution.

• **k-means++ under stability:** We give complementary results showing that for 'typical' problem instances, it is possible to achieve better guarantees with simpler, more scalable algorithms. Specifically, in Section we show the popular k-means++ algorithm outputs a balanced clustering with stronger theoretical guarantees, provided the data satisfies a natural notion of stability. We make nontrivial extensions of previous work to ensure the upper and lower size constraints on the clusters are satisfied. No previous work gives provable guarantees while satisfying both upper and lower bounds on the cluster sizes, and Sections and may be of independent interest beyond distributed learning.

• **Efficient clustering by subsampling:** For datasets large enough to require distributed processing, clustering the entire dataset is prohibitively expensive. A natural way to avoid this cost is to only cluster a small subset of the data and then efficiently extend this clustering to the entire dataset. In Section we show that assigning a new example to the same p clusters as its nearest neighbor in the clustered subsample approximately preserves both the objective value and all constraints. We also use this technique at prediction time to send new examples to the most appropriate machines.

• **Experimental results:** Section presents experiments evaluating both our LP rounding algorithms and k-means++ together with our nearest neighbor extension. We show that our technique strongly scales and that it achieves significantly higher accuracy than baselines based on random partitioning, balanced partition trees, and locality sensitive hashing on several synthetic and real-world datasets.

For the full version of this paper, please refer to (Dick et al. 2015).

Related Work: Currently, the most popular method of dispatch in distributed learning is random dispatch (Zhang et al. 2013; Zhang, Duchi, and Wainwright 2012). Other papers have studied partitioning data in distributed machine learning, but without formal guarantees on balancing data or the quality of the clusters produced (Cooper et al. 2008; Wei et al. 2015; You et al. 2015).

Fault Tolerant Balanced Clustering

In this section, we give an algorithm to cluster a small initial sample of data to create a dispatch rule that sends similar points to the same machine. We measure the similarity of points in the same cluster using the k-means objective, and in the full version, we also consider the k-median and k-center objectives. We impose upper and lower bounds on the cluster sizes and replication constraints. This is the first algorithm with provable guarantees to simultaneously handle both upper and lower bounds on the cluster sizes.

A clustering instance consists of a set V of n points, and a distance metric d. Given two points i and j in V, denote the distance between i and j by $d(i,j)$. The task is to find a set of k centers $C = \{c_1, \dots, c_k\}$ and assignments of each point to p of the centers $f : V \to \binom{C}{p}$, where $\binom{C}{p}$

represents the subset of C^p with no duplicates, to minimize $\sum_{i\in V}\sum_{c\in f(i)} d(i,c)^2$.

We add size constraints $0 < \ell \leq L < 1$, also known as capacity constraints, so each cluster must have a size between $n\ell$ and nL. For simplicity, we assume these values are integral (or replace them by $\lceil n\ell\rceil$ and $\lfloor nL\rfloor$ respectively). Before we present our approximation algorithm, we discuss the challenges introduced by these size constraints.

Structure of Balanced Clustering: It is well-known that exactly minimizing the k-means objective is NP-hard (even without the capacity and fault tolerance generalizations) (Jain et al. 2003). In uncapacitated clustering and clustering with upper bounds only, the cost of the optimal solution is nonincreasing as k increases, because we can pick any point and make it a center, so the new center is now distance 0 from a center. However, when there are lower bounds on the cluster sizes, we prove that the value of the optimal solution (for k-means, k-median, or k-center) can oscillate *arbitrarily many times* as k increases. The proof requires an intricate clustering construction, which we include in the full version.

Approximation Algorithm: In light of these difficulties, one might ask whether any approximation algorithm exists for this problem. We answer affirmatively, by extending previous work (Li 2014a) to fit our more challenging constrained optimization problem. Our algorithm returns a clustering whose cost is at most a constant factor multiple of the optimal solution, while violating the capacity and replication constraints by a small constant factor.

```
1. Find a solution to the following linear program:
       $\min_{x,y} \sum_{i,j\in V} x_{ij} d(i,j)^2$   s.t.
       (a) $\forall j \in V : \sum_{i\in V} x_{ij} = p$;   (b) $\sum_{i\in V} y_i \leq k$;
       (c) $\forall i \in V : \ell y_i \leq \sum_{j\in V} \frac{x_{ij}}{n} \leq L y_i$;
       (d) $\forall i,j \in V : 0 \leq x_{ij} \leq y_i \leq 1$.
2. Greedily place points into a set $\mathcal{M}$ from lowest $C_j := \sum_i x_{ij} d(i,j)^2$ to highest (called "monarchs"), adding point $j$ to $\mathcal{M}$ if it is not within distance $4C_j$ of any point already in $\mathcal{M}$. Partition the points into coarse clusters (called "empires") using the Voronoi partitioning of the monarchs.
3. For each empire $\mathcal{E}_u$ with total fractional opening $Y_u := \sum_{i\in\mathcal{E}_u} y_i$, give opening $Y_u/\lfloor Y_u\rfloor$ to the $\lfloor Y_u\rfloor$ closest points to $u$ and all other points opening 0.
4. Round the $x_{ij}$'s by constructing a minimum cost flow problem on a bipartite graph of centers and points, setting up demands and capacities to handle the bounds on cluster sizes.
```

Algorithm 1: Balanced clustering with fault tolerance

Theorem 1. *Algorithm 1 returns a constant factor approximate solution for balanced k-means with p-replication problem for $p > 1$, where the upper capacity constraints are violated by at most a factor of $\frac{p+2}{p}$, and each point can be assigned to each center at most twice.*

Proof sketch. The first step is to solve the linear program in Algorithm 1 to obtain a fractional solution to the clustering problem. For each point i, the value y_i is the fraction to which this point is opened as a center (which we refer to as the 'opening' of i). Next, we perform a coarse partitioning of the points into 'empires', such that each empire has total opening ≥ 1, and each point is at most $4C_i$ from the monarch of its empire. Then we aggregate all fractional openings to the center of their respective empires. The key insight is that p-replication helps to mitigate the capacity violation, so we end up with $\leq k$ centers total, and the cost of the aggregation procedure can be bounded using the triangle inequality. Finally, we use a novel min-cost flow technique to round the center assignment variables while simultaneously handling the upper and lower bounds on the cluster sizes. □

In the full version of this paper, we show a more involved algorithm specifically for k-center which achieves a 6-approximation with *no violation* to the capacity or replication constraints.

Balanced Clustering Under Stability

In the previous section, we showed an LP-based algorithm which provides theoretical guarantees even on adversarially chosen data. Often real-world data has inherent structure that allows us to use more scalable algorithms and achieve even better clusters (Balcan, Blum, and Gupta 2013; Ostrovsky et al. 2006). In our distributed ML framework, this translates to being able to use a larger initial sample for the same computational power (Section analyzes the effect of sample size). In this section, we prove the popular k-means++ algorithm outputs clusters very close to the optimal solution, provided the data satisfies a natural notion of stability called *approximation stability* (Balcan, Blum, and Gupta 2013; Gupta, Roughgarden, and Seshadhri 2014; Balcan, Haghtalab, and White 2016).

Specifically, we show that given a balanced clustering instance in which clusterings close in *value* to $\mathcal{OPT}$ are also close in terms of the clusters themselves, assuming $L \in O(\ell)$, then k-means++ with a simple pruning step (Ostrovsky et al. 2006) outputs a solution close to optimal. We overcome key challenges that arise when we add upper and lower bounds to the cluster sizes. We summarize the result below, and include the details in the full version.

Approximation Stability: Given a clustering instance (S,d) and inputs ℓ and L, and let $\mathcal{OPT}$ denote the cost of the optimal balanced clustering. Two clusterings $\mathcal{C}$ and $\mathcal{C}'$ are ϵ*-close*, if only an ϵ-fraction of the input points are clustered differently in the two clusterings, i.e., $\min_\sigma \sum_{i=1}^k |\mathcal{C}_i \setminus \mathcal{C}'_{\sigma(i)}| \leq \epsilon n$, where σ is a permutation of $[k]$.

Definition 1 (Balcan, Blum, and Gupta (2013)). *A clustering instance (S, d) satisfies $(1+\alpha, \epsilon)$-approximation stability with respect to balanced clustering if all clusterings $\mathcal{C}$ with cost$(\mathcal{C}) \leq (1+\alpha) \cdot \mathcal{OPT}$ are ϵ-close to $\mathcal{C}$.*

We show that sampling $k \log k$ centers using k-means++, followed by a greedy center-pruning step, (introduced by Ostrovsky et al. (2006)) is sufficient to cluster well with high probability, assuming $(1+\alpha, \epsilon)$-approximation stability for balanced clustering. Our results improve over Agarwal, Jaiswal, and Pal (2015), who showed this algorithm outputs a good clustering with probability $\Omega(\frac{1}{k})$ for standard (unbalanced) clustering under approximation stability. Formally, our result is the following.

Theorem 2. *Given $\frac{\epsilon \cdot k}{\alpha} < \rho < 1$, k-means++ seeding with a greedy pruning step outputs a solution that is $\frac{1}{1-\rho}$ close to the optimal solution with probability $> 1 - O(\rho)$, for clustering instances satisfying $(1+\alpha, \epsilon)$-approximation stability for the balanced k-means objective, with $\frac{L}{\ell} \in O(1)$.*

Proof sketch. Intuitively, $(1+\alpha, \epsilon)$-approximation stability forces the clusters to become "spread out", i.e., the radius of any cluster is much smaller than the inter-cluster distances. This allows us to show for 2-means clustering, the k-means++ seeding procedure will pick one point from each cluster with high probability. However, if we induct on the number of clusters, the probability of success becomes exponentially small in k. We circumvent this issue in a manner similar to Ostrovsky et al. (2006), by sampling $k \log k$ centers, and carefully deleting centers greedily, until we are left with one center per cluster with high probability. □

Efficient Clustering by Subsampling

For datasets large enough to require a distributed learning system, it is expensive to apply a clustering algorithm to the entire dataset. In this section, we show that we can first cluster a small subsample of data and then efficiently extend this clustering to the remaining data. In our technique, each point in the dataset is assigned to the same p clusters as its nearest neighbor in the clustered subsample. In fact, this technique can be used to dispatch any point from the space $\mathcal{X}$ containing the data. We show that the clustering induced over $\mathcal{X}$ approximately inherits all of the desirable properties of the clustered subsample: good objective value, balanced clusters, and replication.

We measure the quality of a clustering of $\mathcal{X}$ as follows: given a data distribution μ over $\mathcal{X}$, our goal is to find a clustering with centers $C = \{c_1, \dots, c_k\}$ and an assignment function $f : \mathcal{X} \to \binom{C}{p}$ for the entire space that minimizes $Q(f, C) = \mathbb{E}_{x \sim \mu}[\sum_{c \in f(x)} d(x, c)^2]$ subject to the balance constraints $\mathbb{P}_{x \sim \mu}(c_j \in f(x)) \in [\ell, L]$ for all j.

The simplest approach to extend a clustering of small subsample is to assign a new example x to the p clusters with the closest centers. This strategy incurs the lowest cost for new examples, but it may severely violate the balance constraints if the distribution is concentrated near one center.

Instead, given a clustering of the subsample S, our technique assigns a new example x to the same p clusters as its nearest neighbor in S, denoted by $NN_S(x)$. Some points in S represent more probability mass of μ than others, so we use a second independent sample S' to estimate weights for each point in S that are used in a weighted version of the objective and balance constraints. Pseudocode is given in Algorithm 2. We obtain the following guarantee:

Theorem 3. *For any $\epsilon, \delta > 0$, let $(\bar{g}_S, C_S)$ be the output of Algorithm 2 with parameters k, p, ℓ, L and second sample size $n' = O\big(\frac{1}{\epsilon^2}(n + \log \frac{1}{\delta})\big)$. Let (f^*, C^*) be any clustering of $\mathcal{X}$ and (g_S^*, C_S^*) be an optimal clustering of S under Q_S satisfying the weighted balance constraints (ℓ, L). Suppose that $Q_S(g_S, C_S) \leq r \cdot Q_S(g_S^*, C_S^*) + s$. Then w.p. $\geq 1 - \delta$ over the second sample, the output $(\bar{g}_S, C_S)$ satisfies the balance constraints with $\ell' = \ell - \epsilon$ and $L' = L + \epsilon$ and*

$$Q(\bar{g}_S, C_S) \leq 4r \cdot Q(f^*, C^*) + 2s + 4(r+1)pD^2\epsilon + 2p(2r+1)\alpha(S) + 4r\beta(S, \ell + \epsilon, L - \epsilon),$$

where D is the diameter of $\mathcal{X}$, the quantity $\alpha(S) = \mathbb{E}_{x \sim \mu}[d(x, \mathrm{NN}_S(x))^2]$ measures how well μ is approximated by S, and $\beta(S, \ell, L) = \min_{h,C}\{Q(\bar{h}, C) - Q(f^*, c^*)\}$ measures the loss incurred by restricting to clusterings that are constant over the Voronoi tiles of S.

Proof sketch. The second sample size n' is large enough that weights $\hat{w}_i$ are good estimates of the true probability mass represented by each point $x_i \in S$. This implies that the extended clustering will approximately satisfy the capacity constraints. The term $\alpha(S)$ is the expected distance from a new point to its nearest neighbor, and this can be used to bound the difference between $Q_S(g_S, C_S)$ and $Q(\bar{g}_S, C_s)$. Finally, β is used to bound the excess cost of outputting a cluster assignment constant on the Voronoi tiles of S. □

The terms $\alpha(S)$ and $\beta(S)$ can be bounded under natural conditions on the distribution μ. For example, when the distribution has doubling dimension d_0 and the optimal clustering of $\mathcal{X}$ is ϕ-probabilistically Lipschitz (Urner, Wulff, and Ben-David 2013) then for $n = \tilde{O}((\frac{1}{\epsilon\phi^{-1}(\epsilon)})^{d_0} d_0)$ we have $\alpha(S) < D^2\epsilon$ and $\beta(S) < pD^2\epsilon$ with high probability. See the full version of the paper for details.

Experiments

In this section, we present an empirical study of the accuracy and scalability of our technique using both the LP rounding algorithms and k-means++ together with the nearest neighbor extension. We compare against three baselines: random partitioning, balanced partition trees, and locality sensitive hashing (LSH) on both synthetic and real world image and advertising datasets. Our findings are summarized below:

- In the full version of the paper we show that for our datasets, both k-means++ and our LP rounding algorithms produce high-quality balanced clusterings. These results complement the results of Section , showing that k-means++ produces high-quality balanced clusterings for 'typical' data. Based on this, our further empirical studies use k-means++.
- We compare the accuracy of our technique (using k-means++ and the nearest neighbor extension) to the three

```
Input: Dataset $S = \{x_1, \dots, x_n\}$, cluster parameters $(k, p, \ell, L)$, second sample size $n'$.
1. Draw second sample $S'$ of size $n'$ iid from $\mu$.
2. For each point $x_i$, set $\hat{w}_i = |S'_i|/n'$, where $S'_i = \{x' \in S' : NN_S(x') = x_i\}$
3. Let $C_S = (c_1, \dots, c_k)$ and $g_S : S \to \binom{C}{p}$ be a clustering of $S$ obtained by minimizing
       $$Q_S(g, C) = \sum_{i=1}^{n} \hat{w}_i \sum_{c_j \in g(x)} d(x_i, c_j)^2$$
       subject to $\sum_{i : c_j \in g_n(x_i)} \hat{w}_i \in [\ell, L]$ for all $j = 1, \dots, k$.
4. Return $\bar{g}_S(x) = g_S(NN_S(x))$ and centers $C_S$.
```

Algorithm 2: Nearest neighbor clustering extension.

baselines for a wide range of values of k in large-scale learning tasks where each machine learns a local SVM classifier. For all values of k and all datasets, our algorithm achieves higher accuracy than all our baselines.

• We show that our framework exhibits strong scaling, meaning that if we double the available computing power, the total running time reduces by a constant fraction.

Experimental Setup: In each run of our experiment, one of the partitioning algorithms produces a dispatch rule from $10,000$ randomly sampled training points. This dispatch rule is then used to distribute the training data among the available worker machines. If the parameter k exceeds the number of machines, we allow each machine to process multiple partitions independently. Next we train a one-vs-all linear separator for each partition in parallel by minimizing the L2-regularized L2-loss SVM objective. The regularization parameter is chosen via 5-fold cross validation. To predict the label of a new example, we use the dispatch rule to send it to the machine with the most appropriate model. All results are averaged over 10 independent runs.

Details for our technique: We run our method with k-means++ and the nearest neighbor dispatch. We use the following heuristics to ensure balancedness: while any cluster is smaller than ℓn points, merge it with the cluster with nearest center. Next, randomly partition each cluster larger than Ln points into evenly sized subsets. This guarantees every cluster satisfies the capacity constraints, but the number of output clusters may differ from k. We use the random partition tree algorithm of Dasgupta and Sinha (2015) for efficient nearest neighbor search. We set $\ell = 1/(2k)$ and $L = 2/k$ and $p = 1$, since our baselines do not support replication.

Baselines: We compare against the following baselines:[1]

• *Random Partitioning:* Points are dispatched uniformly at random. This baseline produces balanced partitions but does not send similar examples to the same machine.

• *Balanced Partition Trees:* Similarly to a kd-tree, this partitioning rule recursively divides the dataset by splitting it at the median point along a randomly chosen dimension. This baseline produces balanced partitions and improves over random partitioning because each machine learns a local model for a different subset of the space.

• *LSH Partitioning:* Our LSH baseline chooses a random hash $h : \mathbb{R}^d \to \mathbb{Z}$ (the concatenation of 10 random projections followed by binning (Datar et al. 2004)) and assigns point x to cluster $h(x) \mod k$. This baseline sends similar examples to the same machine, but produces unbalanced partitions.

Datasets: We use the following datasets:

• *Synthetic:* A synthetic 128 GB dataset with 30 classes and 20 features. The data distribution is a mixture of 200 Gaussians with uniformly random centers in $[0, 1]^{20}$ with covariance $0.09I$. Labels are assigned so that nearby Gaussians have the same label.

• *MNIST-8M:* The MNIST-8M dataset (Loosli, Canu, and Bottou 2007), which has $8M$ examples and 784 features.

• *CIFAR-10:* The CIFAR-10 dataset (Krizhevsky 2009) is an image classification task with 10 classes. We include 50 randomly rotated and cropped copies of each training example to get 2.5 million training examples. We extract the features from the Google Inception network (Szegedy et al. 2015) by using the output of layers in3c and in4d.

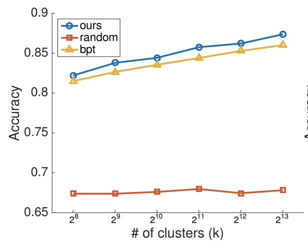
• *CTR:* The CTR dataset contains ad impressions from a commercial search engine where the label indicates whether the ad was clicked. It has 860K examples with 232 features.

Results: Our empirical results are shown in Figure 3. We do not report accuracies when the partitioning is imbalanced, specifically when the largest $k/2$ clusters contain more than 98% of the data. For all values of k and all datasets, our method has higher accuracy than all three baselines. For all datasets except CTR, the accuracy of our method increases as a function of k, until each cluster is data starved.

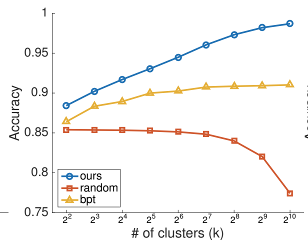
Our method combines the good aspects of both the balanced partition tree and LSH baselines by simultaneously sending similar examples to the same machines and ensuring that every machine gets roughly the same amount of data. In contrast, the balanced partition tree baseline both produce balanced clusters, but do not send similar examples to the same machines, and the LSH baseline sends similar examples to the same machine, but makes no attempt at balancing the partitions. The fact that we get higher accuracy than the LSH baseline demonstrates that it is not enough to send similar examples to the same machines without balancing, and that we get higher accuracy than balanced partition trees shows that simply balancing the cluster sizes is not sufficient.

Figure 3(f) shows the speedup obtained when running our system using 16, 32, or 64 workers compared to using 8. We clock the time taken for the entire experiment: the time for clustering a subsample, dispatch, training and testing. In all cases, doubling the number of workers reduces the total time by a constant factor, showing that our framework strongly scales and can be applied to very large datasets.

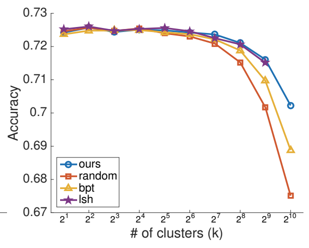
[1]Since our framework does not communicate during training, we do not compare against algorithms that do, e.g. boosting (Balcan et al. 2012).



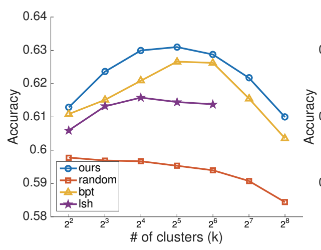
(a) Accuracy on Synthetic Dataset



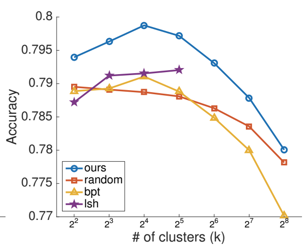
(b) Accuracy on MNIST-8M



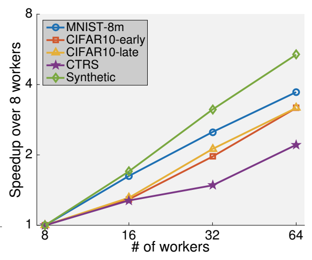
(c) Accuracy on CTR Dataset



(d) Accuracy on CIFAR-10 (in3c)



(e) Accuracy on CIFAR-10 (in4d)



(f) **Strong Scaling**

Figure 3: Figures (a) through (e) show the effect of k on the classification accuracy. Figure (f) shows the speedup factor as we increase the number of workers from 8 to 64 for each dataset.

Conclusion

In this work, we propose and analyze a new framework for distributed learning. Given that similar points tend to have similar classes, we partition the data so that similar examples go to the same machine. We cast the dispatching step as a clustering problem combined with novel fault tolerance and balance constraints necessary for distributed systems. We show the added constraints make the objective highly nontrivial, yet we provide LP rounding algorithms with provable guarantees. This is complemented by our results showing that the k-means++ algorithm is competitive on 'typical' datasets. These are the first algorithms with provable guarantees under both upper and lower capacity constraints, and may be of interest beyond distributed learning. We show that it is sufficient to cluster a small subsample of data and use a nearest neighbor extension technique to efficiently dispatch the remaining data. Finally, we conduct experiments for all our algorithms that show that our framework outperforms several baselines and strongly scales.

Acknowledgements

This work was supported in part by NSF grants CCF-1451177, CCF-1422910, CCF-1535967, IIS-1618714, IIS-1409802, a Sloan Research Fellowship, a Microsoft Research Faculty Fellowship, a Google Research Award, Intel Research, Microsoft Research, and a National Defense Science & Engineering Graduate (NDSEG) fellowship.

References

Agarwal, M.; Jaiswal, R.; and Pal, A. 2015. k-means++ under approximation stability. *Theoretical Computer Science* 588:37–51.

Ahmadian, S., and Swamy, C. 2016. Approximation algorithms for clustering problems with lower bounds and outliers. In *Proceedings of the 43rd annual International Colloquium on Automata, Languages, and Programming*.

Balcan, M.-F.; Blum, A.; Fine, S.; and Mansour, Y. 2012. Distributed learning, communication complexity and privacy. *arXiv preprint arXiv:1204.3514*.

Balcan, M.-F.; Blum, A.; and Gupta, A. 2013. Clustering under approximation stability. *J. ACM* 60(2):8:1–8:34.

Balcan, M.-F.; Haghtalab, N.; and White, C. 2016. k-center clustering under perturbation resilience. In *Proceedings of the 43rd annual International Colloquium on Automata, Languages, and Programming*.

Byrka, J.; Fleszar, K.; Rybicki, B.; and Spoerhase, J. 2015. Bi-factor approximation algorithms for hard capacitated k-median problems. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, 722–736. SIAM.

Cooper, B. F.; Ramakrishnan, R.; Srivastava, U.; Silberstein, A.; Bohannon, P.; Jacobsen, H.-A.; Puz, N.; Weaver, D.; and Yerneni, R. 2008. Pnuts: Yahoo!'s hosted data serving platform. *Proceedings of the VLDB Endowment* 1(2):1277–1288.

Dasgupta, S., and Sinha, K. 2015. Randomized partition trees for exact nearest neighbor search. *Algorithmica* 72(1):237–263.

Datar, M.; Immorlica, N.; Indyk, P.; and Mirrokni, V. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, 253–262.

Dick, T.; Li, M.; Pillutla, V. K.; White, C.; Balcan, M.; and

Smola, A. J. 2015. Data driven resource allocation for distributed learning. *CoRR* abs/1512.04848.

Gupta, R.; Roughgarden, T.; and Seshadhri, C. 2014. Decompositions of triangle-dense graphs. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, 471–482. ACM.

Jain, K.; Mahdian, M.; Markakis, E.; Saberi, A.; and Vazirani, V. V. 2003. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *Journal of the ACM (JACM)* 50(6):795–824.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images. Technical report, University of Toronto.

Li, M.; Andersen, D. G.; Smola, A. J.; and Yu, K. 2014. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, 19–27.

Li, S. 2014a. An improved approximation algorithm for the hard uniform capacitated k-median problem. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, 325–338.

Li, S. 2014b. Approximating capacitated k-median with $(1+\epsilon)k$ open facilities. *arXiv preprint arXiv:1411.5630*.

Loosli, G.; Canu, S.; and Bottou, L. 2007. Training invariant support vector machines using selective sampling. *Large scale kernel machines* 301–320.

Ostrovsky, R.; Rabani, Y.; Schulman, L. J.; and Swamy, C. 2006. The effectiveness of lloyd-type methods for the k-means problem. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, 165–176. IEEE.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Urner, R.; Wulff, S.; and Ben-David, S. 2013. Plal: Cluster-based active learning. In *Conference on Learning Theory*, 376–397.

Vapnik, V. N., and Bottou, L. 1993. Local algorithms for pattern recognition and dependencies estimation. *Neural Computation*.

Wei, K.; Iyer, R. K.; Wang, S.; Bai, W.; and Bilmes, J. A. 2015. Mixed robust/average submodular partitioning: Fast algorithms, guarantees, and applications. In *Advances in Neural Information Processing Systems*, 2233–2241.

You, Y.; Demmel, J.; Czechowski, K.; Song, L.; and Vuduc, R. 2015. CA-SVM: Communication-avoiding support vector machines on clusters. In *IEEE International Parallel and Distributed Processing Symposium*.

Zhang, Y.; Duchi, J.; Jordan, M.; and Wainwright, M. 2013. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *Neural Information Processing Systems*.

Zhang, Y.; Duchi, J. C.; and Wainwright, M. 2012. Communication-efficient algorithms for statistical optimization. In *Neural Information Processing Systems*.