Bregman Divergence for Stochastic Variance Reduction: Saddle-Point and Adversarial Prediction

Authored by:

Xinhua Zhang
Yaoliang Yu
Zhan Shi

Abstract

Adversarial machines, where a learner competes against an adversary, have regained much recent interest in machine learning. They are naturally in the form of saddle-point optimization, often with separable structure but sometimes also with unmanageably large dimension. In this work we show that adversarial prediction under multivariate losses can be solved much faster than they used to be. We first reduce the problem size exponentially by using appropriate sufficient statistics, and then we adapt the new stochastic variance-reduced algorithm of Balamurugan & Bach (2016) to allow any Bregman divergence. We prove that the same linear rate of convergence is retained and we show that for adversarial prediction using KL-divergence we can further achieve a speedup of #example times compared with the Euclidean alternative. We verify the theoretical findings through extensive experiments on two example applications: adversarial prediction and LPboosting.

1 Paper Body

Many algorithmic advances have been achieved in machine learning by ?nely leveraging the separability in the model. For example, stochastic gradient descent (SGD) algorithms typically exploit the fact that the objective is an expectation of a random function, with each component corresponding to a training example. A ?dual? approach partitions the problem into blocks of coordinates and processes them in a stochastic fashion [1]. Recently, by exploiting the ?nite-sum structure of the model, variance-reduction based stochastic methods have surpassed the well-known sublinear lower bound of SGD. Examples include SVRG [2], SAGA [3], SAG [4], Finito [5], MISO [6], and SDCA [7, 8], just to name a few. Specialized algorithms have also been proposed for accommodating proximal terms [9], and for further acceleration through the condition number

[10?13]. However, not all empirical risks are separable in its plain form, and in many cases dualization is necessary for achieving separability. This leads to a composite saddle-point problem with convexconcave (saddle) functions K and M : n (x? , y ?) = arg minx maxy K(x, y) + M (x, y), where K(x, y) = n1 k=1 ?k (x, y). (1) Most commonly used supervised losses for linear models can be written as g (Xw), where g is the Fenchel dual of a convex function g, X is the design matrix, and w is the model vector. So the regularized risk minimization can be naturally written as minw max? ? Xw + ?(w) ? g(?), where ? is a regularizer. This ?ts into our framework (1) with a bilinear function K and a decoupled function M . Optimization for this speci?c form of saddle-point problems has been extensively studied. For example, [14] and [15] performed batch updates on w and stochastic updates on ?, while [16] and [17] performed doubly stochastic updates on both w and ?, achieving O(1) and O(log 1) rates respectively. The latter two also studied the more general form (1). Our interest in this paper is double stochasticity, aiming to maximally harness the power of separability and stochasticity. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Adversarial machines, where the learner competes against an adversary, have re-gained much recent interest in machine learning [18?20]. On one hand they ?t naturally into the saddle-point optimization framework (1) but on the other hand they are known to be notoriously challenging to solve. The central message of this work is that certain adversarial machines can be solved signi?cantly faster than they used to be. Key to our development is a new extension of the stochastic variance-reduced algorithm in [17] such that it is compatible with any Bregman divergence, hence opening the possibility to largely reduce the quadratic condition number in [17] by better adapting to the underlying geometry using non-Euclidean norms and Bregman divergences. Improving condition numbers by Bregman divergence has long been studied in (stochastic, proximal) gradient descent [21, 22]. The best known algorithm is arguably stochastic mirror descent [23], which was extended to saddle-points by [16] and to ADMM by [24]. However, they can only achieve the sublinear rate O(1/) (for an -accurate solution). On the other hand, many recent stochastic variancereduced methods [2?6, 9, 17] that achieve the much faster linear rate O(log 1/) rely inherently on the Euclidean structure, and their extension to Bregman divergence, although conceptually clear, remains challenging in terms of the analysis. For example, the analysis of [17] relied on the resolvent of monotone operators [25] and is hence restricted to the Euclidean norm. In ?2 we extend the notion of Bregman divergence to saddle functions and we prove a new Pythagorean theorem that may be of independent interest for analyzing ?rst order algorithms. In ?4 we introduce a fundamentally different proof technique (details relegated to Appendix C) that overcomes several challenges arising from a general Bregman divergence (e.g. asymmetry and unbounded gradient on bounded domain), and we recover similar quantitative linear rate of convergence as [17] but with the ?exibility of using suitable Bregman divergences to reduce the condition number. The new stochastic variance-reduced algorithm Breg-SVRG is then applied to the adversarial prediction framework (with multivariate losses such as F-score)

[19, 20]. Here we make three novel contributions: (a) We provide a signi?cant reformulation of the adversarial prediction problem that reduces the dimension of the optimization variable from 2n to n2 (where n is the number of samples), hence making it amenable to stochastic variance-reduced optimization (?3). (b) We develop a new ef?cient algorithm for computing the proximal update with a separable saddle KL-divergence (?5). (c) We verify that Breg-SVRG accelerates its Euclidean alternative by a factor of n in both theory and practice (?6), hence con?rming again the uttermost importance of adapting to the underlying problem geometry. To our best knowledge, this is the ?rst time stochastic variance-reduced methods have been shown with great promise in optimizing adversarial machines. Finally, we mention that we expect our algorithm Breg-SVRG to be useful for solving many other saddle-point problems, and we provide a second example (LPboosting) in experiments (?6).

2

Bregman Divergence and Saddle Functions

In this section we set up some notations, recall some background materials, and extend Bregman divergences to saddle functions, a key notion in our later analysis. Bregman divergence. For any convex and differentiable function ? over some closed convex set C ? Rd , its induced Bregman divergence is de?ned as: ?x ? int(C), x ? C, ?? (x , x) := ?(x) ? ?(x) ? ??(x), x ? x ,

(2)

where ?? is the gradient and ?, ? is the standard inner product in Rd . Clearly, ?? (x , x) ? 0 since ? is convex. We mention two familiar examples of Bregman divergence. 2

1 2

2

x ? x 2 , ?(x) = 12 x 2 , where ? 2 is 2 norm.

x ? (Unnormalized) KL-divergence: ?? (x , x) = i xi log xii ? xi + xi , ?(x) = i xi log xi .

? Squared Euclidean distance: ?? (x , x) =

Strong convexity. Following [26] we call a function f ?-convex if f ?? is convex, i.e. for all x, x f (x) ? f (x) + ?f (x), x ? x + ?? (x , x).

(3)

Smoothness. A function f is L-smooth wrt a norm ? if its gradient ?f is L-Lipschitz continuous, i.e., for all x and x , ?f (x) ? ?f (x) ? ? L x ? x , where ? ? is the dual norm of ? . The change of a smooth function, in terms of its induced Bregman divergence, can be upper bounded by the change of its input and lower bounded by the change of its slope, cf. Lemma 2 in Appendix A. 2

Saddle functions. Recall that a function ?(x, y) over Cz = Cx ? Cy is called a saddle function if it is convex in x for any y ? Cy , and concave in y for any x ? Cx . Given a saddle function ?, we call (x? , y ?) its saddle point if ?(x? , y) ? ?(x? , y ?) ? ?(x, y ?),

?x ? Cx , ?y ? Cy , ?

(4)

?

or equivalently (x , y) ? arg minx?Cx maxy?Cy ?(x, y). Assuming ? is differentiable, we denote G? (x, y) := [?x ?(x, y); ??y ?(x, y)].

(5)

Note the negation sign due to the concavity in y. We can quantify the notion of ?saddle?: A function f (x, y) is called ?-saddle iff f ? ? is a saddle function, or equivalently, ?f (z , z) ? ?? (z , z) (see below). Note that any saddle function ? is 0-saddle and ?-saddle. Bregman divergence for saddle functions. We now de?ne the Bregman divergence induced by a saddle function ?: for z = (x, y) and z = (x , y) in Cz , ?? (z , z) := ??y (x , x) + ???x (y , y) = ?(x , y) ? ?(x, y) ? G? (z), z ? z ,

(6)

where ?y (x) = ?(x, y) is a convex function of x for any ?xed y, and similarly ?x (y) = ?(x, y) is a concave (hence the negation) function of y for any ?xed x. The similarity between (6) and the usual Bregman divergence ?? in (2) is apparent. However, ? is never evaluated at z but z (for G) and the cross pairs (x , y) and (x, y). Key to our subsequent analysis is the following lemma that extends a result of [27] to saddle functions (proof in Appendix A). Lemma 1. Let f and g be ?-saddle and ?-saddle respectively, with one of them being differentiable. Then, for any z = (x, y) and any saddle point (if exists) z ? := (x? , y ?) ? arg minx maxy {f (z) + g(z)} , we have f (x, y ?)+g(x, y ?) ? f (x? , y)+g(x? , y)+??+? (z, z ?). Geometry of norms. In the sequel, we will design two convex functions ?x (x) and ?y (y) such that their induced Bregman divergences are ?distance enforcing? (a.k.a. 1-strongly convex), that is, w.r.t. two norms ? x and ? y that we also design, the following inequality holds: ?x (x, x) := ??x (x, x) ?

1 2

2

x ? x x , ?y (y, y) := ??y (y, y) ?

1 2

2

y ? y y .

(7)

Further, for z = (x, y), we de?ne ?z (z, z) := ??x ??y (z, z) ?

1 2

2

z ? z z ,

where

2

2

2

z z := x x + y y

(8)

When it is clear from the context, we simply omit the subscripts and write ?, ? , and ? ? .

3

Adversarial Prediction under Multivariate Loss

A number of saddle-point based machine learning problems have been listed in [17]. Here we give another example (adversarial prediction under multivariate loss) that is naturally formulated as a saddle-point problem but also requires a careful adaptation to the underlying geometry?a challenge that was not addressed in [17] since their algorithm inherently relies on the Euclidean norm. We remark that adaptation to the underlying geometry has been studied in the (stochastic) mirror descent framework [23], with signi?cant improvements on condition numbers or gradient norm bounds. Surprisingly, no analogous efforts have been attempted in the stochastic variance reduction framework?a gap we intend to ?ll in this work. The adversarial prediction framework [19, 20, 28], arising naturally as a saddle-point problem, is a convex alternative to the generative adversarial net [18]. Given a training sample X = [x1 , . . . , xn] ? = [? and y y1 , . . . , y?n] ? {0, 1}n , adversarial prediction optimizes the following saddle function that is an expectation of some multivariate loss (y, z) (e.g. F-score) over the labels y, z ? {0, 1}n of all data points:

? minn maxn (9) E (y, z), s.t. E (n1 Xz) = n1 X y p??2

q??2

y?p,z?q

z?q

Here the proponent tries to ?nd a distribution p(?) over the labeling on the entire training set in n order to minimize the loss (?2 is the 2n dimensional probability simplex). An opponent in contrast tries to maximize the expected loss by ?nding another distribution q(?), but his strategy is subject to the constraint that the feature expectation matches that of the empirical distribution. Introducing a 3

Lagrangian variable ? to remove the feature expectation constraint and specializing the problem to

z F-score where (y, z) = 12y y+1 z and (0, 0) := 1, the partial dual problem can be written as

2 2y z 1 ? + minn maxn E max ? ?2 ? 2 + n1 ? X y (10) 1 y+1 z ? n ? Xy , ?

p??2

q??2

y?p,z?q

where we use y z to denote the standard inner product and we followed [19] to add an 22 regularizer on ? penalizing the dual variables on the constraints over the training data. It appears that solving (10) can be quite challenging, because the variables p and q in the inner minimax problem have 2n entries! A constraint sampling algorithm was adopted in [19] to address this challenge, although no formal guarantee was established. Note that we can maximize the outer unconstrained variable ? (with dimension the same as the number of features) relatively easily using for instance gradient ascent, provided that we can solve the inner minimax problem quickly?a signi?cant challenge to which we turn our attention below. Surprisingly, we show here that the inner minimax problem in (10) can be signi?cantly simpli?ed. The key observation is that the expectation in the objective depends only on a few suf?cient statistics of p and

q. Indeed, by interpreting p and q as probability distributions over {0, 1}n we have: n n

2y z 2y z

E = p({0})q({0}) + E [[1 y = i]][[1 z = j]] (11)

1 y+1 z 1 y + 1 z i=1 j=1 = p({0})q({0}) +

n n

2ij 1 1 ? E (y[[1 y = i]]) ? E (z[[1 z = j]]), i + j i

j i=1 j=1

?i

(12)

?j

where [[?]] = 1 if ? is true, and 0 otherwise. Crucially, the variables ?i and ?j are suf?cient for re-expressing (10), since 1 (13) 1 ?i = E (1 y[[1 y = i]]) = E[[1 y = i]] = p({1 y = i}), i

i?i = E (y[[1 y = i]]) = Ey, (14) i

i

and similar equalities also hold for ?j . In details, the inner minimax problem of (10) simpli?es to: n n

1 2ijn2 2

min max i+j ?i ?j +n ?i 11 ?j ?n1 ?i ?n1 ?j ?? Xi?i +?(?)??(?), (15) ??S ??S n2

i=1 j=1 fij (?i ,?j)

where S = {? ? 0 : 1 ? ? 1, ?i, i?i ? ? ?i 1 },

?(?) = ?

i,j

?ij log(?ij). (16)

2

Importantly, ? = [?1 ; . . . , ?n] (resp. ?) has n entries, which is signi?cantly smaller than the 2n entries of p (resp. q) in (10). For later purpose we have also incorporated an entropy regularizer for ? and ? respectively in (15). To justify the constraint set S, note from (12) and (13) that for any distribution p of y: since ? ? 0 and y ? {0, 1}n , i?i ? ? E y[[1 y = i]] ? ? E[[1 y = i]] = ?i 1 . (17)

Conversely, for any ? ? S, we can construct a distribution p such that i?ij = E (yj [[1 y = i]]) = p({1 y = i, yj = 1}) in the following algorithmic way: Fix i and for each j de?ne Yj = {y ? {0, 1}n : 1 y = i, yj = 1}. Let U = {1, . . . , n}. Find an index j in U that minimizes ?ij and set p({y}) = i?ij /—Yj — for each y ? Yj . Perform the following updates: (18) U ? U {j}, ?k = j, Yk ? Yk Yj , ?ik ? ?ik ? ?ij —Yk ? Yj —/—Yj — Continue this procedure until U is empty. Due to the way we choose j, ? remains nonnegative and by construction ?ij = p({1 y = i, yj = 1}) once we remove j from U . The objective function in (15) ?ts naturally into the framework of (1), with ?(?) ? ?(?) and constraints corresponding to M , and the rest terms to K. The entropy function ? is convex wrt the KL-divergence, which is in turn distance enforcing wrt the 1 norm over the probability simplex [23]. In the next section we propose the SVRG algorithm with Bregman divergence (Breg-SVRG) that (a) provably optimizes strongly convex saddle function with a linear convergence rate, and (b) adapts to the underlying geometry by choosing an appropriate Bregman divergence. Then, in ?5 we apply Breg-SVRG to (15) and achieve a factor of n speedup over a straightforward instantiation of [17]. 4

4

Breg-SVRG for Saddle-Point

In this section we propose an ef?cient algorithm for solving the general saddle-point problem in (1) and prove its linear rate of convergence. Our main assumption is: Assumption 1. There exist two norms ? x and ? y such that each ?k is a saddle function and L-smooth; M is (?x , ?y)-saddle; and ?x and ?y are distance enforcing (cf. (7)).

Algorithm 1: Breg-SVRG for Saddle-Point 1 Initialize z0 randomly. Set z ? = z0 . 2 for s = 1, 2, . . . do epoch index 3 ? ??? ?s := ?K(? z), z0 ? z0s := zm 4 for t = 1, . . . , m do iter index 5 Randomly pick ? ? {1, . . . , n}. 6 Compute vt using (20). 7 Update zt using (21).

m m

8 z? ? z?s := (1 + ?)t zt (1 + ?)t . t=1

t=1

Note that w.l.o.g. we have scaled the norms so that the usual strong convexity parameter of M is 1. Recall we de?ned z z and ?z in (8). For saddle-point optimization, it is common to de?ne a signed gradient G(z) := [?x K(z); ??y K(z)] (since K is concave in y). Recall J = K + M , and (x? , y ?) is a saddle-point of J. Using Assumption 1, we measure the gap of an iterate zt = (xt , yt) as follows: t = (zt) = J(xt , y ?) ? J(x? , yt) ? ?(zt , z ?) ?

1 2

2

zt ? z ? ? 0.

(19)

Inspired by [2, 9, 17], we propose in Algorithm 1 a new stochastic variance-reduced algorithm for solving the saddle-point problem (1) using Bregman divergences. The algorithm proceeds in epochs. In each epoch, we ?rst compute the following stochastic estimate of the signed gradient G(zt) by drawing a random component from K:

z) + ?x K(? z) vx (zt) := ?x ?? (zt) ? ?x ?? (? vx (zt) vt = . (20) where ?vy (zt) vy (zt) := ?y ?? (zt) ? ?y ?? (? z) + ?y K(? z) Here z? is the pivot chosen after completing the previous epoch. We make two important observations: (1) By construction the stochastic gradient vt is unbiased: E? [vt] = G(zt); (2) The expensive gradient evaluation ?K(? z) need only be computed once in each epoch since z? is held unchanged. If z? ? z ? , then the variance of vt would be largely reduced hence faster convergence may be possible. Next, Algorithm 1 performs the following joint proximal update: (xt+1 , yt+1) = arg min max ? vx (zt), x + ? vy (zt), y + ?M (x, y) + ?(x, xt) ? ?(y, yt), (21) x

y

where we have the ?exibility in choosing a suitable Bregman divergence to better adapt to the underlying geometry. When ?(x, xt) = 12 x ? xt 22 , we recover the special case in [17]. However, to handle the asymmetry in a general Bregman divergence (which does not appear for the Euclidean distance), we have to choose the pivot z? in a signi?cantly different way than [2, 9, 17]. We are now ready to present our main convergence guarantee for Breg-SVRG in Algorithm 1. Theorem 1. Let Assumption 1 hold, and choose a suf?ciently small ? ¿ 0 such that m :=

1??L ???1 /log(1 + ?) ? 1. Then Breg-SVRG enjoys linear convergence in expectation: log 18?L 2 E(? z s) ? (1 + ?)?ms [?(z ? , z0) + c(Z + 1)(z0)], where Z =

m?1 t=0

(1+?)t , c =

18? 2 L2 1??L .

(22)

2 1 2 For example, we may set ? = 45L , (1 + ?)m ? 64 2 , which leads to c = O(1/L), m = ? L 45 , and Z = O(L2). Therefore, between epochs, the gap (? z s) decays (in expectation) by a factor of 45 64 , 2 and each epoch needs to conduct the proximal update (21) for m = ?(L) number of times. (We remind that w.l.o.g. we have scaled the norms so that the usual strong convexity parameter is 1.) In total, to reduce the gap below some threshold , Breg-SVRG needs to call the proximal update (21) O(L2 log 1) number of times, plus a similar number of component gradient evaluations. Discussions. As mentioned, Algorithm 1 and Theorem 1 extend those in [17] which in turn extend [2, 9] to saddle-point problems. However, [2, 9, 17] all heavily exploit the Euclidean structure (in particular symmetry) hence their proofs cannot be applied to an asymmetric Bregman divergence. Our innovations here include: (a) A new Pythagorean theorem for the newly introduced saddle Bregman divergence (Lemma 1). (b) A moderate extension of the variance reduction lemma in [9] to accommodate any norm (Appendix B). (c) A different pivot z? is adopted in each epoch to handle 5

asymmetry. (d) A new analysis technique through introducing a crucial auxiliary variable that enables us to bound the function gap directly. See our proof in Appendix C for more details. Compared with classical mirror descent algorithms [16, 23] that can also solve saddle-point problems with Bregman divergences, our analysis is fundamentally different and we achieve the signi?cantly stronger rate O(log(1/) than the sublinear O(1/) rate of [16], at the expense of a squared instead of linear dependence on L. Similar tradeoff also appeared in [17]. We will return to this issue in Section 5. Variants and acceleration. Our analysis also supports to use different ? in vx and vy . The standard acceleration methods such as universal catalyst [10] and non-uniform sampling can be applied directly (see Appendix E where L, the largest smoothness constant over all pieces, is replaced by their mean).

5

Application of Breg-SVRG to Adversarial Prediction

The quadratic dependence on L, the smoothness parameter, in Theorem 1 reinforces the need to choose suitable Bregman divergences. In this section we illustrate how this can be achieved for the adversarial prediction problem in

Section 3. As pointed out in [17], the factorization of K is important, n n and we consider three schemes: (a) ?k = fij ; (b) ?k = n1 j=1 fk,j ; and (c) ?k = n1 i=1 fi,k . W.l.o.g. let us ?x the ? in (16) to 1. Comparison of smoothness constant. Both ? and ? are n2 -dimensional, and the bilinear function 2 2 fij can be written as ? Aij ?, where Aij ? Rn ?n is an n-by-n block matrix, with the (i, j)-th 2ij block being n2 (i+j I + 11) and all other blocks being 0. The linear terms in (15) can be absorbed into the regularizer ? without affecting the smoothness parameter. For scheme (a), the smoothness constant L2 under 2 norm depends on the spectral norm of Aij : 2ij L2 = maxi,j n2 (n + i+j)) = ?(n3). In contrast the smoothness constant L1 under 1 norm depends 2ij) = ?(n3); no saving is achieved. on the absolute value of the entries in Aij : L1 = maxi,j n2 (1 + i+j n For scheme (b), the bilinear function ?k corresponds to n1 ? j=1 Akj ?. Then L1 = O(n2) while n n

1 2 Akj v2 ? n2 max L22 = 2 max max 11 v = n5 . (23) 2 j=1 j=1 n k v:v2 =1 v2 =1 Therefore, L21 saves a factor of n compared with L22 . Comparison of smoothness constant for the overall problem. By strong duality, we may push the maximization over ? to the innermost level of (10), arriving at an overall problem in ? and ? only:

n n 1 i ij 1 2 min max c X?i + ? X X?j + c 2 . fij (?i , ?j) ? (24) ?n 2? i 2?n2 {?i }?S {?j }?S n2 i=1 j=1 2

2

? . The quadratic term w.r.t. ? can be written as ? Bij ?, where Bij ? Rn ?n is an where c = X y ij n-by-n block matrix, with its (i, j)-th block being 2? X X and all other blocks being 0. And we assume each xi 2 ? 1. The smoothness constant can be bounded separately from Aij and Bij ; see (128) in Appendix F. For scheme (a), the smoothness constant square L22 under 2 norm is upper bounded by the sum of ij 2 spectral norm square of Aij and Bij . So L22 ? maxi,j 2? n = ?(n6), i.e. L2 = ?(n3). In contrast 2 the smoothness constant square L1 under 1 norm is at most the sum of square of maximum absolute 2

ij 2 2ij value of the entries in Aij and Bij . Hence L21 ? maxi,j n2 (1+ i+j) + maxi,j 2? = ?(n6), i.e. L1 = ?(n3). So no saving is achieved here. n n For scheme (b), ?k corresponds to n1 (? j=1 Akj ? + ? j=1 Bkj ?). Then

2 2 n n 1 2

L1 ? 2 max max Akj v ? + max Bkj v ? (by (128)) j=1 j=1 n k v:v1 =1 v:v1 =1

2 2 1 2kj) + kj ? 2 max max n2 (1+ k+j = n4 , 2 j n k

(25) (26)

and by setting ? to 0 in (126), we get L22 ? n5 similar to (23). Therefore, L21 saves a factor of n compared with L22 . Similar results apply to scheme (c) too. We also tried non-uniform sampling, but 6

it does not change the order in n. It can also be shown that if our scheme randomly samples n entries from {Aij , Bij }, the above L1 and L2 cannot be improved by further engineering the factorization. Computational complexity. We ?nally seek ef?cient algorithms for the proximal update (21) used by Breg-SVRG. When M (?, ?) = ?(?) ? ?(?) as in (16), we can solve ? and ? separately as:

?ik log(?ik /bik) ? cik , s.t. 1 ? ? 1, ?i ?k, 0 ? i?ik ? 1 ?i . (27) min ?
ik

where bik and cik are constants. In Appendix D we designe an ef?cient ?closed form? algorithm which ?nds an accurate solution in O(n2 log2 1) time, which is also on par with that for computing the stochastic gradient in schemes (b) and (c). Although scheme (a) reduces the cost of gradient computation to O(n), its corresponding smoothness parameter L21 is increased by n2 times, hence ? not worthwhile. We did manage to design an O(n) algorithm for the proximal update in scheme (a), but empirically the overall convergence is rather slow. 2

If we use the Euclidean squared distance as the Bregman divergence, then a term ? ? ?t 2 needs to be added to the objective (27). No ef?cient ?closed form? solution is available, and so in experiments we simply absorbed M into K, and then the proximal update becomes the Euclidean projection onto S, which does admit a competitive O(n2 log2 (1/)) time solution.

6

Experimental Results

Our major goal here is to show that empirically Entropy-SVRG (Breg-SVRG with KL divergence) is signi?cantly more ef?cient than Euclidean-SVRG (Breg-SVRG with squared Euclidean distance) on some learning problems, especially those with an entropic regularizer and a simplex constraint. 6.1

Entropy regularized LPBoost

We applied Breg-SVRG to an extension of LP Boosting using entropy regularization [29]. In a binary classi?cation setting, the base hypotheses over the training set can be compactly represented as U = (y1 x1 , . . . , yn xn) . Then the model considers a minimax game between a distribution d ? ?n over training examples and a distribution w ? ?m over the hypotheses: min

d??n ,di ??

max d U w + ??(d) ? ??(w).

w??m

(28)

Here w tries to combine the hypotheses to maximize the edge (prediction con?dence) yi xi w, while the adversary d tries to place more weights (bounded by ?) on ?hard? examples to reduce the edge. Settings. We experimented on the adult dataset from the UCI repository, which we partitioned into n = 32, 561 training examples and 16,281 test examples, with m = 123 features. We set ? = ? = 0.01 and ? = 0.1 due to its best prediction accuracy. We tried a range of values of the step size ?, and the best we found was 10?3 for Entropy-SVRG and 10?6 for Euclidean-SVRG (larger step size for Euclidean-SVRG ?uctuated even worse). For both methods, m = 32561/50 gave good results. The stochastic gradient in d was computed by U:j wj , where U:j is the j-th column and j is randomly sampled. The stochastic gradient in w is di Ui: . We tried with Uij wj and Uij di (scheme (a) in ?5), but they performed worse. We also tried with the universal catalyst in the same form as [17], which can be directly extended to Entropy-SVRG. Similarly we used the non-uniform sampling based on the 2 norm of the rows and columns of U . It turned out that the Euclidean-SVRG can bene?t slightly from it, while Entropy-SVRG does not. So we only show the ?accelerated? results for the former. To make the computational cost comparable across machines, we introduced a counter called effective number of passes: #pass. Assume the proximal operator has been called #po number of times, then #pass := number of epochs so far +

n+m nm

? #po.

(29)

We also compared with a ?convex? approach. Given d, the optimal w in (28) obviously admits a closed-form solution. General saddle-point problems certainly do not enjoy such a convenience. However, we hope to take advantage of this opportunity to study the following question: suppose we solve (28) as a convex optimization in d and the stochastic gradient were computed from the optimal 7

10-2 10

-3

10 0

Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst

10 -2

10 0

Primal gap

Primal gap

10-1

Entropy, Saddle Entropy, Convex Euclidean, Saddle Euclidean, Convex

Primal gap

100

Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst

10 -2

10-4 -5

0

10 -4

200 400 600 800 Number of effective passes

81 79 77 75 0

200 400 600 800 Number of effective passes

(b) Test accuracy v.s. #pass

Figure 1: Entropy Regularized LPBoost on adult

5

10 15 CPU time(mins)

20

0.96

0.94 Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst

0.92

0.9

0
(b) Primal gap v.s. CPU time
0.96 Test F-score
Test accuracy (%)
85 Entropy, Saddle Entropy, Convex Euclidean, Saddle Euclidean, Convex
10 -4
100 200 300 400 Number of effective passes
(a) Primal gap v.s. #pass
(a) Primal gap v.s. #pass
83
0
0
100 200 300 Number of effective passes
(c) Test F-score v.s. #pass
Test F-score
10
0.94 Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst
0.92
0.9
0
5 10 CPU time(mins)
15
(d) Test F-score v.s. CPU time
Figure 2: Adversarial Prediction on the synthetic dataset.

w, would it be faster than the saddle SVRG? Since solving w requires visiting the entire U , strictly speaking the term n+m nm ?#po in the de?nition of #pass in (29) should be replaced by #po. However, we stuck with (29) because our interest is whether a more accurate stochastic gradient in d (based on the optimal w) can outperform doubly stochastic (saddle) optimization. We emphasize that this comparison is only for conceptual understanding, because generally optimizing the inner variable requires costly iterative methods. Results. Figure 1(a) demonstrated how fast the primal gap (with w optimized out for each d) is reduced as a function of the number of effective passes. Methods based on entropic prox are clearly much more ef?cient than Euclidean prox. This corroborates our theory that for problems like (28), Entropy-SVRG is more suitable for the underlying geometry (entropic regularizer with simplex constraints). We also observed that using entropic prox, our doubly stochastic method is as ef?cient as the ?convex? method, meaning that although at each iteration the w in saddle SVRG is not the optimal for the current d, it still allows the overall algorithm to perform as fast as if it were. This suggests that for general saddle-point problems where no closed-form inner solution is available, our method will still be ef?cient and competitive. Note this ?convex? method is similar to the optimizer used by [29]. Finally, we investigated the increase of test accuracy as more passes over the data are performed. Figure 1(b) shows, once more, that the entropic prox does allow the accuracy to be improved much faster than Euclidean prox. Again, the convex and saddle methods perform similarly. As a ?nal note, the Euclidean/entropic proximal operator for both d and w can be solved in either closed form, or by a 1-D line search based on partial Lagrangian. So their computational cost differ in the same order of magnitude as multiplication v.s. exponentiation, which is much smaller than the difference of #pass shown in Figure 1. 6.2

Adversarial prediction with F-score

Datasets. Here we considered two datasets. The ?rst is a synthetic dataset where the positive examples are drawn from a 200 dimensional normal distribution with mean 0.1 ? 1 and covariance 0.5 ? I, and negative examples are drawn from N (?0.1 ? 1, 0.5 ? I). The training set has n = 100 samples, half are positive and half are negative. The test set has 200 samples with the same class ratio. Notice that n = 100 means we are optimizing over two 100-by-100 matrices constrained to a challenging set S. So the optimization problem is indeed not trivial. 8

10 0 10 -2 10 -4

0

10 2

Primal gap

Primal gap

Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst

(a) Primal gap v.s. #pass

10 -2

0

20 40 60 CPU time(mins)

80

(b) Primal gap v.s. CPU time

0.9

0.9

0.85 Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst

0.8 0.75

Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst

10 0

10 -4

200 400 600 Number of effective passes

0

50 100 150 200 Number of effective passes

(c) Test F-score v.s. #pass

Test F-score

Methods. To apply saddle SVRG, we used strong duality to push the optimization over ? to the inner-most level of (10), and then eliminated ? because it is a simple quadratic. So we ended up with the convexconcave optimization as shown in (24), where the K part of (15) is augmented with a quadratic term in ?. The formulae for computing the stochastic gradient using scheme (b) are detailed in Appendix G. We ?xed ? = 1, ? = 0.01 for the ionosphere dataset, and ? = 1, ? = 0.1 for the synthetic dataset.

10 2

Test F-score

The second dataset, ionosphere, has 211 training examples (122 pos and 89 neg). 89 examples were used for testing (52 pos and 37 neg). Each example has 34 features.

0.85 Euclidean, Convex Euclidean, Saddle Euclidean, Saddle, Catalyst Entropy, Convex Entropy, Saddle Entropy, Saddle, Catalyst

0.8 0.75

0

10 20 CPU time(mins)

30

(d) Test F-score v.s. CPU time

Figure 3: Adversarial Prediction on the ionosphere dataset. We also tried the universal catalyst along with non-uniform sampling where each i was samn 2 pled with a probability proportional to k=1 Aik F , and similarly for j. Here ? F is the Frobenious norm. Parameter Tuning. Since each entry in the n ? n matrix ? is relatively small when n is large, we needed a relatively small step size. When n = 100, we used 10?2 for Entropy-SVRG and 10?6 for Euclidean-SVRG (a larger step size makes it over-?uctuate). When applying catalyst, the catalyst regularizor can suppress the noise from larger step size. After a careful trade off between catalyst regularizor parameter and larger step size, we managed to achieve faster convergence empirically. Results. The results on the two datasets are shown in Figures 2 and 3 respectively. We truncated the #pass and CPU time in subplots (c) and (d) because the F-score has stabilized and we would rather zoom in to see the initial growing phase. In terms of primal gap versus #pass (subplot a), the entropy based method is signi?cantly more effective than Euclidean methods on both datasets (Figure 2(a) and 3(a)). Even with catalyst, Euclidean-Saddle is still much slower than the entropy based methods on the synthetic dataset in Figure 2(a). The CPU time comparisons (subplot b) follow the similar trend, except that the ?convex methods? should be ignored because they are introduced only to compare #pass. The F-score is noisy because, as is well known, it is not monotonic with the primal gap and glitches can appear. In subplots 2(d) and 3(d), the entropy based methods achieve higher F-score signi?cantly faster than the plain Euclidean based methods on both datasets. In terms of passes (subplots 2(c) and 3(c)), Euclidean-Saddle and Entropy-Saddle achieved a similar F-score at ?rst because their primal gaps are comparable at the beginning. After 20 passes, the F-score of Euclidean-Saddle is overtaken by Entropy-Saddle as the primal gap of Entropy-Saddle become much smaller than Euclidean-Saddle.

7

Conclusions and Future Work

We have proposed Breg-SVRG to solve saddle-point optimization and proved its linear rate of convergence. Application to adversarial prediction con?rmed its effectiveness. For future work, we are interested in relaxing the (potentially hard) proximal update in (21). We will also derive similar reformulations for DCG and precision@k, with a quadratic number of variables and with a ?nite sum structure that is again amenable to Breg-SVRG, leading to a similar reduction of the condition number compared to Euclidean-SVRG. These reformulations, however, come with different constraint sets, and new proximal algorithms with similar complexity as for the F-score can be developed. 9

2 References

[1] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate frank-wolfe optimization for structural SVMs. In ICML. 2013. [2] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In NIPS. 2013. [3] A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In NIPS. 2014. [4] M. Schmidt, N. L. Roux, and F. Bach. Minimizing ?nite sums with the stochastic average gradient. Mathematical Programming, 2016. [5] A. J. Defazio, T. S. Caetano, and J. Domke. Finito: A faster, permutable incremental gradient method for big data problems. In ICML. 2014. [6] J. Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. SIAM Journal on Optimization, 25(2):829?855, 2015. [7] S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. J. Mach. Learn. Res., 14:567?599, 2013. [8] S. Shalev-Shwartz. SDCA without duality, regularization, and individual convexity. In ICML. 2016. [9] L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. SIAM Journal on Optimization, 24(4):2057?2075, 2014. [10] H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for ?rst-order optimization. In NIPS. 2015. [11] A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. In NIPS. 2014. [12] S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In ICML. 2014. [13] R. Babanezhad, M. O. Ahmed, A. Virani, M. Schmidt, J. Kone?cn?y, and S. Sallinen. Stop wasting my gradients: Practical svrg. In NIPS. 2015. [14] Y. Zhang and L. Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. In ICML. 2015. [15] Z. Zhu and A. J. Storkey. Adaptive stochastic primal-dual coordinate descent for separable saddle point problems. In Machine Learning and Knowledge Discovery in Databases, pp. 645?658. 2015. [16] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. SIAM Journal on Optimization, 19(4):1574?1609, 2009. [17] P. Balamurugan and F. Bach. Stochastic variance reduction methods for saddle-point problems. In NIPS. 2016. [18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NIPS. 2014. [19] H. Wang, W. Xing, K. Asif, and B. D. Ziebart. Adversarial prediction games for multivariate losses. In NIPS. 2015.

[20] F. Farnia and D. Tse. A minimax approach to supervised learning. In NIPS. 2016. [21] Y. Nesterov. Smooth minimization of non-smooth functions. Mathematical Programming, 103(1):127?152, 2005. [22] Y. Nesterov. Excessive gap technique in nonsmooth convex minimization. SIAM J. on Optimization, 16(1):235?249, 2005. ISSN 1052-6234. [23] A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. Operations Research Letters, 31(3):167?175, 2003. [24] H. Wang and A. Banerjee. Bregman alternating direction method of multipliers. In NIPS. 2014. [25] R. T. Rockafellar. Monotone operators associated with saddle functions and minimax problems. Nonlinear Functional Analysis, 18(part 1):397?407, 1970. 10

[26] J. C. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite objective mirror descent. In Proc. Annual Conf. Computational Learning Theory. 2010. [27] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. submitted to SIAM Journal on Optimization, 2009. [28] K. Asif, W. Xing, S. Behpour, and B. D. Ziebart. Adversarial cost-sensitive classi?cation. In UAI. 2015. [29] M. K. Warmuth, K. A. Glocer, and S. V. N. Vishwanathan. Entropy regularized LPBoost. In Y. Freund, Y. L. Gy?or?, and G. Tur'an, eds., Proc. Intl. Conf. Algorithmic Learning Theory, no. 5254 in Lecture Notes in Arti?cial Intelligence, pp. 256 ? 271. Springer-Verlag, Budapest, October 2008. [30] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. Fixed-Point Algorithms for Inverse Problems in Science and Engineering, 49:185?212, 2011.

11