Create a Stochastic Average Gradient R package

Eric Xin Zhou

March 16, 2015

Eric Xin Zhou(xinchoubiology)

PhD candidate of Electrical and Computer Engineering

University of Miami,

Coral Gables, Florida

United States

Email: xxz220@miami.edu

1 Summary

Schmidt et al. [3] proposed the Stochastic Average Gradient(SAG) algorithm as a faster solver for optimizing the sum of finite number of smooth convex functions. Our project aims to create a SAG R package of optimizer solver which applies SAG algorithm to several optimization problems.

2 Motivation

In large-scale machine learning, there are plethora of optimization problems based on empirical risk minimization principle in statistical learning theory. This class of problems involves computing a minimizer of a finite sum of a set of smooth functions. Assuming we have N data points and a convex loss function that can be decomposed into a sum of N convex cost function, this loss function is called sum structure function. Then, our goal is to minimize the loss function with respect to the optimization coefficient variables.

Stochastic gradient method is known as a successful algorithm that is widely applied to minimize the sum structure problem, but its convergence rate is sub-linear. To improve this class of optimization problems' convergence rate, Schmidt et al. [3] proposed the Stochastic Average Gradient(SAG) algorithm, which achieves linear coverage rate when the summed function is strongly-convex[2, 3]. Schmidt et al. [3] also compared the SAG with known FG and SG methods on L-2 regularized logistic regression problem, such as L-BGFS,

AFG and SG. Finally, they find that SAG achieve the best speed to converge to optimum. Therefore, Schmidt et al. [3] claims that their SAG achieved the best performance on classification data set.

There are many packages that have been created to help R users handle big data these years, and accelerating the classification problems on big data is still an important task for R comunity. Therefore, we expect the SAG to be faster than existing R packages, such as glmnet.

3 Project Description

3.1 Create SAG R package

SAG R package will contains three modified solvers based on Schmidt et al. [3]'s C code with 'mex.h' of SAG, then we will implement these SAG solvers on 2 practical problems, L-2 regularized logistic regression problem and L-2 regularized conditional random field(CRF) training.

3.2 SAG applied to logistic regression

When each data point is a feature vector with binary label, we can use L-2 regularized logistic regression model to predict binary responses from their feature vectors. Since L-2 regularized logistic regression model is a sum structure function[1], we will implement the solvers of SAG R package for L-2 regularized logistic regression. SAG can update parameters by randomly selecting a summed function from the entire observations.

Then we will compare the training time measured using the `system.time` function and the result of the SAG solver with the L-2 regularized logistic regression in package glmnet[1]/optimx.

3.3 SAG applied to CRF

Also, Schmidt et al. [4] claimed that SAG can be implemented to train the conditional random fields(CRF).

CRF is an undirected graph model for observations and latent variables. For example, in a chain graph model of word recognition, we can observe the different image patterns of the letters that constitute the entire word. If we wish to infer the hidden state variable(real letter of word) of each given observation, we will assume that these hidden letter states have certain structure to form a chain or graph, and the image of each letter depends its corresponding hidden state . As the result, we can define a graph for these observations and their corresponding hidden states, then we define each edge in this graph by a *feature functions*.

[1]Please read section 2.2 Intro2Lr_and_MRF for more mathematical details

Assuming that we have N words(or observations) for the CRF, and their liklihood functions can be decomposed into a sum of N liklihood functions[2]. Then, SAG can be applied to maximize the CRF's liklihood function in the CRF's training process.

Our implementation of SAG R package will contain a solver for training the L-2 regularize CRF model. Then we will compare traning time measured by `system.time` function and estimate parameters' value of our CRF training function and the training function of R package CRF.

4 Roadmap

4.1 Relate tasks

- Mark Schmidt has published C/matlab code implement SAG for L2 regularization logistic regression and least-square regression. Therefore, out first goal is to convert Mark's C code with 'mex.h' headers to C code with 'R.h' headers by Rcpp for the three main SAG methods(SAG.c, SAGlinesearch.c and SAG_LipschitzLS.c).

- Converting Mark's documentation comments in C code to .Rd file.

- Creating vignettes using Mark's two example data sets `rcv1_train` and `covtype.libsvm` data sets. In these vignettes, we will show these 3 solvers work well and we will compare time and result of SAG with result of function `glmnet` in glmnet.

```
fit <- glmnet(X, y, family = "binomial",
                    lambda = lambda, alpha = 0)
par.est <- coef(fit)
system.time(glmnet(X, y, family = "binomial",
                         lambda = lambda, alpha = 0))
```

  Also, we will compare SAG solvers to the function optimx in package optimx. And then, we will compare SAG's result and time to results of function `glmnet` and `optimx`.

```
lr.f <- function(theta, lambda = 0.1){
  fval <- lambda * crossprod(theta)[1] / 2
          + sum(log(1 + exp(-y * X %*% theta))) / dim(X)[1]
  return(fval)
}

par.init <- runif(dim(X)[2],min = 0, max = 0.001)
```

[2]Please read section 2.3 Intro2Lr_and_MRF for more mathematical details of conditional random field

```
fit <- optimx(par.init, fn = lr.f, method = "CG")
par <- coef(fit)
```

- Validating SAG R package :
  - get the right answer(gradient with norm close to 0)
  - get the same answer as glmnet
- Creating SAG_CRF.update function based on these three solvers and comparing its output to the result of function `train.crf(crf)` in package CRF by Lin-Yun Wu. In CRF package, we can estimate parameter for given data and graph structure by function `train.crf(crf, instance)`.

```
crf <- make.crf(adj, n.states = nstats)
crf <- train.crf(crf, instances, nodefeature, edgefeature)
parameter <- crf$par  ## estimation parameters
node.potential <- crf$node.pot
edge.potential <- crf$edge.pot
```

4.2 Timeline

(2∼3 weeks) Build SAG R package and convert Mark Schmidt's C code with mex.h into C code can be called by R. The SAG's functions contain:

SAG_logistic.c

SAG_LipschitzLS_logsitic.c

SAGlineSearch_logistic.c

(1∼2 weeks) Convert the three SAG solvers BLAS version to C code with "R.h". These three functions are

SAG_logistic_BLAS.c

SAG_LipschitzLS_logsitic_BLAS.c

SAGlineSearch_logistic_BLAS.c

(2 weeks) Build vignettes with Mark's example data `rcv1_train` and `covtype.libsvm` data sets, and compare result of SAG with glmnet.

(2 weeks) Build SAG_CRF.update function for conditional random field and write another vignettes about SAG's speed and result comparing with cran CRF package's for test data rain data.

(1∼2 weeks) Write documents for these functions. We will convert Mark's documentation comments in C code to .Rd files.

(1∼2 weeks) Write code and document for a `SAGoptim` function which is analogous to function `optimx` in R package optimx, and `SAGoptim` function is an extension of function `optimx`, it can compute a minimizer of finite sum of functions.

5 Mentors

John Nash(nashjc@uottawa.ca) and Toby Dylan Hocking(toby.hocking@mail.mcgill.ca) are mentors of this project.

References

[1] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[2] Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. pages 2663–2671, 2012.

[3] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *arXiv preprint arXiv:1309.2388*, 2013.

[4] Mark Schmidt, Ann Clifton, and Anoop Sarkar. Non-uniform stochastic average gradient method for training conditional random fields. 2014.