Coresets for Data-efficient Training of Machine Learning Models

Baharan Mirzasoleiman [1] **Jeff Bilmes** [2] **Jure Leskovec** [3]

Abstract

Incremental gradient (IG) methods, such as stochastic gradient descent and its variants are commonly used for large scale optimization in machine learning. Despite the sustained effort to make IG methods more data-efficient, it remains an open question how to select a training data subset that can theoretically and practically perform on par with the full dataset. Here we develop CRAIG, a method to select a weighted subset (or coreset) of training data that closely estimates the full gradient by maximizing a submodular function. We prove that applying IG to this subset is guaranteed to converge to the (near)optimal solution with the same convergence rate as that of IG for convex optimization. As a result, CRAIG achieves a speedup that is inversely proportional to the size of the subset. To our knowledge, this is the *first* rigorous method for data-efficient training of *general* machine learning models. Our extensive set of experiments show that CRAIG, while achieving practically the same solution, speeds up various IG methods by up to 6x for logistic regression and 3x for training deep neural networks.

1. Introduction

Mathematical optimization lies at the core of training large-scale machine learning systems, and is now widely used over massive data sets with great practical success, assuming sufficient data resources are available. Achieving this success, however, also requires large amounts of (often GPU) computing, as well as concomitant financial expenditures and energy usage (Strubell et al., 2019). Significantly decreasing these costs without decreasing the learnt system's resulting accuracy is one of the grand challenges of machine learning and artificial intelligence today (Asi & Duchi, 2019).

Training machine learning models often reduces to optimizing a regularized empirical risk function. Given a convex loss l, and a μ-strongly convex regularizer r, one aims to find model parameter vector w_* over the parameter space $\mathcal{W}$ that minimizes the loss f over the training data V:

$$w_* \in \arg\min_{w\in\mathcal{W}} f(w), \quad f(w) := \sum_{i\in V} f_i(w) + r(w), \\ f_i(w) = l(w, (x_i, y_i)), \tag{1}$$

where $V = \{1, \ldots, n\}$ is an index set of the training data, and functions $f_i : \mathbb{R}^d \to \mathbb{R}$ are associated with training examples (x_i, y_i), where $x_i \in \mathbb{R}^d$ is the feature vector, and y_i is the point i's label.

Standard Gradient Descent can find the minimizer of this problem, but requires repeated computations of the full gradient $\nabla f(w)$—sum of the gradients over all training data points/functions i—and is therefore prohibitive for massive data sets. This issue is further exacerbated in case of deep neural networks where gradient computations (backpropagation) are expensive. Incremental Gradient (IG) methods, such as Stochastic Gradient Descent (SGD) and its accelerated variants, including SGD with momentum (Qian, 1999), Adagrad (Duchi et al., 2011), Adam (Kingma & Ba, 2014), SAGA (Defazio et al., 2014), and SVRG (Johnson & Zhang, 2013) iteratively estimate the gradient on random subsets/-batches of training data. While this provides an unbiased estimate of the full gradient, the randomized batches introduce variance in the gradient estimate (Hofmann et al., 2015), and therefore stochastic gradient methods are in general slow to converge (Johnson & Zhang, 2013; Defazio et al., 2014). The majority of the work speeding up IG methods has thus primarily focused on reducing the variance of the gradient estimate (SAGA (Defazio et al., 2014), SVRG (Johnson & Zhang, 2013), Katysha (Allen-Zhu, 2017)) or more carefully selecting the gradient stepsize (Adagrad (Duchi et al., 2011), Adadelta (Zeiler, 2012), Adam (Kingma & Ba, 2014)).

However, the direction that remains largely unexplored is how to carefully select a small subset $S \subseteq V$ of the full training data V, so that the model is trained only on the subset S while still (approximately) converging to the globally optimal solution (i.e., the model parameters that would be obtained if training/optimizing on the full V). If such

[1]Department of Computer Science, University of California, Los Angeles, USA [2]Department of Electrical Engineering, University of Washington, Seattle, USA [3]Department of Computer Science, Stanford University, Stanford, USA. Correspondence to: Baharan Mirzasoleiman <baharan@cs.ucla.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

a subset S can be quickly found, then this would directly lead to a speedup of $|V|/|S|$ (which can be very large if $|S| \ll |V|$) per epoch of IG.

There are four main challenges in finding such a subset S. First is that a guiding principle for selecting S is unclear. For example, selecting training points close to the decision boundary might allow the model to fine tune the decision boundary, while picking the most diverse set of data points would allow the model to get a better sense of the training data distribution. Second is that finding S must be fast, as otherwise identifying the set S may take longer than the actual optimization, and so no overall speed-up would be achieved. Third is that finding a subset S is not enough. One also has to decide on a gradient stepsize for each data point in S, as they affect the convergence. And last, while the method might work well empirically on some data sets, one also requires theoretical understanding and mathematical convergence guarantees.

Here we develop *Coresets for Accelerating Incremental Gradient descent (*CRAIG*)*, for selecting a subset of training data points to speed up training of large machine learning models. Our key idea is to select a weighted subset S of training data V that best approximates the full gradient of V. We prove that the subset S that minimizes an upper-bound on the error of estimating the full gradient maximizes a submodular facility location function. Hence, S can be efficiently found using a fast greedy algorithm.

We also provide theoretical analysis of CRAIG and prove its convergence. Most importantly, we show that any incremental gradient method (IG) on S converges in the same number epochs as the same IG would on the full V, which means that we obtain a speed-up inversely proportional to the size of S. In particular, for a μ-strongly convex risk function and a subset S selected by CRAIG that estimates the full gradient by an error of at most ϵ, we prove that IG on S with diminishing stepsize $\alpha_k = \alpha/k^\tau$ at epoch k (with $0 < \tau < 1$ and $0 < \alpha$), converges to an $2R\epsilon/\mu^2$ neighborhood of the optimal solution at rate $\mathcal{O}(1/\sqrt{k})$. Here, $R = \min\{d_0, (r\gamma_{\max}C + \epsilon)/\mu\}$ where d_0 is the initial distance to the optimum, C is an upper-bound on the norm of the gradients, $r = |S|$, and $\gamma_{\max}$ is the largest weight for the elements in the subset obtained by CRAIG. Moreover, we prove that if in addition to the strong convexity, component functions have smooth gradients, IG with the same diminishing step size on subset S converges to a $2\epsilon/\mu$ neighborhood of the optimum solution at rate $\mathcal{O}(1/k^\tau)$.

The above implies that IG on S converges to the same solution and in the same number of epochs as IG on the full V. But because every epoch only uses a subset S of the data, it requires fewer gradient computations and thus leads to a $|V|/|S|$ speedup over traditional IG methods, while still (approximately) converging to the optimal solution. We also note that CRAIG is complementary to various incremental gradient (IG) methods (SGD, SAGA, SVRG, Adam), and such methods can be used on the subset S found by CRAIG.

We also demonstrate the effectiveness of CRAIG via an extensive set of experiments using logistic regression (a convex optimization problem) as well as training deep neural networks (non-convex optimization problems). We show that CRAIG speeds up incremental gradient methods, including SGD, SAGA, and SVRG. In particular, CRAIG while achieving practically the same loss and accuracy as the underlying incremental gradient descent methods, speeds up gradient methods by up to 6x for convex and 3x for non-convex loss functions.

2. Related Work

Convergence of IG methods has been long studied under various conditions (Zhi-Quan & Paul, 1994; Mangasarian & Solodovy, 1994; Bertsekas, 1996; Solodov, 1998; Tseng, 1998), however IG's convergence rate has been characterized only more recently (see (Bertsekas, 2015b) for a survey). In particular, (Nedić & Bertsekas, 2001) provides a $\mathcal{O}(1/\sqrt{k})$ convergence rate for diminishing stepsizes α_k per epoch k under a strong convexity assumption, and (Gürbüzbalaban et al., 2015) proves a $\mathcal{O}(1/k^\tau)$ convergence rate with diminishing stepsizes $\alpha_k = \Theta(1/k^\tau)$ for $\tau \in (0, 1]$ under an additional smoothness assumption for the components. While these works provide convergence on the full dataset, our analysis provides the same convergence rates on subsets obtained by CRAIG.

Techniques for speeding up SGD, are mostly focused on variance reduction techniques (Roux et al., 2012; Shalev-Shwartz & Zhang, 2013; Johnson & Zhang, 2013; Hofmann et al., 2015; Allen-Zhu et al., 2016), and accelerated gradient methods when the regularization parameter is small (Frostig et al., 2015; Lin et al., 2015; Xiao & Zhang, 2014). Very recently, (Hofmann et al., 2015; Allen-Zhu et al., 2016) exploited neighborhood structure to further reduce the variance of stochastic gradient descent and improve its running time. Our CRAIG method and analysis are complementary to variance reduction and accelerated methods. CRAIG can be applied to all these methods as well to speed them up.

Coresets are weighted subsets of the data, which guarantee that models fitting the coreset also provide a good fit for the original data. Coreset construction methods traditionally perform importance sampling with respect to sensitivity score, to provide high-probability solutions (Har-Peled & Mazumdar, 2004; Lucic et al., 2017; Cohen et al., 2017) for a particular problem, such as k-means and k-median clustering (Har-Peled & Mazumdar, 2004), naïve Bayes and nearest-neighbors (Wei et al., 2015), mixture mod-

els (Lucic et al., 2017), low rank approximation (Cohen et al., 2017), spectral approximation (Agarwal et al., 2004; Li et al., 2013), Nystrom methods (Agarwal et al., 2004; Musco & Musco, 2017), and Bayesian inference (Campbell & Broderick, 2018). Unlike existing coreset construction algorithms, our method is not problem specific and can be applied for training general machine learning models.

3. Coresets for Accelerating Incremental Gradient Descent (CRAIG)

We proceed as follows: First, we define an objective function L for selecting an optimal set S of size r that best approximates the gradient of the full training dataset V of size n. Then, we show that L can be turned into a submodular function F and thus S can be efficiently found using a fast greedy algorithm. Crucially, we also show that for convex loss functions the approximation error between the estimated and the true gradient can be efficiently minimized in a way that is independent of the actual optimization procedure. Thus, CRAIG can simply be used as a preprocessing step before the actual optimization starts.

Incremental gradient methods aim at estimating the full gradient $\nabla f(w)$ over V by iteratively making a step based on the gradient of every function f_i. Our key idea in CRAIG is that if we can find a small subset S such that the weighted sum of the gradients of its elements closely approximates the full gradient over V, we can apply IG only to the set S (with stepsizes equal to the weight of the elements in S), and we should still converge to the (approximately) optimal solution, but much faster.

Specifically, our goal in CRAIG is to find the smallest subset $S \subseteq V$ and corresponding per-element stepsizes $\gamma_j > 0$ that approximate the full gradient with an error at most $\epsilon > 0$ for all the possible values of the optimization parameters $w \in \mathcal{W}$.[1]

$$S^* = \arg\min_{S\subseteq V, \gamma_j \geq 0\ \forall j} |S|, \quad \text{s.t.} \quad \max_{w\in\mathcal{W}} \|\sum_{i\in V} \nabla f_i(w) - \sum_{j\in S} \gamma_j \nabla f_j(w)\| \leq \epsilon. \tag{2}$$

Given such an S^* and associated weights $\{\gamma\}_j$, we are guaranteed that gradient updates on S^* will be similar to the gradient updates on V regardless of the value of w.

Unfortunately, directly solving the above optimization problem is not feasible, due to two problems. Problem 1: Eq. (2) requires us to calculate the gradient of all the functions f_i over the entire space $\mathcal{W}$, which is too expensive and would not lead to overall speedup. In other words, it would appear that solving for S^* is as difficult as solving Eq. (1), as it involves calculating $\sum_{i\in V} \nabla f_i(w)$ for various $w \in \mathcal{W}$. And Problem 2: even if calculating the normed difference between the gradients in Eq. (2) would be fast, as we discuss later finding the optimal subset S^* in NP-hard. In the following, we address the above two challenges and discuss how we can quickly find a near-optimal subset S.

[1] Note that in the worst case we may need $|S^*| \approx |V|$ to approximate the gradient. However, as we show in experiments, in practice we find that a small subset is sufficient to accurately approximate the gradient.

3.1. Upper-bound on the Estimation Error

We first address Problem 1, i.e., how to quickly estimate the error/discrepancy of the weighted sum of gradients of functions f_j associate with data points $j \in S$, vs the full gradient, for every $w \in \mathcal{W}$.

Let S be a subset of r data points. Furthermore, assume that there is a mapping $\varsigma_w : V \to S$ that for every $w \in \mathcal{W}$ assigns every data point $i \in V$ to one of the elements j in S, i.e., $\varsigma_w(i) = j \in S$. Let $C_j = \{i \in [n] | \varsigma(i) = j\} \subseteq V$ be the set of data points that are assigned to $j \in S$, and $\gamma_j = |C_j|$ be the number of such data points. Hence, $\{C_j\}_{j=1}^r$ form a partition of V. Then, for any arbitrary (single) $w \in \mathcal{W}$ we can write

$$\sum_{i\in V} \nabla f_i(w) = \sum_{i\in V} \big(\nabla f_i(w) - \nabla f_{\varsigma_w(i)}(w) + \nabla f_{\varsigma_w(i)}(w)\big) \tag{3}$$

$$= \sum_{i\in V} \big(\nabla f_i(w) - \nabla f_{\varsigma_w(i)}(w)\big) + \sum_{j\in S} \gamma_j \nabla f_j(w). \tag{4}$$

Subtracting and then taking the norm of the both sides, we get an upper bound on the error of estimating the full gradient with the weighted sum of the gradients of the functions f_j for $j \in S$. I.e.,

$$\|\sum_{i\in V} \nabla f_i(w) - \sum_{j\in S} \gamma_j \nabla f_j(w)\| \leq \sum_{i\in V} \|\nabla f_i(w) - \nabla f_{\varsigma_w(i)}(w)\|, \tag{5}$$

where the inequality follows from the triangle inequality. The upper-bound in Eq. (5) is minimized when ς_w assigns every $i \in V$ to an element in S with most gradient similarity at w, or minimum Euclidean distance between the gradient vectors at w. That is: $\varsigma_w(i) \in \arg\min_{j\in S} \|\nabla f_i(w) - \nabla f_j(w)\|$. Hence,

$$\min_{S\subseteq V} \|\sum_{i\in V} \nabla f_i(w) - \sum_{j\in S} \gamma_j \nabla f_j(w)\| \leq \sum_{i\in V} \min_{j\in S} \|\nabla f_i(w) - \nabla f_j(w)\|. \tag{6}$$

The right hand side of Eq. (6) is minimized when S is the set of r *medoids* (exemplars) (Kaufman et al., 1987) for all the components in the gradient space.

So far, we considered upper-bounding the gradient estimation error at a particular $w \in \mathcal{W}$. To bound the estimation error for all $w \in \mathcal{W}$, we consider a worst-case approximation of the estimation error over the entire parameter space $\mathcal{W}$. Formally, we define a distance metric d_{ij} between gradients of f_i and f_j as the maximum normed difference between $\nabla f_i(w)$ and $\nabla f_j(w)$ over all $w \in \mathcal{W}$:

$$d_{ij} \triangleq \max_{w \in \mathcal{W}} \|\nabla f_i(w) - \nabla f_j(w)\|. \tag{7}$$

Thus, by solving the following minimization problem, we obtain the smallest weighted subset S^* that approximates the full gradient by an error of at most ϵ for all $w \in \mathcal{W}$:

$$S^* = \arg\min_{S \subseteq V} |S|, \quad \text{s.t.} \quad L(S) \triangleq \sum_{i \in V} \min_{j \in S} d_{ij} \leq \epsilon. \tag{8}$$

Note that Eq. (8) requires that the gradient error is bounded over $\mathcal{W}$. However, we show (Appendix B.1) for several classes of convex problems, including linear regression, ridge regression, logistic regression, and regularized support vector machines (SVMs), the normed gradient difference between data points can be efficiently boundedly approximated by (Allen-Zhu et al., 2016; Hofmann et al., 2015):

$$\forall w, i, j \quad \|\nabla f_i(w) - \nabla f_j(w)\| \leq d_{ij} \leq \max_{w \in \mathcal{W}} \mathcal{O}(\|w\|) \cdot \|x_i - x_j\| = \text{const.} \|x_i - x_j\|. \tag{9}$$

Note when $\|w\|$ is bounded for all $w \in \mathcal{W}$, i.e., $\max_{w \in \mathcal{W}} \mathcal{O}(\|w\|) < \infty$, upper-bounds on the Euclidean distances between the gradients can be pre-computed. This is crucial, because it means that estimation error of the full gradient can be efficiently bounded independent of the actual optimization problem (i.e., point w). Thus, these upper-bounds can be computed only once as a pre-processing step before any training takes place, and then used to find the subset S by solving the optimization problem (8). We address upper-bounding the normed difference between gradients for deep models in Section 3.4.

3.2. The CRAIG Algorithm

Optimization problem (8) produces a subset S of elements with their associated weights $\{\gamma\}_{j \in S}$ or per-element stepsizes that closely approximates the full gradient. Here, we show how to efficiently approximately solve the above optimization problem to find a near-optimal subset S.

The optimization problem (8) is NP-hard as it involves calculating the value of $L(S)$ for all the $2^{|V|}$ subsets $S \subseteq V$. We show, however, that we can transform it into a *submodular set cover problem*, that can be efficiently approximated.

Formally, F is submodular if $F(S \cup \{e\}) - f(S) \geq F(T \cup \{e\}) - F(T)$, for any $S \subseteq T \subseteq V$ and $e \in V \setminus T$. We denote the *marginal* utility of an element s w.r.t. a subset S as $F(e|S) = F(S \cup \{e\}) - F(S)$. Function F is called *monotone* if $F(e|S) \geq 0$ for any $e \in V \setminus S$ and $S \subseteq V$. The submodular cover problem is defined as finding the smallest set S that achieves utility ρ. Precisely,

$$S^* = \arg\min_{S \subseteq V} |S|, \quad \text{s. t.} \quad F(S) \geq \rho. \tag{10}$$

Although finding S^* is NP-hard since it captures such well-known NP-hard problems such as Minimum Vertex Cover, for many classes of submodular functions, a simple greedy algorithm is known to be very effective (Nemhauser et al., 1978; Wolsey, 1982). The greedy algorithm starts with the empty set $S_0 = \emptyset$, and at each iteration i, it chooses an element $e \in V$ that maximizes $F(e|S_{i-1})$, i.e., $S_i = S_{i-1} \cup \{\arg\max_{e \in V} F(e|S_{i-1})\}$. Greedy gives us a logarithmic approximation, i.e. $|S| \leq \big(1 + \ln(\max_e F(e|\emptyset))\big)|S^*|$ (Wolsey, 1982). The computational complexity of the greedy algorithm is $\mathcal{O}(|V| \cdot |S|)$. However, its running time can be reduced to $\mathcal{O}(|V|)$ using stochastic algorithms (Mirzasoleiman et al., 2015a) and further improved using lazy evaluation (Minoux, 1978), and distributed implementations (Mirzasoleiman et al., 2015b; 2016). Given a subset $S \subseteq V$, the facility location function quantifies the coverage of the whole data set V by the subset S by summing the similarities between every $i \in V$ and its closest element $j \in S$. Formally, facility location is defined as $F_{fl}(S) = \sum_{i \in V} \max_{j \in S} s_{i,j}$, where $s_{i,j}$ is the similarity between $i, j \in V$. The facility location function has been used in various summarization applications (Lin et al., 2009; Lin & Bilmes, 2012). By introducing an auxiliary element s_0 we can turn $L(S)$ in Eq. (8) into a monotone submodular facility location function,

$$F(S) = L(\{s_0\}) - L(S \cup \{s_0\}), \tag{11}$$

where $L(\{s_0\})$ is a constant. In words, F measures the decrease in the estimation error associated with the set S versus the estimation error associated with just the auxiliary element. For a suitable choice of s_0, maximizing F is equivalent to minimizing L. Therefore, we apply the greedy algorithm to approximately solve the following problem to get the subset S defined in Eq. (8):

$$S^* = \arg\min_{S \subseteq V} |S|, \quad \text{s.t.} \quad F(S) \geq L(\{s_0\}) - \epsilon. \tag{12}$$

At every step, the greedy algorithm selects an element that reduces the upper bound on the estimation error the most. In fact, the size of the smallest subset S that estimates the full gradient by an error of at most ϵ depends on the structural properties of the data. Intuitively, as long as the marginal gains of facility location are considerably large, we need more elements to improve our estimation of the full gradient. Having found S, the weight γ_j of every element $j \in S$ is the number of components that are closest to it in the gradient space, and are used as stepsize of element $j \in S$ during IG. The pseudocode for CRAIG is outlined in Algorithm 1.

Algorithm 1 CRAIG (CoResets for Accelerating Incremental Gradient descent)

Input: Set of component functions f_i for $i \in V = [n]\}$.
Output: Subset $S \subseteq V$ with corresponding per-element stepsizes $\{\gamma\}_{j \in S}$.

$S_0 \leftarrow \emptyset, s_0 = 0, i = 0$.
while $F(S) < L(\{s_0\}) - \epsilon$ **do**
  $j \in \arg\max_{e \in V \setminus S_{i-1}} F(e|S_{i-1})$
  $S_i = S_{i-1} \cup \{j\}$
  $i = i + 1$
end while
for $j = 1$ to $|S|$ **do**
  $\gamma_j = \sum_{i \in V} \mathbb{I}\big[j = \arg\min_{s \in S} \max_{w \in \mathcal{W}} \|\nabla f_i(w) - \nabla f_s(w)\|\big]$
end for

Notice that CRAIG creates subset S incrementally one element at a time, which produces a natural order to the elements in S. Adding the element with largest marginal gain $j \in \arg\max_{e \in V} F(e|S_{i-1})$ improves our estimation from the full gradient by an amount bounded by the marginal gain. At every step i, we have $F(S_i) \geq (1 - e^{-i/|S|}) F(S^*)$. Hence, for a greedily ordered subset $S = \{s_1, \cdots, s_k\}$, we have

$$\|\sum_{i \in V} \nabla f_i(w) - \sum_{j=1}^{k} \gamma_{s_j} \nabla f_{s_j}(w)\| \leq c - (1 - e^{-j/k}) L(S^*), \tag{13}$$

where c is a constant. Intuitively, the first elements of the ordering contribute the most to provide a close approximation of the full gradient and the rest of the elements further refine the approximation. Hence, the first incremental gradient updates gets us close to w_*, and the rest of the updates further refine the solution.

3.3. CRAIG with Limited Budget

In practice, we often have a limited budget in terms of time or computational resources, and we are interested to find a near-optimal subset of size r that best approximates the full gradient. This problem can be formulated as a submodular maximization problem which is dual to the submodular cover problem (12):

$$S^* \in \arg\max_{S \subseteq V} F(S), \quad \text{s.t.} \quad |S| \leq r. \tag{14}$$

For the above submodular maximization problem, the greedy algorithm discussed in Section 3.2 provides a $(1 - 1/e)$-approximation to the optimal solution. For a subset S of size at most r obtained by the greedy algorithm, we can calculate the value of ϵ as follows:

$$\epsilon \leq F(S) - L(\{s_0\}). \tag{15}$$

We use this formulation in our experiments in Section 5.

3.4. Application of CRAIG to Deep Networks

As discussed, CRAIG selects a subset that closely approximates the full gradient, and hence can be also applied for speeding up training deep networks. The challenge here is that we cannot use inequality (9) to bound the normed difference between gradients for all $w \in \mathcal{W}$ and find the subset as a preprocessing step.

However, for deep neural networks, the variation of the gradient norms is mostly captured by the gradient of the loss w.r.t. the input to the last layer [Section 3.2 of (Katharopoulos & Fleuret, 2018)]. We show (Appendix B.1) that the normed gradient difference between data points can be efficiently bounded approximately by

$$\|\nabla f_i(w) - \nabla f_j(w)\| \leq \tag{16}$$
$$c_1 \|\Sigma'_L(z_i^{(L)}) \nabla f_i^{(L)}(w) - \Sigma'_L(z_j^{(L)}) \nabla f_j^{(L)}(w)\| + c_2,$$

where $\Sigma'_L(z_i^{(L)}) \nabla f_i^{(L)}(w)$ is gradient of the loss w.r.t. the input to the last layer for data point i, and c_1, c_2 are constants. The above upper-bound depends on parameter vector w which changes during the training process. Thus, we need to use CRAIG to update the subset S after a number of parameter updates.

The above upper-bound is often only slightly more expensive than calculating the loss. For example, in cases where we have cross entropy loss with soft-max as the last layer, the gradient of the loss w.r.t. the i-th input to the soft-max is simply $p_i - y_i$, where p_i are logits (dimension $p-1$ for p classes) and y is the one-hot encoded label. In this case, CRAIG does not need any backward pass or extra storage. Note that, although CRAIG needs an additional $\mathcal{O}(|V| \cdot |S|)$ complexity (or $\mathcal{O}(|V|)$ using stochastic greedy) to find the subset S at the beginning of every epoch, this complexity does not involve any (exact) gradient calculations and is negligible compared to the cost of backpropagations performed during the epoch. Hence, as we show in the experiments CRAIG is practical and scalable.

4. Convergence Rate Analysis of CRAIG

The idea of CRAIG is to selects a subset that closely approximates the full gradient, and hence can be applied to speed up most IG variants as we show in our experiments. Here, we briefly introduce the original IG method, and then prove the convergence rate of IG applied to CRAIG subsets.

4.1. Incremental Gradient Methods (IG)

Incremental gradient (IG) methods are core algorithms for solving Problem (1) and are widely used and studied. IG aims at approximating the standard gradient method by sequentially stepping along the gradient of the component functions f_i in a cyclic order. Starting from an initial point

$w_0^1 \in \mathbb{R}^d$, it makes k passes over all the n components. At every epoch $k \geq 1$, it iteratively updates w_{i-1}^k based on the gradient of f_i for $i = 1, \cdots, n$ using stepsize $\alpha_k > 0$. I.e.,

$$w_i^k = w_{i-1}^k - \alpha_k \nabla f_i(w_{i-1}^k), \qquad i = 1, 2, \cdots, n, \quad (17)$$

with the convention that $w_0^{k+1} = w_n^k$. Note that for a closed and convex subset $\mathcal{W}$ of $\mathbb{R}^d$, the results can be projected onto $\mathcal{W}$, and the update rule becomes

$$w_i^k = P_{\mathcal{W}}(w_{i-1}^k - \alpha_k \nabla f_i(w_{i-1}^k)), \qquad i = 1, 2, \cdots, n, \quad (18)$$

where $P_{\mathcal{W}}$ denotes projection on the set $\mathcal{W} \subset \mathbb{R}^d$.

IG with diminishing stepsizes converges at rate $\mathcal{O}(1/\sqrt{k})$ for strongly convex sum function (Nedić & Bertsekas, 2001). If in addition to the strong convexity of the sum function, every component function f_i is smooth, IG with diminishing stepsizes $\alpha_k = \Theta(1/k^s), s \in (0, 1]$ converges at rate $\mathcal{O}(1/k^s)$ (Gürbüzbalaban et al., 2015).

The convergence rate analysis of IG is valid regardless of order of processing the elements. However, in practice, the convergence rate of IG is known to be quite sensitive to the order of processing the functions (Bertsekas, 2015a; Gurbuzbalaban et al., 2017). If problem-specific knowledge can be used to find a favorable order σ (defined as a permutation $\{\sigma_1, \cdots, \sigma_n\}$ of $\{1, 2, ..., n\}$), IG can be updated to process the functions according to this order, i.e.,

$$w_i^k = w_{i-1}^k - \alpha_k \nabla f_{\sigma_i}(w_{i-1}^k), \qquad i = 1, 2, \cdots, n. \quad (19)$$

In general a favorable order is not known in advance. A common approach is sampling the function indices with replacement from the set $\{1, 2, \cdots, n\}$, which is called the Stochastic Gradient Descent (SGD) method.

4.2. Convergence Rate of IG on CRAIG Subsets

Next we analyze the convergence rate of IG applied to the weighted subset S found by CRAIG. Note that CRAIG finds S by greedily minimizing (12) (or maximizing (14)). Therefore, S is a near-optimal solution of problem (2) and estimates the full gradient by an error of at most ϵ, i.e., $\max_{w \in \mathcal{W}} \| \sum_{i \in V} \nabla f_i(w) - \sum_{j \in S} \gamma_j \nabla f_j(w) \| \leq \epsilon$.

Here, we show that (1) applying IG to S converges to a close neighborhood of the optimal solution and that (2) this convergence happens at the same rate (same number of epochs) as IG on the full data. Formally, every step of IG on the subset becomes

$$w_i^k = w_{i-1}^k - \alpha_k \gamma_{s_{\sigma_i}} \nabla f_{s_{\sigma_i}}(w_{i-1}^k), \quad i = 1, 2, \cdots, r, \quad s_i \in S, \quad |S| = r. \quad (20)$$

Here, σ is a permutation of $\{1, 2, \cdots, r\}$, and the per-element stepsize γ_{s_i} for every function f_{s_i} is the weight of the element $s_i \in S$ and is fixed for all epochs.

4.3. Convergence for Strongly Convex f

We first provide the convergence analysis for the case where the function f in Problem (1) is strongly convex, i.e. $\forall w, w' \in \mathbb{R}^d$ we have $f(w) \geq f(w') + \langle \nabla f(w'), w - w' \rangle + \frac{\mu}{2} \|w' - w\|^2$.

Theorem 1. *Assume that f is strongly convex, and S is a weighted subset of size r obtained by* CRAIG *that estimates the full gradient by an error of at most ϵ, i.e., $\max_{w \in \mathcal{W}} \| \sum_{i \in V} \nabla f_i(w) - \sum_{j \in S} \gamma_j \nabla f_j(w) \| \leq \epsilon$. Then for the iterates $\{w_k = w_0^k\}$ generated by applying IG to S with per-epoch stepsize $\alpha_k = \alpha / k^\tau$ with $\alpha > 0$ and $\tau \in [0, 1]$, we have*

(i) if $\tau = 1$, then $\|w_k - w_*\|^2 \leq 2\epsilon R/\mu^2 + \alpha r^2 \gamma_{\max}^2 C^2 / k\mu$,

(ii) if $0 < \tau < 1$, then $\|w_k - w_*\|^2 \leq 2\epsilon R/\mu^2$, for $k \to \infty$

(iii) if $\tau = 0$, then $\|w_k - w_*\|^2 \leq (1 - \alpha\mu)^{k+1} \|w_0 - w_*\|^2 + 2\epsilon R/\mu^2 + \alpha r^2 \gamma_{\max}^2 C^2 / \mu$,

where C is an upper-bound on the norm of the component function gradients, i.e. $\max_{i \in V} \sup_{w \in \mathcal{W}} \|\nabla f_i(w)\| \leq C$, $\gamma_{\max} = \max_{j \in S} \gamma_j$ is the largest per-element step size, and $R = \min\{d_0, (r\gamma_{\max} C + \epsilon)/\mu\}$, where $d_0 = \|w_0 - w_*\|$ is the initial distance to the optimum w_*.

All the proofs can be found in the Appendix. The above theorem shows that IG on S converges at the same rate $\mathcal{O}(1/\sqrt{k})$ of IG on the entire data set V. However, compared to IG on V, the $|V|/|S|$ speedup of IG on S comes at the price of getting an extra error term, $2\epsilon R/\mu^2$.

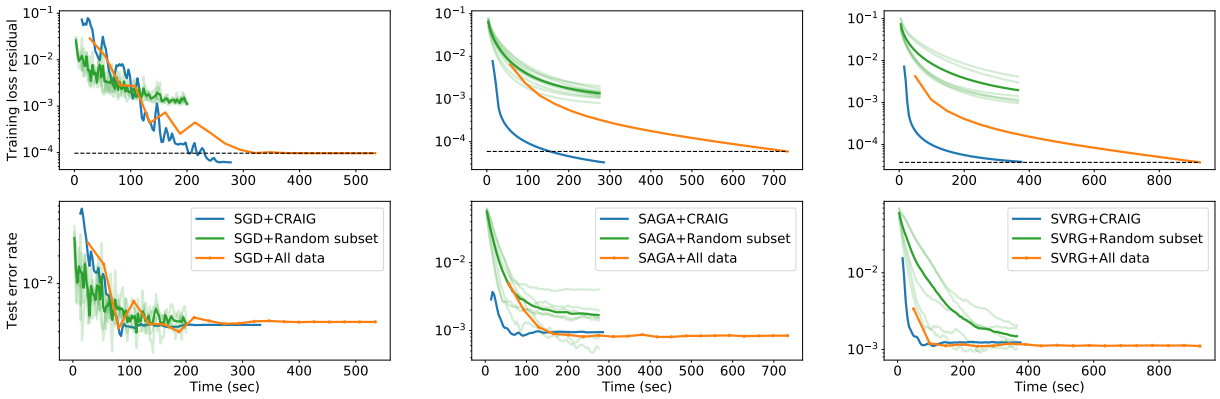
4.4. Convergence for Smooth and Strongly Convex f

If in addition to strong convexity of the expected risk, each component function has a Lipschitz gradient, i.e. $\forall w \in \mathcal{W}, i \in [n]$ we have $\|\nabla f_i(w) - \nabla f_i(w')\| \leq \beta_i \|w - w'\|$, then we get the following results about the iterates generated by applying IG to the weighted subset S returned by CRAIG.

Theorem 2. *Assume that f is strongly convex and let $f_i(w), i = 1, 2, \cdots, n$ be convex and twice continuously differentiable component functions with Lipschitz gradients on $\mathcal{W}$. Supposed that S is a weighted subset of size r obtained by* CRAIG *that estimates the full gradient by an error of at most ϵ, i.e., $\max_{w \in \mathcal{W}} \| \sum_{i \in V} \nabla f_i(w) - \sum_{j \in S} \gamma_j \nabla f_j(w) \| \leq \epsilon$. Then for the iterates $\{w_k = w_0^k\}$ generated by applying IG to S with per-epoch stepsize $\alpha_k = \alpha / k^\tau$ with $\alpha > 0$ and $\tau \in [0, 1]$, we have*

(i) if $\tau = 1$, then $\|w_k - w_*\| \leq 2\epsilon/\mu + \alpha\beta C r \gamma_{\max}^2 / k\mu$,

(ii) if $0 < \tau < 1$, then $\|w_k - w_*\| \leq 2\epsilon/\mu$, for $k \to \infty$



(a) SGD (b) SAGA (c) SVRG

Figure 1. Loss residual and error rate of SGD, SVRG, SAGA for Logistic Regression on Covtype data set with 581,012 data points. We compare CRAIG (10% selected subset) (blue) vs. 10% random subset (green) vs. entire data set (orange). CRAIG gives the average speedup of 3x for achieving similar loss residual and error rate across the three optimization methods.

(iii) if $\tau = 0$, *then* $\|w_k - w_*\| \leq (1-\alpha\mu)^k \|w_0 - w_*\| + 2\epsilon/\mu + \alpha\beta C r \gamma_{\max}^2/\mu$,

where $\beta = \sum_{i=1}^n \beta_i$ *is the sum of gradient Lipschitz constants of the component functions.*

The above theorem shows that for $\tau > 0$, IG applied to S converges to a $2\epsilon/\mu$ neighborhood of the optimal solution, with a rate of $\mathcal{O}(1/k^\tau)$ which is the same convergence rate for IG on the entire data set V. As shown in our experiments, in real data sets small weighted subsets constructed by CRAIG provide a close approximation to the full gradient. Hence, applying IG to the weighted subsets returned by CRAIG provides a solution of the same or higher quality compared to the solution obtained by applying IG to the whole data set, in a considerably shorter amount of time.

5. Experiments

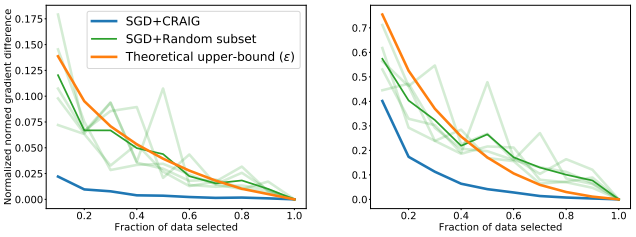
In our experimental evaluation we wish to address the following questions: (1) How do loss and accuracy of IG applied to the subsets returned by CRAIG compare to loss and accuracy of IG applied to the entire data? (2) How small is the size of the subsets that we can select with CRAIG and still get a comparable performance to IG applied to the entire data? And (3) How well does CRAIG scale to large data sets, and extends to non-convex problems? In our experiments, we report the run-time as the wall-clock time for subset selection with CRAIG, plus minimizing the loss using IG or other optimizers with the specified learning rates. For the classification problems, we separately select subsets from each class while maintaining the class ratios in the whole data, and apply IG to the union of the subsets. Note that the upper bounds on the gradient differences derived in Appendix B.1 only hold for points with similar labels. Thus, theoretically we need to select subsets separately. For neural networks, we observed that separately selecting subsets from each class helps the performance. We separately tune each method so that it performs at its best.

5.1. Convex Experiments

In our convex experiments, we apply CRAIG to SGD, as well as SVRG (Johnson & Zhang, 2013), and SAGA (Defazio et al., 2014). We apply L2-regularized logistic regression: $f_i(x) = \ln(1 + \exp(-w^T x_i y_i)) + 0.5\lambda w^T w$ to classify the following two datasets from LIBSVM: (1) *covtype.binary* including 581,012 data points of 54 dimensions, and (2) *Ijcnn1* including 49,990 training and 91,701 test data points of 22 dimensions. As *covtype* does not come with labeled test data, we randomly split the training data into halves to make the training/test split (training and set sets are consistent for different methods).

For the convex experiments, we tuned the learning rate for each method (including the random baseline) by preferring smaller training loss from a large number of parameter combinations for two types of learning scheduling: exponential decay $\alpha_k = \alpha_0 b^k$ and k-inverse $\alpha_k = \alpha_0(1+bk)^{-1}$ with parameters α_0 and b to adjust. For convergence of IG to $2\epsilon/\mu$ neighborhood of the optimal solution, we require that $\sum_{k=0}^\infty \alpha_k = \infty$, and $\sum_{k=0}^\infty \alpha_k^2 = 0$ (Nedić & Bertsekas, 2001). Hence, while the convergence of exponentially decaying learning rate is not theoretically guaranteed, it often worked better in our experiments. Furthermore, following (Johnson & Zhang, 2013) we set λ to 10^{-5}.

CRAIG effectively minimizes the loss. Figure 1(top) compares training loss residual of SGD, SVRG, and SAGA on the 10% CRAIG set (blue), 10% random set (green),



(a) Covtype (b) Ijcnn1

Figure 2. Normed difference between the full gradient, the gradient of the subset found by CRAIG (Eq. 2), and the theoretical upper-bound ϵ (Eq. 8). The values are normalized by the largest full gradient norm. The transparent green lines demonstrate various random subsets, and the opaque green line shows their average.

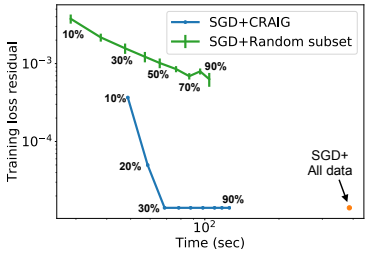


Figure 3. Training loss residual for SGD applied to subsets of size $10\%, 20\%, \cdots, 90\%$ found by CRAIG vs. random subsets of the same size from Ijcnn1. We get 5.6x speedup from applying SGD to subset of size 30% compared to the entire dataset.

and the full dataset (orange). CRAIG effectively minimizes the training data loss (blue line) and achieves the same minimum as the entire dataset training (orange line) but much faster. Also notice that training on the random 10% subset of the data does not effectively minimize the training loss.

CRAIG has a good generalization performance. Figure 1(bottom) shows the test error rate of models trained on CRAIG vs. random vs. the full data. Notice that training on CRAIG subsets achieves the same generalization performance (test error rate) as training on the full data.

CRAIG achieves significant speedup. Figure 1 also shows that CRAIG achieves a similar training loss (top) and test error rate (bottom) as training on the entire set, but much faster. In particular, we obtain a speedup of 2.75x, 4.5x, 2.5x from applying IG, SVRG and SAGA on the subsets of size 10% from covtype obtained by CRAIG. Furthermore, Figure 3 compares the speedup achieved by CRAIG to reach a similar loss residual as that of SGD for subsets of size $10\%, 20\%, \cdots, 90\%$ of Ijcnn1. We get a 5.6x speedup by applying SGD to subsets of size 30% obtained by CRAIG.

CRAIG gradient approximation is accurate. Figure 2 demonstrates the norm of the difference between the weighted gradient of the subset found by CRAIG and the full gradient compared to the theoretical upper-bound ϵ specified in Eq. (8). The gradient difference is calculated by sampling the full gradient at various points in the parameter space. Gradient differences are then normalized by the largest norm of the sampled full gradients. The figure also compares the normed gradient difference between gradients of several random subsets S where each data point is weighted by $|V|/|S|$. Notice that CRAIG's gradient estimate is more accurate than the gradient estimate obtained by the same-size random subset of points (which is how methods like SGD approximate the gradient). This demonstrates that our gradient approximation in Eq. (8) is reliable in practice.

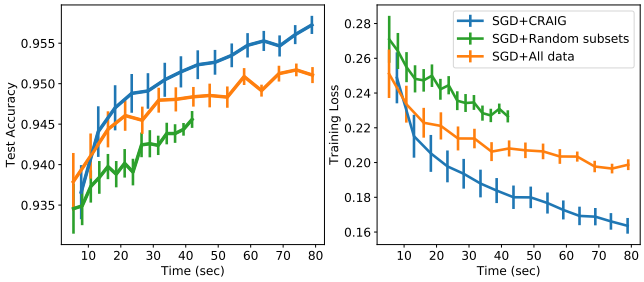
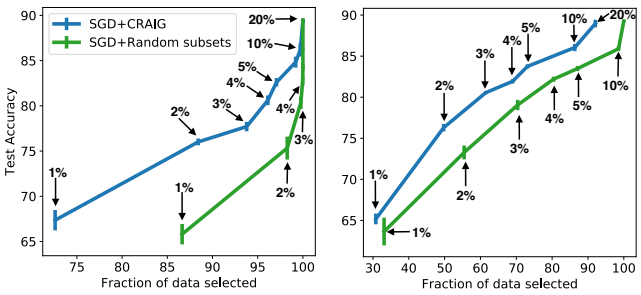


Figure 4. Test accuracy and training loss of SGD applied to subsets found by CRAIG vs. random subsets on MNIST with a 2-layer neural network. CRAIG provides 2x to 3x speedup and a better generalization performance.

5.2. Non-convex Experiments

Our non-convex experiments involve applying CRAIG to train the following two neural networks: (1) Our smaller network is a fully-connected hidden layer of 100 nodes and ten softmax output nodes; sigmoid activation and L2 regularization with $\lambda = 10^{-4}$ and mini-batches of size 10 on MNIST dataset of handwritten digits containing 60,000 training and 10,000 test images. (2) Our large neural network is ResNet-20 for CIFAR10 with convolution, average pooling and dense layers with softmax outputs and L2 regularization with $\lambda = 10^{-4}$. CIFAR 10 includes 50,000 training and 10,000 test images from 10 classes, and we used mini-batches of size 128. Both MNIST and CIFAR10 data sets are normalized into [0, 1] by division with 255. In all these experiments, we report average test accuracy across 10 trials.

CRAIG achieves considerable speedup. Figure 4 shows training loss and test accuracy for training a 2-layer neural net on MNIST. For this problem, we used a constant learning rate of 10^{-2}. Here, we apply CRAIG to select a subset of 50% of the data at the beginning of every epoch and train only on the selected subset with the corresponding per-element stepsizes. Interestingly, in addition to achieving a speedup of 2x to 3x for training the network, the subsets selected by CRAIG provide a better generalization performance compared to models trained on the entire dataset.

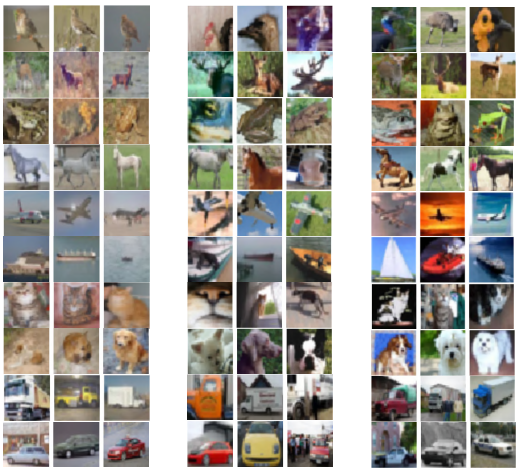


(a) (b)

Figure 5. Test accuracy vs. fraction of data selected during training of ResNet-20 on CIFAR10. (a) At the beginning of ever epoch, a new subset of size 1%, 2%, 3%, 4%, 5%, 10%, or 20% is selected by CRAIG. (b) Every 5 epochs a new subset of similar size is selected by CRAIG. SGD is then applied to training on the selected subsets. The x-axis shows the fraction of training data points that are used by SGD during the training process. Note that for a given subset size, backpropagation is done on the same number of data points for CRAIG and random. However, CRAIG selects a smaller number of distinct data points during the training. Therefore, CRAIG is data-efficient for training neural networks.

CRAIG is data-efficient for training neural networks. Figure 5 shows test accuracy vs. the fraction of data points selected for training ResNet-20 on CIFAR10. We trained the network for 200 epochs, and used the standard learning rate schedule for training ResNet-20 on CIFAR10. I.e., we start with initial learning rate of 0.1, and exponentially decay the learning rate by a factor of 0.1 at epochs 100 and 150. To prevent weights from diverging when training with subsets of all sizes, we used linear learning rate warm-up for 20 epochs from 0. For optimization we used SGD with a momentum of 0.9.

Figure 5a shows the test accuracy when at the beginning of every epoch a subset of size 1%, 2%, 3%, 4%, 5%, 10%, or 20% is chosen at random or by CRAIG from the training data. The network is trained only on the selected subset of a given size for that epoch. For every subset size, the x-axis shows the fraction of training data points that are used during the entire training process. Figure 5b shows the test accuracy when a subset of size 1%, 2%, 3%, 4%, 5%, 10%, or 20% is chosen at random or by CRAIG every 5 epochs. The network is trained only on the selected subset for 5 epochs. Generally, larger subsets or more frequent updates lead to more data exposure and hence better performance. However, since in deep networks the gradients may change rapidly after a small number of parameter updates (Defazio & Bottou, 2019), selecting smaller subsets and more frequent updates result in a larger improvement over the random baseline. Note that for a given subset size, backpropagation is done on the same number of data points for CRAIG and random. However, it can be seen that CRAIG can identify the data points that are effective for training the neural network, and hence achieves a superior test accuracy by training on a smaller fraction of the training data.



(a) First (b) Middle (c) Last

Figure 6. A subset of images selected by CRAIG from CIFAR10. Subsets are selected at the (a) beginning of training (epoch 1), (b) middle of training (epoch 100), and (c) end of training (epoch 200). We notice that during the training, the semantic redundancies decrease considerably, and coreset images better represent various types of images (that are more difficult to learn) in every class.

Insights from CRAIG subsets. Figure 6 shows a subset of images selected by CRAIG for training CIFAR10 at the beginning (6a), middle (6b), and end (6a) of training. Since gradients are more uniformly distributed at initialization, subsets contain semantic redundancies at the beginning of the training (6a). We notice that during the training, semantic redundancies decrease considerably. In particular, as training proceeds coreset images represent groups of images that are more difficult to learn, e.g., contain parts of an object (6b), or have a different foreground/background color than the rest of the images in a class (6c).

6. Conclusion

We developed a method, CRAIG, for selecting a subset (coreset) of data points with their corresponding per-element stepsizes to speed up iterative gradient (IG) methods. In particular, we showed that weighted subsets that minimize the upper-bound on the estimation error of the full gradient, maximize a submodular facility location function. Hence, the subset can be found using a fast greedy algorithm. We proved that IG on subsets S found by CRAIG converges at the same rate as IG on the entire dataset V, while providing a $|V|/|S|$ speedup. In our experiments, we showed that various IG methods, including SGD, SAGA, and SVRG runs up to 6x faster on convex and up to 3x on non-convex problems on subsets found by CRAIG while achieving practically the same training loss and test error.

Acknowledgement

This work was supported in part by the SNSF P2EZP2_172187, and the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA. We also gratefully acknowledge the support of DARPA under Nos. FA865018C7880 (ASED), N660011924033 (MCS); ARO under Nos. W911NF-16-1-0342 (MURI), W911NF-16-1-0171 (DURIP); NSF under Nos. OAC-1835598 (CINES), OAC-1934578 (HDR), CCF-1918940 (Expeditions), IIS-2030477 (RAPID); Stanford Data Science Initiative, Wu Tsai Neurosciences Institute, Chan Zuckerberg Biohub, Amazon, Boeing, Chase, Docomo, Hitachi, Huawei, JD.com, NVIDIA, Dell. J. L. is a Chan Zuckerberg Biohub investigator.

References

Agarwal, P. K., Har-Peled, S., and Varadarajan, K. R. Approximating extent measures of points. *Journal of the ACM (JACM)*, 51(4):606–635, 2004.

Allen-Zhu, Z. Katyusha: The first direct acceleration of stochastic gradient methods. *The Journal of Machine Learning Research*, 18(1):8194–8244, 2017.

Allen-Zhu, Z., Yuan, Y., and Sridharan, K. Exploiting the structure: Stochastic gradient methods using raw clusters. In *Advances in Neural Information Processing Systems*, pp. 1642–1650, 2016.

Asi, H. and Duchi, J. C. The importance of better models in stochastic optimization. *arXiv preprint arXiv:1903.08619*, 2019.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bertsekas, D. P. Incremental least squares methods and the extended kalman filter. *SIAM Journal on Optimization*, 6 (3):807–822, 1996.

Bertsekas, D. P. *Convex optimization algorithms*. Athena Scientific Belmont, 2015a.

Bertsekas, D. P. Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *arXiv preprint arXiv:1507.01030*, 2015b.

Campbell, T. and Broderick, T. Bayesian coreset construction via greedy iterative geodesic ascent. In *International Conference on Machine Learning*, 2018.

Chung, K. L. et al. On a stochastic approximation method. *The Annals of Mathematical Statistics*, 25(3):463–483, 1954.

Cohen, M. B., Musco, C., and Musco, C. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1758–1777. SIAM, 2017.

Defazio, A. and Bottou, L. On the ineffectiveness of variance reduced optimization for deep learning. In *Advances in Neural Information Processing Systems*, pp. 1755–1765, 2019.

Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pp. 1646–1654, 2014.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Frostig, R., Ge, R., Kakade, S., and Sidford, A. Unregularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *International Conference on Machine Learning*, pp. 2540–2548, 2015.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Gürbüzbalaban, M., Ozdaglar, A., and Parrilo, P. Why random reshuffling beats stochastic gradient descent. *arXiv preprint arXiv:1510.08560*, 2015.

Gurbuzbalaban, M., Ozdaglar, A., and Parrilo, P. A. On the convergence rate of incremental aggregated gradient algorithms. *SIAM Journal on Optimization*, 27(2):1035–1048, 2017.

Har-Peled, S. and Mazumdar, S. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pp. 291–300. ACM, 2004.

Hofmann, T., Lucchi, A., Lacoste-Julien, S., and McWilliams, B. Variance reduced stochastic gradient descent with neighbors. In *Advances in Neural Information Processing Systems*, pp. 2305–2313, 2015.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

Katharopoulos, A. and Fleuret, F. Not all samples are created equal: Deep learning with importance sampling. In *International Conference on Machine Learning*, pp. 2525–2534, 2018.

Kaufman, L., Rousseeuw, P., and Dodge, Y. Clustering by means of medoids in statistical data analysis based on the, 1987.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Li, M., Miller, G. L., and Peng, R. Iterative row sampling. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pp. 127–136. IEEE, 2013.

Lin, H. and Bilmes, J. A. Learning mixtures of submodular shells with application to document summarization. *arXiv preprint arXiv:1210.4871*, 2012.

Lin, H., Bilmes, J., and Xie, S. Graph-based submodular selection for extractive summarization. In *Proc. IEEE Automatic Speech Recognition and Understanding (ASRU)*, Merano, Italy, December 2009.

Lin, H., Mairal, J., and Harchaoui, Z. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, pp. 3384–3392, 2015.

Lucic, M., Faulkner, M., Krause, A., and Feldman, D. Training gaussian mixture models at scale via coresets. *The Journal of Machine Learning Research*, 18(1):5885–5909, 2017.

Mangasariany, O. and Solodovy, M. Serial and parallel backpropagation convergence via nonmonotone perturbed minimization. 1994.

Minoux, M. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization techniques*, pp. 234–243. Springer, 1978.

Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrák, J., and Krause, A. Lazier than lazy greedy. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015a.

Mirzasoleiman, B., Karbasi, A., Badanidiyuru, A., and Krause, A. Distributed submodular cover: Succinctly summarizing massive data. In *Advances in Neural Information Processing Systems*, pp. 2881–2889, 2015b.

Mirzasoleiman, B., Zadimoghaddam, M., and Karbasi, A. Fast distributed submodular cover: Public-private data summarization. In *Advances in Neural Information Processing Systems*, pp. 3594–3602, 2016.

Musco, C. and Musco, C. Recursive sampling for the nystrom method. In *Advances in Neural Information Processing Systems*, pp. 3833–3845, 2017.

Nedić, A. and Bertsekas, D. Convergence rate of incremental subgradient algorithms. In *Stochastic optimization: algorithms and applications*, pp. 223–264. Springer, 2001.

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

Qian, N. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

Roux, N. L., Schmidt, M., and Bach, F. R. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in neural information processing systems*, pp. 2663–2671, 2012.

Shalev-Shwartz, S. and Zhang, T. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

Solodov, M. V. Incremental gradient algorithms with stepsizes bounded away from zero. *Computational Optimization and Applications*, 11(1):23–35, 1998.

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

Tseng, P. An incremental gradient (-projection) method with momentum term and adaptive stepsize rule. *SIAM Journal on Optimization*, 8(2):506–531, 1998.

Wei, K., Iyer, R., and Bilmes, J. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pp. 1954–1963, 2015.

Wolsey, L. A. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4): 385–393, 1982.

Xiao, L. and Zhang, T. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Zeiler, M. D. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Zhi-Quan, L. and Paul, T. Analysis of an approximate gradient projection method with applications to the backpropagation algorithm. *Optimization Methods and Software*, 4 (2):85–101, 1994.

A. Convergence Rate Analysis

We firs proof the following Lemma which is an extension of the [(Chung et al., 1954), Lemma 4].

Lemma 3. *Let $u_k \geq 0$ be a sequence of real numbers. Assume there exist k_0 such that*

$$u_{k+1} \leq (1 - \frac{c}{k})u_k + \frac{e}{k^p} + \frac{d}{k^{p+1}}, \quad \forall k \geq k_0,$$

where $e > 0, d > 0, c > 0$ are given real numbers. Then

$$u_k \leq (dk^{-1} + e)(c - p + 1)^{-1}k^{-p+1} + o(k^{-p+1}) \qquad \text{for } c > p - 1, p \geq 1 \tag{21}$$

$$u_k = O(k^{-c}\log k) \qquad \text{for } c = p - 1, p > 1 \tag{22}$$

$$u_k = O(k^{-c}) \qquad \text{for } c < p - 1, p > 1 \tag{23}$$

$$\tag{24}$$

Proof. Let $c > p-1$ and $v_k = k^{p-1}u_k - \frac{d}{k(c-p+1)} - \frac{e}{c-p+1}$. Then, using Taylor approximation $(1+\frac{1}{k})^p = (1+\frac{p}{k}) + o(\frac{1}{k})$ we can write

$$v_{k+1} = (k+1)^{p-1}u_{k+1} - \frac{d}{(k+1)(c-p+1)} - \frac{e}{c-p+1} \tag{25}$$

$$\leq k^{p-1}(1+\frac{1}{k})^{p-1}\Big((1-\frac{c}{k})u_k + \frac{e}{k^p} + \frac{d}{k^{p+1}}\Big) - \frac{d}{(k+1)(c-p+1)} - \frac{e}{c-p+1} \tag{26}$$

$$= k^{p-1}u_k\Big(1 - \frac{c-p+1}{k} + o(\frac{1}{k})\Big) + \frac{e}{k}\Big(1 + \frac{p-1}{k} + o(\frac{1}{k})\Big) \tag{27}$$

$$+ \frac{d}{k^2}\Big(1 + \frac{p-1}{k} + o(\frac{1}{k})\Big) - \frac{d}{(k+1)(c-p+1)} - \frac{e}{c-p+1} \tag{28}$$

$$= \Big(v_k + \frac{d}{k(c-p+1)} + \frac{e}{c-p+1}\Big)\Big(1 - \frac{c-p+1}{k} + o(\frac{1}{k})\Big) \tag{29}$$

$$+ \frac{e}{k}\Big(1 + \frac{p-1}{k} + o(\frac{1}{k})\Big) + \frac{d}{k^2}\Big(1 + \frac{p-1}{k} + o(\frac{1}{k})\Big) \tag{30}$$

$$- \frac{d}{(k+1)(c-p+1)} - \frac{e}{c-p+1} \tag{31}$$

$$= v_k\Big(1 - \frac{c-p+1}{k} + o(\frac{1}{k})\Big) + \frac{d/(c-p+1)}{k(k+1)} + \frac{e(p-1)}{k^2} + \frac{d(p-1)}{k^3} + o(\frac{1}{k^2}) \tag{32}$$

Note that for v_k, we have

$$\sum_{k=0}^{\infty}\Big(1 - \frac{c-p+1}{k} + o(\frac{1}{k})\Big) = \infty$$

and

$$\Big(\frac{d/(c-p+1)}{k(k+1)} + \frac{e(p-1)}{k^2} + \frac{d(p-1)}{k^3} + o(\frac{1}{k^2})\Big)\Big(1 - \frac{c-p+1}{k} + o(\frac{1}{k})\Big)^{-1} \to 0.$$

Therefore, $\lim_{k\to\infty} v_k \leq 0$, and we get Eq. 21. For $p = 1$, we have $u_k \leq \frac{e}{c}$. Hence, u_k converges into the region $u \leq \frac{e}{c}$, with ratio $1 - \frac{c}{k}$.

Moreover, for $p - 1 \geq c$ we have

$$v_{k+1} = u_{k+1}(k+1)^c \leq \Big[(1-\frac{c}{k})u_k + \frac{e}{k^p} + \frac{d}{k^{p+1}}\Big]k^c\Big(1 + \frac{c}{k} + \frac{c^2}{2k^2} + o(\frac{1}{k^2})\Big) \tag{33}$$

$$= \Big(1 - \frac{c^2}{2k^2} + o(\frac{1}{k^2})\Big)v_k + \frac{d}{k^{p-c+1}}\Big(1 + O(\frac{1}{k})\Big) + \frac{e}{k^{p-c}}\Big(1 + \frac{c}{k} + O(\frac{1}{k^2})\Big) \tag{34}$$

$$\leq v_k + \frac{e'}{k^{p-c}} \tag{35}$$

for sufficiently large k. Summing over k, we obtain that v_k is bounded for $p - 1 > c$ (since the series $\sum_{k=1}^{\infty}(1/k^{\alpha})$ converges for $\alpha > 1$) and $v_k = O(\log k)$ for $p = c + 1$ (since $\sum_{i=1}^{k}(1/i) = O(\log k)$). □

In addition, based on [(Chung et al., 1954), Lemma 5] for $u_k \geq 0$, we can write

$$u_{k+1} \leq (1 - \frac{c}{k^s})u_k + \frac{e}{k^p} + \frac{d}{k^t}, \qquad 0 < s < 1, s \leq p < t. \tag{36}$$

Then, we have

$$u_k \leq \frac{e}{c}\frac{1}{k^{p-s}} + o(\frac{1}{k^{p-s}}). \tag{37}$$

A.1. Convergence Rate for Strongly Convex Functions

Proof of Theorem 1

We now provide the convergence rate for strongly convex functions building on the analysis of (Nedić & Bertsekas, 2001). For non-smooth functions, gradients can be replaced by sub-gradients.

Let $w_k = w_0^k$. For every IG update on subset S we have

$$\|w_j^k - w_*\|^2 = \|w_{j-1}^k - \alpha_k\gamma_j\nabla f_j(w_{j-1}^k) - w_*\|^2 \tag{38}$$

$$= \|w_{j-1}^k - w_*\|^2 - 2\alpha_k\gamma_j\nabla f_j(w_{j-1}^k)(w_{j-1}^k - w_*) + \alpha_k^2\|\gamma_j\nabla f_j(w_{j-1}^k)\|^2 \tag{39}$$

$$\leq \|w_{j-1}^k - w_*\|^2 - 2\alpha_k(f_j(w_{j-1}^k) - f_i(w_*)) + \alpha_k^2\|\gamma_i\nabla f_j(w_{j-1}^k)\|^2. \tag{40}$$

Adding the above inequalities over elements of S we get

$$\|w_{k+1} - w_*\|^2 \leq \|w_k - w_*\|^2 - 2\alpha_k\sum_{j\in S}(f_i(w_{j-1}^k) - f_j(w_*)) + \alpha_k^2\sum_{j\in S}\|\gamma_j\nabla f_j(w_{j-1}^k)\|^2 \tag{41}$$

$$= \|w_k - w_*\|^2 - 2\alpha_k\sum_{j\in S}(f_j(w_k) - f_i(w_*)) + 2\alpha_k\sum_{j\in S}(f_j(w_{j-1}^k) - f_j(w_k)) + \alpha_k^2\sum_{j\in S}\|\gamma_j\nabla f_j(w_{j-1}^k)\|^2 \tag{42}$$

Using strong convexity we can write

$$\|w_{k+1} - w_*\|^2 \leq \|w_k - w_*\|^2 - 2\alpha_k\big(\sum_{j\in S}\gamma_j\nabla f_j(w_*)\cdot(w_k - w_*) + \frac{\mu}{2}\|w_k - w_*\|^2\big) + 2\alpha_k\sum_{j\in S}(f_j(w_{j-1}^k) - f_j(w_k)) + \alpha_k^2\sum_{j\in S}\|\gamma_j\nabla f_j(w_{j-1}^k)\|^2 \tag{43}$$

Using Cauchy–Schwarz inequality, we know

$$|\sum_{j\in S}\gamma_j\nabla f_j(w_*)\cdot(w_k - w_*)| \leq \|\sum_{j\in S}\gamma_j\nabla f_j(w_*)\|\cdot\|w_k - w_*\|. \tag{44}$$

Hence,

$$-\sum_{j\in S}\gamma_j\nabla f_j(w_*)\cdot(w_k - w_*) \leq \|\sum_{j\in S}\gamma_j\nabla f_j(w_*)\|\cdot\|w_k - w_*\|. \tag{45}$$

From reverse triangle inequality, and the facts that S is chosen in a way that $\|\sum_{i\in V}\nabla f_i(w_*) - \sum_{j\in S}\gamma_j\nabla f_j(w_*)\| \leq \epsilon$, and that $\sum_{i\in V}\nabla f_i(w_*) = 0$ we have $\|\sum_{j\in S}\gamma_j\nabla f_j(w_*)\| \leq \|\sum_{i\in V}\nabla f_i(w_*)\| + \epsilon = \epsilon$. Therefore

$$\|\sum_{j\in S}\gamma_j\nabla f_j(w_*)\|\cdot\|w_k - w_*\| \leq \epsilon\cdot\|w_k - w_*\| \tag{46}$$

For a continuously differentiable function, the following condition is implied by strong convexity condition

$$\|w_k - w_*\| \leq \frac{1}{\mu}\|\sum_{j\in S}\gamma_j\nabla f_j(w_k)\|. \tag{47}$$

Assuming gradients have a bounded norm $\max_{\substack{x\in\mathcal{X},\\ j\in V}} \|\nabla f_j(w)\| \leq C$, and the fact that $\sum_{j\in S}\gamma_j = n$ we can write

$$\|\sum_{j\in S}\gamma_j \nabla f_j(w_k)\| \leq n\cdot C. \tag{48}$$

Thus for initial distance $\|w_0 - w_*\| = d_0$, we have

$$\|w_k - w_*\| \leq \min(n\cdot C, d_0) = R \tag{49}$$

Putting Eq. 45 to Eq. 49 together we get

$$\begin{aligned}\|w_{k+1} - w_*\|^2 \leq (1-\alpha_k\mu)\|w_k - w_*\|^2 + 2\alpha_k\epsilon R/\mu \\ + 2\alpha_k\sum_{j\in S}(f_j(w_{j-1,k}) - f_j(w_k)) + \alpha_k^2 r\gamma_{\max}^2 C^2.\end{aligned} \tag{50}$$

Now, from convexity of every f_j for $j\in S$ we have that

$$f_j(w_k) - f_j(w_{j-1}^k) \leq \|\gamma_j\nabla f_j(w_k)\|\cdot\|w_{j-1}^k - w_k\|. \tag{51}$$

In addition, incremental updates gives us

$$\|w_{j-1}^k - w_k\| \leq \alpha_k\sum_{i=1}^{j-1}\|\gamma_i\nabla f_i(w_{i-1}^k)\| \leq \alpha_k(j-1)\gamma_{\max}C. \tag{52}$$

Therefore, we get

$$\begin{aligned}2\alpha_k\sum_{j\in S}(f_j(w_k) - f_j(w_{j-1}^k)) + \alpha_k^2 r\gamma_{\max}^2C^2 \\ \leq 2\alpha_k\sum_{i=1}^{r}\gamma_{\max}C\cdot\alpha_k(j-1)\gamma_{\max}C + \alpha_k^2 r\gamma_{\max}^2C^2 \qquad (53)\\ = \alpha_k^2 r^2\gamma_{\max}^2C^2 \qquad (54)\end{aligned}$$

Hence,

$$\|w_{k+1} - w_*\|^2 \leq (1-\alpha_k\mu)\|w_k - w_*\|^2 + 2\alpha_k\epsilon R/\mu + \alpha_k^2 r^2\gamma_{\max}^2C^2. \tag{55}$$

where $\gamma_{\max}$ is the size of the largest cluster, and C is the upperbound on the gradients.

For $0<\tau\leq 1$, the theorem follows by applying Lemma 3 to Eq. 55, with $c=\alpha\mu$, $e = 2\alpha\epsilon R/\mu$, and $d = \alpha^2r^2\gamma_{\max}^2C^2$.

For $\tau = 0$, where we have a constant step size $\alpha_k = \alpha \leq \frac{1}{\mu}$, we get

$$\begin{aligned}\|w_{k+1} - w_*\|^2 \leq (1-\alpha\mu)^{k+1}\|w_0 - w_*\|^2 \\ + 2\alpha\epsilon R\sum_{j=0}^{k}(1-\alpha\mu)^j/\mu + \alpha^2r^2\gamma_{\max}^2C^2\sum_{j=0}^{k}(1-\alpha\mu)^j\end{aligned} \tag{56}$$

Since $\sum_{j=0}^{k}(1-\alpha\mu)^j \leq \frac{1}{\alpha\mu}$, we get

$$\|w_{k+1} - w_*\|^2 \leq (1-\alpha\mu)^{k+1}\|w_0 - w_*\|^2 + 2\alpha\epsilon R/(\alpha\mu^2) + \alpha^2r^2\gamma_{\max}^2C^2/(\alpha\mu), \tag{57}$$

and therefore,

$$\|w_{k+1} - w_*\|^2 \leq (1-\alpha\mu)^{k+1}\|w_k - w_*\|^2 + 2\epsilon R/\mu^2 + \alpha r^2\gamma_{\max}^2C^2/\mu. \tag{58}$$

A.2. Convergence Rate for Strongly Convex and Smooth Component Functions

Proof of Theorem 2

IG updates for cycle k on subset S can be written as

$$w_{k+1} = w_k - \alpha_k(\sum_{j\in S}\gamma_j \nabla f_j(w_k) - e_k) \tag{59}$$

$$e_k = \sum_{j\in S}\gamma_i(\nabla f_j(w_k) - \nabla f_j(w_{j-1}^k)) \tag{60}$$

Building on the analysis of (Gürbüzbalaban et al., 2015), for convex and twice continuously differentiable function, we can write

$$\sum_{j\in S}\gamma_j \nabla f_j(w_k) - \sum_{j\in S}\gamma_j \nabla f_j(w_*) = A_k^r(w_k - w_*) \tag{61}$$

where $A_k^r = \int_0^1 \nabla^2 f(w_* + \tau(w_k - w_*))d\tau$ is average of the Hessian matrices corresponding to the r (weighted) elements of S on the interval $[w_k, w_*]$.

From Eq. 61 we have

$$\sum_{i\in V}(\nabla f_i(w_k) - \nabla f_i(w_*)) - \sum_{j\in S}\gamma_j(\nabla f_j(w_k) - \nabla f_j(w_*)) = A_k(w_k - w_*) - A_k^r(w_k - w_*), \tag{62}$$

where A_k is average of the Hessian matrices corresponding to all the n component functions on the interval $[w_k, w_*]$. Taking norm of both sides and noting that $\sum_{i\in V} f_i(w_*) = 0$ and hence $\|\sum_{j\in S}\gamma_j f_j(w_*)\| \leq \epsilon$, we get

$$\|(A_k - A_k^r)(w_k - w_*)\| = \|\Big(\sum_{i\in V}\nabla f_i(w_k) - \sum_{j\in S}\gamma_j \nabla f_j(w_k)\Big) + \sum_{j\in S}\gamma_j f_j(w_*)\| \leq 2\epsilon, \tag{63}$$

where ϵ is the estimation error of the full gradient by the weighted gradients of the elements of the subset S, and we used $\|\sum_{i\in V}\nabla f_i(w_k) - \sum_{j\in S}\gamma_j \nabla f_j(w_k)\| \leq \epsilon$.

Substituting Eq. 61 into Eq. 59 we obtain

$$w_{k+1} - w_* = (I - \alpha_k A_k^r)(w_k - w_*) + \alpha_k e_k \tag{64}$$

Taking norms of both sides, we get

$$\|w_{k+1} - w_*\| \leq \|(I - \alpha_k A_k^r)(w_k - w_*\|) + \alpha_k \|e_k\| \tag{65}$$

Now, we have

$$\|(I - \alpha_k A_k^r)(w_k - w_*)\| = \|I(w_k - w_*) - \alpha_k A_k^r(w_k - w_*)\| \tag{66}$$

$$= \|I(w_k - w_*) - \alpha_k(A_k^r - A_k)(w_k - w_*) - \alpha_k A_k(w_k - w_*)\| \tag{67}$$

$$\leq \|(I - \alpha_k A_k)(w_k - w_*)\| + \alpha_k\|(A_k - A_k^r)(w_k - w_*)\| \tag{68}$$

$$\leq \|(I - \alpha_k A_k)(w_k - w_*)\| + 2\alpha_k\epsilon \tag{69}$$

Substituting into Eq. 65, we obtain

$$\|w_{k+1} - w_*\| \leq \|I - \alpha_k A_k\| \cdot \|w_k - w_*\| + 2\alpha_k\epsilon + \alpha_k\|e_k\| \tag{70}$$

From strong convexity of $\sum_{i\in V} f_i(w)$, and gradient smoothness of each component $f_i(w)$ we have

$$\mu I_n \preceq \sum_{i\in V}\nabla^2 f_i(w), A_k \preceq \beta I_n, \quad x \in \mathcal{X}, \tag{71}$$

where $\beta = \sum_{i \in V} \beta_i$ In addition, from the gradient smoothness of the components we can write

$$\|e_k\| \leq \sum_{j \in S} \gamma_j \beta_j \|w_k - w_j^k\| \tag{72}$$

$$\leq \sum_{j \in S} \gamma_j \beta_j \sum_{i=1}^{j-1} \|w_{i-1}^k - w_i^k\| \tag{73}$$

$$\leq \sum_{j \in S} \gamma_j \beta_j \alpha_k \sum_{i=1}^{j-1} \|\gamma_i \nabla f_i(w_i^k)\| \tag{74}$$

$$\leq \alpha_k \beta C r \gamma_{\max}^2, \tag{75}$$

where in the last line we used $|S| = r$. Therefore,

$$\|w_{k+1} - w_*\| \leq \max(\|1 - \alpha_k \mu\|, \|1 - \alpha_k \beta\|) \|w_k - w_*\| + 2\alpha_k \epsilon + \alpha_k^2 \beta C r \gamma_{\max}^2 \tag{76}$$

$$\leq (1 - \alpha_k \mu) \|w_k - w_*\| + 2\alpha_k \epsilon + \alpha_k^2 \beta C r \gamma_{\max}^2 \quad \text{if} \quad \alpha_k \beta \leq 1. \tag{77}$$

For $0 < \tau \leq 1$, the theorem follows by applying Lemma 3 to Eq. 76 with $c = \alpha\mu$, $e = 2\alpha\epsilon$, $d = \alpha^2 \beta C r \gamma_{\max}^2$. For $\tau = 0$, where we have a constant step size $\alpha_k = \alpha \leq \frac{1}{\beta}$, we get

$$\|w_{k+1} - w_*\| \leq (1 - \alpha\mu)^{k+1} \|w_k - w_*\| + 2\alpha\epsilon \sum_{i=0}^{k} (1 - \alpha\mu)^i + \alpha^2 \sum_{i=0}^{k} (1 - \alpha\mu)^i \beta C r \gamma_{\max}^2 \tag{78}$$

$$\leq (1 - \alpha\mu)^{k+1} \|w_k - w_*\| + 2\epsilon/\mu + \alpha\beta C r \gamma_{\max}^2/\mu, \tag{79}$$

$$\leq (1 - \alpha\mu)^{k+1} \|w_k - w_*\| + 2\epsilon/\mu + C r \gamma_{\max}^2/\mu, \tag{80}$$

where the inequality in Eq. 79 follows since $\sum_{i=0}^{k} (1 - \alpha\mu)^i \leq \frac{1}{\alpha\mu}$.

B. Norm of the Difference Between Gradients

B.1. Convex Loss Functions

For ridge regression $f_i(w) = \frac{1}{2}(\langle x_i, w \rangle - y_i)^2 + \frac{\lambda}{2}\|w\|^2$, we have $\nabla f_i(w) = x_i(\langle x_i, w \rangle - y_i) + \lambda w$. Therefore,

$$\|\nabla f_i(w) - \nabla f_j(w)\| = (\|x_i - x_j\|.\|w\| + \|y_i - y_j\|)\|x_j\| \tag{81}$$

For $\|x_i\| \leq 1$, and $|y_i - y_j| \approx 0$ we get

$$\|\nabla f_i(w) - \nabla f_j(w)\| \leq \|x_i - x_j\| O(\|w\|) \tag{82}$$

For reguralized logistic regression with $y \in \{-1, 1\}$, we have $\nabla f_i(w) = y_i/(1 + e^{y_i \langle x_i, w \rangle})$. For $y_i = y_j$ we get

$$\|\nabla f_i(w) - \nabla f_j(w)\| \quad = \quad \frac{e^{\|x_i - x_j\|.\|w\|} - 1}{1 + e^{-\langle x_i, x \rangle}} \|x_j\|. \tag{83}$$

For $\|x_i\| \leq 1$, using Taylor approximation $e^x \leq 1 + x$, and noting that $\frac{1}{1 + e^{-\langle x_i, w \rangle}} \leq 1$ we get

$$\|\nabla f_i(w) - \nabla f_j(w)\| \leq \frac{\|x_i - x_j\|.\|w\|}{1 + e^{-\langle x_i, w \rangle}} \|x_j\| \leq \|x_i - x_j\| O(\|w\|). \tag{84}$$

For classification, we require $y_i = y_j$, hence we can select subsets from each class and then merge the results. On the other hand, in ridge regression we also need $|y_i - y_j|$ to be small. Similar results can be deduced for other loss functions including square loss, smoothed hinge loss, etc.

Assuming $\|w\|$ is bounded for all $w \in \mathcal{W}$, upper-bounds on the euclidean distances between the gradients can be pre-computed.

B.2. Neural Networks

Formally, consider an L-layer perceptron, where $w^{(l)} \in \mathbb{R}^{M_l \times M_{l-1}}$ is the weight matrix for layer l with M_l hidden units, and $\sigma^{(l)}(.)$ be a Lipschitz continuous activation function. Then, let

$$x_i^{(0)} = x_i, \tag{85}$$
$$z_i^{(l)} = w^{(l)} x_i^{(l-1)}, \tag{86}$$
$$x_i^{(l)} = \sigma^{(l)}(z_i^{(l)}). \tag{87}$$

With

$$\Sigma_l'(z_i^{(l)}) = \text{diag}\big(\sigma'^{(l)}(z_{i,1}^{(l)}), \cdots \sigma'^{(l)}(z_{i,M_l}^{(l)})\big), \tag{88}$$
$$\Delta_i^{(l)} = \Sigma_l'(z_i^{(l)}) w_{l+1}^T \cdots \Sigma_l'(z_i^{(L-1)}) w_L^T, \tag{89}$$

we have

$$\begin{aligned}
&\|\nabla f_i(w) - \nabla f_j(w)\| \\
&= \|\big(\Delta_i^{(l)} \Sigma_L'(z_i^{(L)}) \nabla f_i^{(L)}(w)\big)\big(x_i^{(l-1)}\big)^T - \big(\Delta_j^{(l)} \Sigma_L'(z_j^{(L)}) \nabla f_j^{(L)}(w)\big)\big(x_j^{(l-1)}\big)^T\| && (90)\\
&\leq \|\Delta_i^{(l)}\| \cdot \|x_i^{(l-1)}\| \cdot \|\Sigma_L'(z_i^{(L)}) \nabla f_i^{(L)}(w) - \Sigma_L'(z_j^{(L)}) \nabla f_j^{(L)}(w)\| \\
&\quad + \|\Sigma_L'(z_j^{(L)}) \nabla f_i^{(L)}(w)\| \cdot \|\Delta_i^{(l)}\big(x_i^{(l-1)}\big)^T - \Delta_j^{(l)}\big(x_j^{(l-1)}\big)^T\| && (91)\\
&\leq \|\Delta_i^{(l)}\| \cdot \|x_i^{(l-1)}\| \cdot \|\Sigma_L'(z_i^{(L)}) \nabla f_i^{(L)}(w) - \Sigma_L'(z_j^{(L)}) \nabla f_j^{(L)}(w)\| \\
&\quad + \|\Sigma_L'(z_j^{(L)}) \nabla f_i^{(L)}(w)\| \cdot \big(\|\Delta_i^{(l)}\| \cdot \|x_i^{(l-1)}\| + \|\Delta_j^{(l)}\| \cdot \|x_j^{(l-1)}\|\big) && (92)\\
&\leq \underbrace{\max_{l,i} \big(\|\Delta_i^{(l)}\| \cdot \|x_i^{(l-1)}\|\big)}_{c_1} \cdot \|\Sigma_L'(z_i^{(L)}) \nabla f_i^{(L)}(w) - \Sigma_L'(z_j^{(L)}) \nabla f_j^{(L)}(w)\| \\
&\quad \underbrace{+ \|\Sigma_L'(z_i^{(L)}) \nabla f_i^{(L)}(w)\| \cdot \max_{l,i,j} \big(\|\Delta_i^{(l)}\| \cdot \|x_i^{(l-1)}\| + \|\Delta_j^{(l)}\| \cdot \|x_j^{(l-1)}\|\big)}_{c_2} && (93)
\end{aligned}$$

Various weight initialization (Glorot & Bengio, 2010) and activation normalization techniques (Ioffe & Szegedy, 2015; Ba et al., 2016) uniformise the activations across samples. As a result, the variation of the gradient norm is mostly captured by the gradient of the loss function with respect to the pre-activation outputs of the last layer of our neural network (Katharopoulos & Fleuret, 2018). Assuming $\|\Sigma_L'(z_i^{(L)}) \nabla f_i^{(L)}(w)\|$ is bounded, we get

$$\|\nabla f_i(w) - \nabla f_j(w)\| \leq c_1 \|\Sigma_L'(z_i^{(L)}) \nabla f_i^{(L)}(w) - \Sigma_L'(z_j^{(L)}) \nabla f_j^{(L)}(w)\| + c_2, \tag{94}$$

where c_1, c_2 are constants. The above analysis holds for any affine operation followed by a slope-bounded non-linearity ($|\sigma'(w)| \leq K$).