A General and Fine-grained Analysis of the Variance Reduction Technique in SVRG and a Novel Acceleration Method

Abstract

Stochastic gradient descent (SGD) with variance reduction technique such as SVRG is efficient to train parameters of many machine learning algorithms. Although many variants of SVRG have been proposed, the analysis of variance has not been thoroughly discussed. Besides, the variants of SVRG have to keep a snapshot of the full gradient for every epoch, which is computationally expensive. In this paper, we propose a general framework EUI to unify the existing variants of SVRG, and then provide a fine-grained analysis of the variance from a new prospective. Moreover, a new variant of SGD with the variance reduction technique named SAMPLEVR has been proposed. SAMPLEVR replaces the full gradient computation with an estimation, thus decreasing the computational cost significantly. Both the theoretical analysis and the empirical studies show that SAMPLEVR make the training loss converge at a linear rate, and outperforms its counterparts significantly.

Introduction

Many machine learning problems such as classification and regression can be presented as the optimisation problem described by Equation 1. $F(\omega)$ denotes the training loss or the loss function which is the sum of the finite functions, i.e. $\nabla f_i(\omega)$ with $i \in \{1, 2, ..., n\}$. ω is the parameter of the machine learning model, and n represents the size of the training data. $R(\omega)$ is the regulariser which is used to prevent overfitting.

$$\min F(\omega), \quad F(\omega) = \frac{1}{n}\sum_{i=1}^{n} f_i(\omega) + R(\omega) \tag{1}$$

Gradient descent (GD) is used to train the parameters for such underlying machine learning problems. Since GD computes a full gradient every iteration, it performs a large amount of gradient calculations. This would affect the performance significantly in the presence of a large amount of training data. The stochastic gradient descent (SGD) overcomes the weakness of GD by using a stochastic gradient instead of the full gradient to train parameter. However, variance caused by the stochastic gradient usually impairs convergence of the training loss. Specifically, when the parameter is close to the optimum, it is increasingly difficult to make a further progress due to the variance. Conventional studies show that a decaying learning rate can be used to decrease the variance. But the training loss converges slowly when the learning rate is small.

Recently, Johnson et al. improve SGD with the variance reduction technique named SVRG which uses a constant learning rate to train the parameter (Johnson and Zhang 2013). Based on the variance reduction technique adopted by SVRG, many variants of SVRG such as S2GD (Konečnỳ and Richtárik 2013), mS2GD (Konečnỳ et al. 2016), EMGD (Zhang, Mahdavi, and Jin 2013), SVR-GHT (Li et al. 2016), Prox-SVRG (Xiao and Zhang 2014), SVRG++ (Allen-Zhu and Yuan 2016), and Katyusha (Allen-Zhu 2016) have been proposed. However, the analysis of the variance, which is essential to understand and exploit the variance reduction technique, lacks enough discussion. Although Zeyuan et al. present an upper bound of the variance, this achievement is obtained with many simplifications (Allen-Zhu and Hazan 2016). Moreover, those existing algorithms are organised by epochs. One epoch consists of some iterations. SVRG and its variants have to keep a snapshot of the full gradient for every epoch, leading to a large amount of gradient calculations. When the size of the training data is huge, the snapshot of the full gradient is extremely time-consuming. In short, it is meaningful to give a quantitative analysis of the variance in general settings so that the potential of the variance reduction technique can be exploited. Additionally, if the snapshot of the full gradient in an epoch can be avoided, SVRG and its variants will be accelerated a lot, which results in a better performance of the convergence.

As an important improvement of SVRG and its variants, we provide a thorough analysis about the variance from a new prospective. To present the analysis, we propose a general framework named EUI, and then perform the analysis under the framework. The update rule of the parameter in the variance reduction technique can be divided into three aspects, including the variance source, the variance reducer and the progressive direction. The variance reducer is the real reason to reduce the variance. More specifically, we provide both lower and upper bounds of the variance, and then analyse improvement of the variance reduction in the existing variants of SVRG. Moreover, a new variant of SGD with the variance reduction technique denoted by SAMPLEVR is proposed, which replaces the snapshot of the full gradient

Copyright © 2017, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

by using an unbiased estimation, and accelerates the convergence of the training loss. The contributions of the paper are outlined as follows:

- A general framework, i.e. EUI is proposed to unify the existing variants of SVRG.
- The variance caused by the stochastic gradient is analysed in the underlying framework. Both lower and upper bounds of the variance are presented in general settings.
- A new variant of SGD with the variance reduction technique, i.e. SAMPLEVR is proposed, which achieves a linear convergence rate with low gradient complexity.
- Extensive evaluation tests show that SAMPLEVR outperforms other previous work significantly.

To keep the paper reasonably concise, a list of symbols used in the paper and the proofs are given in support materials. The paper is organised as follows. Section reviews recent related work about SGD with the variance reduction technique. Section presents the general framework, i.e. EUI, and then provides the analysis of the variance reduction under the framework. Section presents a new reduced variance SGD, i.e. SAMPLEVR, and provides the theoretical analysis of both the performance of convergence and the gradient complexity. Section demonstrates the extensive performance evaluations to verify our theoretical analysis. Section concludes this paper.

Related work

Various variants of SGD with the variance reduction technique have been proposed, including SAG (Schmidt, Roux, and Bach 2016), SAGA (Defazio, Bach, and Lacoste-Julien 2014), SDCA (**?**), and SVRG (Johnson and Zhang 2013) and its variants and so on. It is noting that SAG, SAGA and SDCA adopt different variance reduction techniques from SVRG and its variants. Although variance reduction techniques used in SAG, SAGA and SDCA are competitive with that of SVRG and its variants, this paper focuses on the variance reduction technique used in SVRG and its variants. To the best of our knowledge, the variants of SVRG includes S2GD (Konečnỳ and Richtárik 2013), mS2GD (Konečnỳ et al. 2016), EMGD (Zhang, Mahdavi, and Jin 2013), SVRGHT (Li et al. 2016), Prox-SVRG (Xiao and Zhang 2014), SVRG++ (Allen-Zhu and Yuan 2016), and CHEAPSVRG (Shah et al. 2016).

Zeyuan et al. have presented an upper bound of the variance caused by the stochastic gradients when using the variance reduction technique (Allen-Zhu and Hazan 2016). Such upper bound of the variance is subject to many simplifications. Our analysis about the variance is a complement to that work, which is obtained in general settings, and does not require the simplifications assumed in that work. Vatsal et al. have proposed CHEAPSVRG which uses sampled instances to estimate the full gradient (Shah et al. 2016). The number of those sampled instances is a parameter which needs to be identified before running of CHEAPSVRG. Comparing with CHEAPSVRG, our proposed algorithm, i.e. SAMPLEVR does not need to tune extra parameters. Additionally, although both CHEAPSVRG and SAMPLEVR can achieve a linear convergence rate, the analysis of CHEAPSVRG is obtained with two extra strong assumptions, which are not required in SAMPLEVR. Furthermore, extensive empirical studies show that SAMPLEVR outperforms CHEAPSVRG significantly.

Algorithm 1 EUI: the general framework of reduced variance SGD

Require: $\omega_0{=}\tilde{\omega}_0{=}\mathbf{0}$. $\forall i_t{\in}\{1,2,...,n\}$ where t is a non-negative integer.

Epoch: identify the sequence of epoch size $\{m_0, m_1, ..., m_S\}{\leftarrow}\mathcal{E}(s)$ with $s \in \{0, 1, ..., S\}$;
for $s = 0, 1, 2, ..., S-1$ **do**
  $\omega_0 = \tilde{\omega}_s$;
  $g = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\omega}_s)$;
  for $t = 0, ..., m_s{-}1$ **do**
    pick an instance $<x_{i_t}, y_{i_t}>$ randomly;
    $v = \nabla f_{i_t}(\omega_{i_t}) - \nabla f_{i_t}(\tilde{\omega}_s)$;
    $\gamma_t = v + g$;
    Update: $\omega_{t+1} = \mathcal{U}(\eta, \omega_t, \gamma_t)$;
  Identify: $\tilde{\omega}_{s+1}{\leftarrow}\mathcal{I}(\omega_j)$ with $j{\in}\{0, 1, ..., m_s\}$;
return $\tilde{\omega}_S$.

EUI: the general framework of reduced variance SGD

Framework

As illustrated in Algorithm 1, we present a general framework named EUI which contains a loop of epochs. Every epoch consists of a number of iterations. The epoch size needs to be identified, as calculated by the function $\mathcal{E}$. After that, a snapshot of the full gradient is computed at the beginning of an epoch. During the training of the parameter in an epoch, EUI randomly picks an instance x_{i_t} from the training data first. The parameter is then trained by the update rule, as denoted by the function $\mathcal{U}$. At the end of an epoch, the global parameter $\tilde{\omega}_s$ will be identified by the local parameter ω_t with $t \in \{0, 1, ..., m_s\}$, which is shown by the function $\mathcal{I}$.

Table 1 illustrates that SVRG and its variants can be unified by EUI when the functions, i.e. $\mathcal{E}$, $\mathcal{U}$, and $\mathcal{I}$ are implemented. For example, the function $\mathcal{E}$ in SVRG is implemented by a constant, that is, $m_s{=}C$ with $s{=}\{0, 1, ..., S{-}1\}$. The function $\mathcal{U}$ in SVRG is implemented by the steepest descent, and the function $\mathcal{I}$ is implemented by either any of the local parameters, i.e. ω_j with $j \in \{0, 1, ..., m_s\}$ or ω_{m_s}. Compared to SVRG, its variants implement those functions by using different strategies. Those strategies are explained with the analysis of the variance reduction in the following part.

The analysis of the variance

As illustrated in Table 1, the variants of SVRG usually adopt the same update rule of the parameter with SVRG. The update rule is shown by Equ. 3. We take SVRG as an example

Table 1: Variants of SVRG can be unified by the general framework EUI when the functions $\mathcal{E}, \mathcal{U}$ and $\mathcal{I}$ are implemented.

Name	Algorithms					
	SVRG	S2GD	EMGD	SVR-GHT	Prox-SVRG	SVRG++
$\mathcal{E}$	$m_s{=}C$	$P(m_s{=}t){=}\frac{\phi(t)}{\sum_{t=1}^{M}\phi(t)}$ †	$m_s{=}C$	$m_s{=}C$	$m_s{=}C$	$m_s{=}2^sC$
$\mathcal{U}$	$\omega_t{-}\eta\gamma_t$	$\omega_t{-}\eta\gamma_t$	$\omega_t{-}\mathbb{B}_{\Delta_t}(\eta\gamma_t)$	$\mathcal{H}_k(\omega_t{-}\eta\gamma_t)$	$\omega_t{-}\eta\gamma_t$ ‡	$\omega_t{-}\eta\gamma_t$ ‡
$\mathcal{I}$	randomly pick any of ω_j with $j{\in}\{0,1,...,m_s{-}1\}$	ω_{m_s}	$\frac{1}{m_s+1}\sum_{i=0}^{m_s}\omega_i$	ω_{m_s}	$\frac{1}{m_s}\sum_{i=1}^{m_s}\omega_i$	$\frac{1}{m_s}\sum_{i=1}^{m_s}\omega_i$
	ω_{m_s}					

† $\phi(t){=}(1{-}\check{\mu}\eta)^{M-t}$. $P(m_s{=}t)$ means the probability of $m_s{=}t$, which shows that a large epoch size is used with a high probability.
‡ The update rules of Prox-SVRG and SVRG++ are presented in the setting of the differentiable optimisation objective.

to present the analysis of the variance, and the improvement of variance reduction in other existing algorithms.

As illustrated in Equ. 3, the first item of γ_t is the stochastic gradient which is denoted by the "variance source". The second item of γ_t is denoted by the "variance reducer" which is used to reduce the variance. The third item of γ_t is denoted by the "progressive direction" which makes sure that γ_t will not be too far away from the full gradient when updating parameter in an epoch. The update rule of the parameter in SGD and GD are denoted by γ_t^{SGD} and γ_t^{GD}, respectively. We refer the variance of SGD to the maximum, and the variance of GD to the minimum.

$$\gamma_t^{\text{SGD}} = \nabla f_{i_t}(\omega_{i_t}) - \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{\omega}) + \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{\omega}); \quad (2)$$

$$\gamma_t = \nabla f_{i_t}(\omega_{i_t}) - \nabla f_{i_t}(\tilde{\omega}) + \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{\omega}); \quad (3)$$

$$\gamma_t^{\text{GD}} = \nabla f_{i_t}(\omega_{i_t}) - \nabla f_{i_t}(\omega_{i_t}) + \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{\omega}); \quad (4)$$

It is obvious that the difference among the update rule of SGD, GD and SVRG is the variance reducer. SGD has the maximal variance because its variance reducer is a constant which does not help to reduce the variance. GD has no variance because that its variance reducer decreases all the variance caused by the variance source. The variance reducer in SVRG is a tradeoff between those of SGD and GD. It does not reduce all the variance like that of GD because that its input parameter, i.e. $\tilde{\omega}$ becomes stale against ω_t when ω_t is updated during the iterations in the epoch. The variance due to the staleness will be accumulated with the iterative updates of the parameter ω_t. Such the staleness of the parameter can be measured by the distance d_t with $d_t = \| \omega_t{-}\tilde{\omega} \|^2$. $d_0 = \| \omega_0{-}\tilde{\omega} \|^2 = 0$ holds according to the framework. If γ_t is p-dimensional, and can be denoted by $\gamma_t = (a_{t1}, a_{t2}..., a_{tp})$, we obtain Theorem 1 as follows.

Theorem 1. *After t iterations in an epoch, the distance d_t holds that $d_t{=}\eta^2\sum_{j=1}^{p}\left(\sum_{i=1}^{t}a_{ij}\right)^2$. Furthermore, d_t has an upper bound such that $d_t{\leq}\eta^2t^2p\left(\frac{1}{tp}\sum_{i=1}^{t}\sum_{j=1}^{p}a_{ij}^2\right)$, and a lower bound such that $d_t{\geq}\eta^2t^2p\left(\frac{1}{tp}\sum_{i=1}^{t}\sum_{j=1}^{p}a_{ij}\right)^2$.*

As demonstrated in Theorem 1, the distance becomes large with a large learning rate η, a high dimension p, and iterative updates of the parameter. Given a specific machine learning task, if a constant learning rate is adopted, the variance is dominated by the design of the epoch size m_s and the update gradient, i.e. γ_t.

As illustrated in Table 1, the epoch size, i.e. m_s is designed as a constant in EMGD, SVR-GHT and Prox-SVRG, and designed as an ascending variable for S2GD and SVRG++. It is noting that the design of an appropriate epoch size is not trivial. A large size of an epoch causes much variance due to the large distance according to Theorem 1. However, a small epoch size means that more epochs are required to make the loss function converge within a given bound. Since the full gradient is computed every epoch, a large amount of epochs lead to a large number of gradient calculations which are very time-consuming.

The update rule has an immediately impact on the variance according to Theorem 1. Most variants of SVRG including S2GD, Prox-SVRG, and SVRG++ adopt the same update rule with SVRG. EMGD updates the parameter within a decaying bound in the high dimensional space. The bound is decaying so that the value of parameters will not be changed sharply, which decreases the variance. SVR-GHT is designed for a class of optimisation problems with cardinality constraints where there are at most k non-zero entries in the optimal solution. SVR-GHT introduces a hard-thresholding mechanism to update the parameter, which keeps the k largest entries and sets other entries to zero. Benefiting from the property of the optimisation objective, the variance of SVR-GHT is bounded.

The identification of the parameters is to provide the initial parameters, i.e. ω^{s+1} for the next epoch. The variance in the current epoch will be passed to the next epoch via the identification of the parameter. Generally, the identification of the parameter is a tradeoff between the updates and the variance. First, we would like to use the updates of parameter in the next epoch because those updates help converge the loss function. Second, the variance existing in those updates impedes the convergence of the loss function. The more updates we use, the more variance is introduced

Algorithm 2 SAMPLEVR

Require: $\alpha = 0.01, g = \mathbf{0}$, and $\tilde{\omega}_0 = \mathbf{0}$. $\forall i_t \in \{1, 2, ..., n\}$. ϵ is a positive real number.

for $s = 0, 1, 2, ..., S-1$ **do**
  $\omega_0 = \tilde{\omega}_s$;
  for $t = 0, 1, ..., m-1$ **do**
    pick an instance $<x_{i_t}, y_{i_t}>$ randomly;
    $v = \nabla f_{i_t}(\omega_t) - \nabla f_{i_t}(\tilde{\omega}_s)$;
    $\gamma_t = v + \dot{g}$;
    $\omega_{t+1} = \omega_t - \eta\gamma_t$;
  $\tilde{\omega}_{s+1}$ is identified by using any of ω_i with $i \in \{0, 2, ..., m-1\}$ randomly;
  $k = -\frac{s \log \frac{\alpha}{2}}{\epsilon}$;
  $\dot{g} = \frac{1}{k} \sum_{j=1}^{k} \nabla f_{i_j}(\tilde{\omega}_{s+1})$;
return $\tilde{\omega}_S$.

in the next epoch. As illustrated in Table 1, the majority of previous work use ω_{m_s} as the initial parameter of the next epoch, which contains all the updates of the current epoch. EMGD, Prox-SVRG and SVRG++ use the mean of the sequence of the parameters which leads to less variance for the initial parameter of the next epoch, but discards some updates of the parameter.

SAMPLEVR: a new reduced variance SGD

Although the variance reduction technique is effective for decreasing the variance caused by the stochastic gradient, it has to keep a snapshot of the full gradient every epoch. Unfortunately, the computation of the full gradient requires extensive gradient calculations which are extremely time-consuming. We design a new reduced variance SGD which uses an estimation of the full gradient to replace the real computation of it. As illustrated in Algorithm 2, the new variant of SGD with the variance reduction technique is denoted by SAMPLEVR. SAMPLEVR estimates the full gradient by using stochastic gradients which are computed by using k sampled instances from the training data. The mean of the stochastic gradients denoted by $\dot{g}$ is used as the progressive direction for the next epoch. Since $\mathbb{E}(\dot{g}) = \mathbb{E}\left(\frac{1}{k}\sum_{i=1}^{k}\nabla f_i(\tilde{\omega})\right) = \mathbb{E}\left(\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{\omega})\right) = \nabla F(\tilde{\omega})$ holds according to Equ. 5, the new update gradient, i.e. $\dot{\gamma}_t$ is the same with γ_t in probability, that is, $\mathbb{E}(\dot{\gamma}_t) = \mathbb{E}(\gamma_t) = \nabla F(\omega_t)$.

$$\dot{\gamma}_t = \nabla f_{i_t}(\omega_t) - \nabla f_{i_t}(\tilde{\omega}) + \frac{1}{k}\sum_{i=1}^{k}\nabla f_{i_k}(\tilde{\omega}); \quad (5)$$

Although the estimation of the full gradient, i.e. $\dot{g}$ is unbiased, the variance between $\dot{g}$ and g impedes the convergence of the training loss, especially when the parameter gets close to the optimum. This problem is mitigated in SAMPLEVR by increasing the number of the sampled instances over epochs linearly. In specific, according to Assumption 1 and Assumption 2, every stochastic gradient $\nabla f_i(\omega)$ with $i \in \{1, 2, ..., n\}$ is bounded by a positive constant denoted by C, that is, $\| \nabla f_i(\omega) \| \leq C$. Without loss of generality, suppose $\nabla f_i(\omega)$ with $i \in \{1, 2, ..., n\}$ is p-dimensional, and can be denoted by $\nabla f_i(\omega) = (b_{i1}, b_{i2}, ..., b_{ip})^{\mathrm{T}}$. Considering any two stochastic gradients, e.g. $\nabla f_i(\omega)$ and $\nabla f_j(\omega)$, we obtain: $\max_l \| b_{il} - b_{jl} \| \leq \max_l (\| b_{il} \| + \| b_{jl} \|) \leq 2C$ with $l \in \{1, 2, ..., p\}$. The variance between the full gradient and its estimation is denoted by ν with $\nu = \dot{g} - \mathbb{E}(\dot{g}) = \frac{1}{k}\sum_{i=1}^{k}\nabla f_i(\tilde{\omega}_{s+1}) - \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{\omega}_{s+1})$. Generally, considering the j_{th} entry of ν, we have $P(|\nu_j| \geq \rho) \leq 2e^{\frac{-2k^2\rho^2}{4kC^2}} \leq 2e^{\frac{-k\rho^2}{2C^2}}$, according to Hoeffding's inequality. Let $\alpha = P(|\nu_j| \geq \rho) = P(\nu_j \notin [-\rho, \rho]) \leq 2e^{\frac{-k\rho^2}{2C^2}}$. Such the probability represents the level of significance, i.e. α for a confidence interval around the expectation of $\dot{g}$ of size 2ρ. Let $\epsilon = \frac{\rho^2}{2C^2}$. If we require at least k sampled instances to acquire $(1-\alpha)$-confidence interval $[-\rho, \rho]$, k should satisfy

$$k \geq -2C^2\frac{\log\frac{\alpha}{2}}{\rho^2} = -\frac{\log\frac{\alpha}{2}}{\rho^2/(2C^2)} = -\frac{\log\frac{\alpha}{2}}{\epsilon} \quad (6)$$

. As illustrated in Algorithm 2, the number of sampled instances, i.e. k is increased linearly, which leads to the decrease of ρ. As a result, the variance between the full gradient and its estimation is decreased.

Assumption 1. *Each a function f_i with $i \in \{1, 2, ..., n\}$ in Equ. 1 is L-Liptchiz continuous, that is, for any two parameters ω_i and ω_j:*

$$f_i(\omega_i) \leq f_i(\omega_j) + \nabla f_i(\omega_j)^{\mathrm{T}}(\omega_i - \omega_j) + \frac{L}{2}\| \omega_i - \omega_j \|^2 \quad (7)$$

.

Assumption 2. *The function F in Equ. 1 is μ-strongly convex, that is, for any two parameters ω_i and ω_j:*

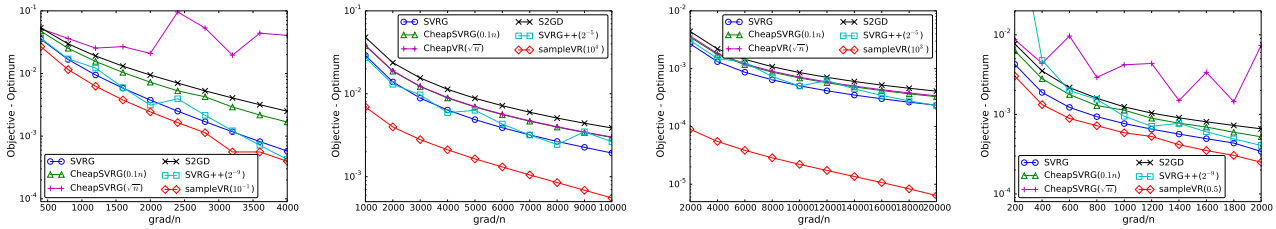
$$F(\omega_i) \geq F(\omega_j) + \nabla F(\omega_j)^{\mathrm{T}}(\omega_i - \omega_j) + \frac{\mu}{2}\| \omega_i - \omega_j \|^2 \quad (8)$$

.

Theorem 2. *ω_* denotes the optimum of the parameter. m can be large enough, so that $\delta = \frac{8L\eta^2 m}{\eta(1-2\eta L)m - \frac{1}{\mu}} < 1$,* SAMPLEVR *makes the optimisation objective converge linearly as: $F(\tilde{\omega}_{s+1}) - F(\omega_*) \leq \delta[F(\tilde{\omega}_s) - F(\omega_*)]$.*

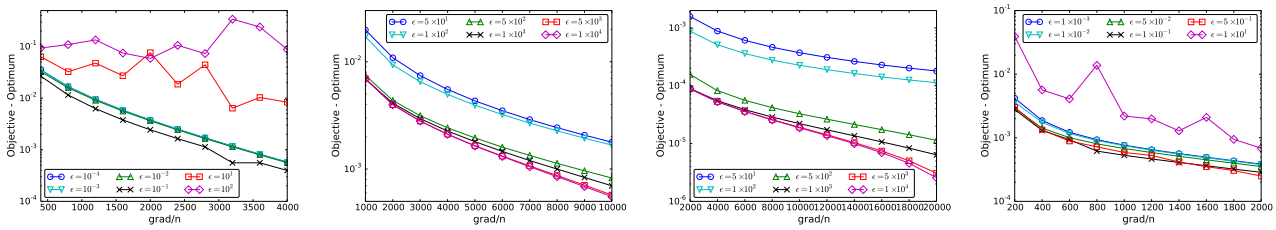
Theorem 3. SAMPLEVR *requires at least $\frac{\ln\zeta}{\ln\delta}m + \left(-\frac{\log\frac{\alpha}{2}}{2\epsilon}(\frac{\ln\zeta}{\ln\delta}+1)(\frac{\ln\zeta}{\ln\delta})\right)$ atomic gradient calculations with $\delta = \frac{8L\eta^2 m}{\eta(1-2\eta L)m - \frac{1}{\mu}}$ to achieve $F(\tilde{\omega}_s) - F(\omega_*) \leq \zeta[F(\omega_0) - F(\omega_*)]$.*

As illustrated in Theorem 2 and Theorem 3, SAMPLEVR makes the optimisation objective converge at a linear rate. To be honest, the convergence performance is not the best when comparing with SVRG and its variants. However, SAMPLEVR has a significant advantage on the gradient complexity according to Theorem 3. For example, when the optimisation objective is strongly convex, and achieves $F(\tilde{\omega}_s) - F(\omega_*) \leq \zeta[F(\omega_0) - F(\omega_*)]$,



(a) ijcnn1 (b) colon-cancer (c) duke-breast-cancer (d) a9a

Figure 1: SAMPLEVR makes the training loss of the $l2$-regularised logistic regression tasks converge faster than the other existing algorithms.



(a) ijcnn1 (b) colon-cancer (c) duke-breast-cancer (d) a9a

Figure 2: Generally, SAMPLEVR with a large ϵ has a better performance for the the $l2$-regularised logistic regression tasks. However, the increase of the variance caused by an extremely large ϵ slows the convergence of the training loss.

the gradient complexity of SVRG and its variants is $O\left(\left(n+\frac{L}{\mu}\right)\left(\frac{\ln\zeta}{\ln\delta}\right)\right)$ with $m=\frac{L}{\mu}$ (Allen-Zhu and Yuan 2015). But, the gradient complexity of SAMPLEVR is $O\left(\left(-\frac{\log\frac{\alpha}{2}}{2\epsilon}(\frac{\ln\zeta}{\ln\delta}+1)+\frac{L}{\mu}\right)\left(\frac{\ln\zeta}{\ln\delta}\right)\right)$ with $m=\frac{L}{\mu}$. Considering $\ln n \ll n$ with $\zeta=\frac{1}{n}$, SAMPLEVR outperforms SVRG and its variants on the gradient complexity obviously.

Performance evaluation

Experimental settings

The existing variants of SGD with the variance reduction technique, including SVRG, S2GD, SVRG++, CHEAPSVRG have been used to conduct the performance evaluation with our proposed algorithm, i.e. SAMPLEVR. The number of sampled instances in CHEAPSVRG is identified as $0.1n$ and $\sqrt{n}$ where n represents the size of the training data. Those algorithms are evaluated on eight datasets, including ijcnn1, colon-cancer, duke-brease-cancer, a9a, mg, cpusmall, yearPredictionMSD, and space-ga. All of those datasets are public on the LibSVM website [1].

First, those algorithms are compared by conducting the $l2$-regularised logistic regression tasks on the datasets: ijcnn1, colon-cancer, duke-breast-cancer, and a9a. When the label of an instance is set to be 1 or -1, the loss function of the $l2$-regularised logistic regression tasks is:

$$\min_{\omega}\frac{1}{n}\sum_{i=1}^{n}\log(1+e^{-y_i\omega^{\mathrm{T}}x_i})+\lambda\parallel\omega\parallel^2 \qquad (9)$$

[1] http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/

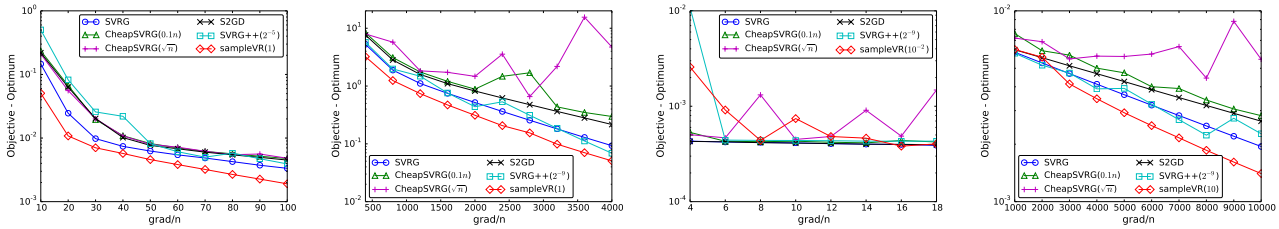
. Second, we compare those algorithms by conducting ridge regression tasks on the other four datasets, i.e. mg, cpusmall, yearPredictionMSD, and space-ga. The loss function of the ridge regression tasks is:

$$\min_{\omega}\frac{1}{n}\sum_{i=1}^{n}\left(\omega^{\mathrm{T}}x_i-y_i\right)^2+\lambda\parallel\omega\parallel^2 \qquad (10)$$

. We set λ to be 10^{-5}, and the learning rate, i.e. η to be 10^{-4} for all the evaluation tests. The epoch size m_s in SVRG and CHEAPSVRG is set to be the size of the training data, i.e. $m_s=n$. The maximal epoch size in S2GD is set to be the size of the training data, i.e. n. The x-axis in all figures of the evaluation tests represents the computational cost. The computational cost is measured by the number of gradient computations divided by the size of the training data, i.e. n. The y-axis in all the figures denotes the training loss residual which is the training loss minus the optimum. Here, the optimum is estimated by running the gradient descent for a long time. The value in the bracket of the legend of SVRG++ and SAMPLEVR represents the initial epoch size divided by the size of the training data, i.e. n and ϵ according to Algorithm 2, respectively.

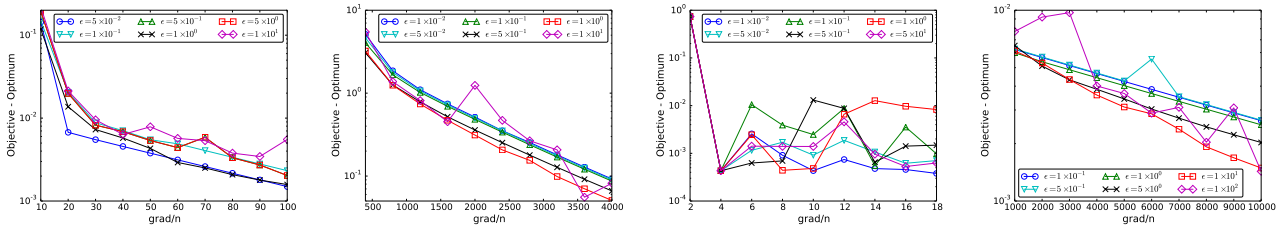
$l2$-regularised logistic regression

As illustrated in Figure 1, we compare the performance of all the algorithms by conducting the $l2$-regularised logistic regression tasks. It is obvious that our proposed algorithm, i.e. SAMPLEVR makes the training loss converge linearly, and outperforms other existing algorithms. The main reason is that SAMPLEVR replaces the computation of the full



(a) mg (b) cpusmall (c) yearPredictionMSD (d) space-ga

Figure 3: Generally, SAMPLEVR outperforms the other existing algorithms on the convergence of the training loss when conducting the ridge regression tasks.



(a) mg (b) cpusmall (c) yearPredictionMSD (d) space-ga

Figure 4: A large ϵ leads to the fast convergence of the training loss for SAMPLEVR when conducting the ridge regression tasks. However, the performance of SAMPLEVR is impaired due to the increase of variance when ϵ is set to be too large.

gradient with an estimation, thus getting rid of the time-consuming calculations of the gradient. Although the estimation of the full gradient is used in CHEAPSVRG, the number of sampled instances in CHEAPSVRG is determined as a constant such as $\sqrt{n}$, leading to much variance. Benefiting from the increase of the number of sampled instances, SAMPLEVR reduces the variance over epochs, and thus outperforms CHEAPSVRG . Additionally, the comparison of the performance of SAMPLEVR by varying ϵ is shown in Figure 2. SAMPLEVR generally obtains a better performance with a larger ϵ. It is because that the number of the required sampled instances becomes small with a large ϵ according to Equ. 6, which leads to the acceleration of SAMPLEVR. However, as illustrated in Figure 2(a) and 2(d), if ϵ is set to be too large, the performance will be impaired due to the variance between the full gradient and the estimation. In specific, a very large ϵ means that the number of the sampled instances becomes extremely small, leading to much variance between the full gradient and the estimation, which slows the convergence of the training loss.

Ridge regression

As illustrated in Figure 3, we report the comparison of the performance by using all the algorithms to conduct ridge regression tasks. SAMPLEVR has a better performance for the datasets mg, cpusmall, and space-ga than the existing algorithms significantly. The main reason is that SAMPLEVR uses an unbiased estimation of the full gradient, instead of costing much time to compute it. Although SAMPLEVR does not outperform other algorithms for the dataset yearPredictionMSD at the beginning of the train process, its performance is comparable to the other algorithms, and finally shows the advantage over most of the existing algorithms. As illustrated in Figure 3(c), the variance of the training loss by using SAMPLEVR is decreased, which benefits from the increase of the sampled instances. As illustrated in Figure 4, the performance of SAMPLEVR has been compared by varying the value of ϵ. It is significant that the variance becomes noticeable with the increase of ϵ. It is because that a large ϵ leads to few sampled instances according to Equ. 6, incurring much variance in the estimation of the full gradient. Moreover, as illustrated in Figure 4(a), 4(b) and 4(d), an extremely small ϵ impairs the performance of SAMPLEVR. The reason is that a small ϵ means that a large number of the instances are required to be sampled, thus incurring much calculations of gradients. A large amount of calculations of gradients cost much time, and thus makes the convergence of the loss function slow down.

Conclusion

This paper proposes a general framework, i.e. EUI to unify the existing variants of SVRG. After that, we explain the reason of the variance reduction technique from a new prospective in the framework. Additionally, we propose a new variant of SGD with the variance reduction technique, i.e. SAMPLEVR which gets rid of computing the full gradient, thus decreasing much computational cost. The theoretical and empirical studies show that SAMPLEVR outperforms the previous work, accelerating the training loss significantly.

References

Allen-Zhu, Z., and Hazan, E. 2016. Variance Reduction for Faster Non-Convex Optimization. In ICML.

Allen-Zhu, Z., and Yuan, Y. 2015. Univr: A universal variance reduction framework for proximal stochastic gradient method. arXiv preprint arXiv:1506.01972.

Allen-Zhu, Z., and Yuan, Y. 2016. Improved SVRG for Non-Strongly-Convex or Sum-of-Non-Convex Objectives. In ICML.

Allen-Zhu, Z. 2016. Katyusha: Accelerated variance reduction for faster sgd. arXiv preprint arXiv:1603.05953.

Defazio, A.; Bach, F.; and Lacoste-Julien, S. 2014. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In Advances in Neural Information Processing Systems, 1646–1654.

Johnson, R., and Zhang, T. 2013. Accelerating stochastic gradient descent using predictive variance reduction. Advances in Neural Information Processing Systems 315–323.

Konečnỳ, J., and Richtárik, P. 2013. Semi-stochastic gradient descent methods. arXiv preprint arXiv:1312.1666.

Konečnỳ, J.; Liu, J.; Richtárik, P.; and Takáč, M. 2016. Mini-batch semi-stochastic gradient descent in the proximal setting. IEEE Journal of Selected Topics in Signal Processing 10(2):242–255.

Li, X.; Zhao, T.; Arora, R.; Liu, H.; and Haupt, J. 2016. Stochastic Variance Reduced Optimization for Nonconvex Sparse Learning. In ICML.

Schmidt, M.; Roux, N. L.; and Bach, F. 2016. Minimizing finite sums with the stochastic average gradient. Mathematical Programming 1–30.

Shah, V.; Asteris, M.; Kyrillidis, A.; and Sanghavi, S. 2016. Trading-off variance and complexity in stochastic gradient descent. arXiv preprint arXiv:1603.06861.

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

Zhang, L.; Mahdavi, M.; and Jin, R. 2013. Linear convergence with condition number independent access of full gradients. In Advances in Neural Information Processing Systems, 980–988.