A Comparison of First-order Algorithms for Machine Learning

Wei Yu Thomas Pock
Computer Graphics and Vision, Graz University of Technology, Austria

Abstract. Using an optimization algorithm to solve a machine learning problem is one of mainstreams in the field of science. In this work, we demonstrate a comprehensive comparison of some state-of-the-art first-order optimization algorithms for convex optimization problems in machine learning. We concentrate on several smooth and non-smooth machine learning problems with a loss function plus a regularizer. The overall experimental results show the superiority of primal-dual algorithms in solving a machine learning problem from the perspectives of the ease to construct, running time and accuracy.

1 Introduction

Optimization is the key of machine learning. Most machine learning problems can be cast as optimization problems. Furthermore, practical applications of machine learning usually involve a massive and complex data set. Thus, efficiency, accuracy and generalization of the optimization algorithm (solver) should be regarded as a crucial issue [2]. Many papers present dedicated optimization algorithms for specific machine learning problems. However, little attention has been devoted to the ability of a solver for a specific class of machine learning problems. The most common structure of machine learning problems is a loss function plus a regularizer. The loss function calculates the disparity between the prediction of a solution and the ground truth. This term usually involves the training data set. For example, the well known square loss is for the purpose of regression problems and hinge loss is for the purpose of maximum margin classification. The regularizer usually uses a norm function. For example, group lasso is an extension of the lasso for feature selection. It can lead to a sparse solution within a group.

In general, we consider convex optimization problems of the following form

$$\begin{cases} \text{minimize} & E(x) = F(x) + \lambda G(x) \\ \text{such that} & x \in C \end{cases}$$

where F and G are continuous, convex functions and C is a convex set. E denotes the energy of a machine learning problem. By convention, F usually denotes a loss function and G denotes a regularization term. λ is a parameter controlling the tradeoff between a good generalization performance and overfitting. This kind of problems frequently arise in machine learning. A substantial amount of literature assumes that either F or G is smooth and cannot be used to optimize the case where F and G are both non-smooth.

Some solvers provide an upper bound N on the number of iterations n such that $E^n - \widehat{E} \leq e, n \geq N$, where e is an error tolerance and $\widehat{E}$ is the minimum of E. Sometimes this estimation is too pessimistic which means the resultant

N is excessive large. In this case, it is hard to evaluate the performance of a solver by this upper bound. On the other side, convergence rate describes the speed of converging when a solver approaches the optimal solution. But it is unpredictable to know when n is large enough. Therefore, the performance of solvers is still difficult to tractable.

In this paper, we compare four state-of-the-art first-order solvers (Fobos[5], FISTA[1], OSGA[7] and primal-dual algorithms[3]) by the following properties: convergence rate, running time, theoretically known parameters, robustness in practice for machine learning problems. We present tasks within dimensionality reduction via compressive sensing, SVMs, group lasso regularizer for grouped feature selection, $\ell_{1,\infty}$ regularization for multi-task learning, trace norm regularization for max-margin matrix factorization. The last three machine learning problems are chosen from [10]. Unlike other literature which plots energy versus the number of iterations, in this paper we illustrate the results by log-log figures which clearly show the convergence rate in applications of machine learning.

The paper is organized as follows. Section 2 introduces four solvers. Then it summarizes primal-dual algorithm of Chambolle and Pock [3] and describes heuristic observations. Section 3 gives an introduction about the general structure (a loss function plus a regularizer) of machine learning problems we focus on in this paper. Section 4 demonstrates the performance of different solvers and the conclusion is presented at the end.

2 Solvers

2.1 Review

Fobos [5] and fast iterative shrinkage-thresholding algorithm [1] (FISTA) aim to solve a convex problem which is a sum of two convex functions. Neumaier [7] proposes a fast subgradient algorithm with optimal complexity for minimizing a convex function named optimal subgradient algorithm (OSGA). Chambolle and Pock [3] propose a primal-dual algorithm (hereinafter referred as PD CP) and applied it to several imaging problems. Tianbao Yang et al. [10] propose another primal-dual algorithm and applied it to machine learning tasks. However, PD CP is more general in the following aspects. The step size of PD CP is $\sqrt{2}$ times larger than [10] and PD CP makes steps in both primal and dual variables.

Solver	Convergence rate	F	G	E
Fobos	$O(1/\sqrt{n})$	convex	convex	-
FISTA	$O(1/n^2)$	$C^{1,1}$	convex	-
OSGA	$O(1/\sqrt{n})$	-	-	convex
PD CP	$O(1/n)$	convex	convex	-

Table 1: Tab Comparison of solvers

We summarize four solvers by Table 1. Each solver can achieve the convergence rate under the property of F, G, E given by each row of Table 1. When we solve machine learning problems using four solvers, we need to set the value of parameters and format the machine learning problems to a suitable model for a solver. Setting the initial step size C in Fobos for a problem with non-smooth function is an open question. For PD CP, we do not know the best ratio $a = \sqrt{\tau/\sigma}$. When solving a problem with a non-smooth function by FISTA, we have to smooth the non-smooth loss function. This rises a problem of selecting the value of smoothness parameter ϵ in smoothing techniques. To make the comparison convincing, we examine several values for the above three parameters C, a, ϵ and choose the best one.

2.2 The general PD CP

In this section, we review the primal dual algorithm proposed in [3]. Let X, Y be two finite dimensional real vector spaces with an inner product $\langle \cdot, \cdot \rangle$ and norm $\|\cdot\| = \langle \cdot, \cdot \rangle^{\frac{1}{2}}$. The map $K : X \to Y$ is a continuous linear operator with induced norm

$$\|K\| = \max \left\{ \|Kx\|_2 : x \in X,\ \|x\|_2 \leq 1 \right\}.$$

PD CP is to solve the generic saddle-point problem

$$\min_{x \in X} \max_{y \in Y} \left\{ \langle Kx, y \rangle + G(x) - F^*(y) \right\} \tag{1}$$

where x is the primal variable and y is the dual variable. G and F^* both are proper convex, lower-semicontinuous functions. The primal form of Equation (1) is $\min\limits_{x \in X} F(Kx) + G(x)$. To introduce the dual variable y, one way is using Lagrange multipliers, e.g., applicable when the function represents hard constraints. The other way is to calculate the convex conjugate F^* of loss function F. Then the loss function can be expressed by its convex conjugate. Take the hinge loss for example. We simplify the definition of hinge loss as $f(z) = \|\max(0, z)\|_1, z \in \mathbb{R}^N$. Its convex conjugate is,

$$f^*(y) = \begin{cases} 0 & y \in P \\ +\infty & y \notin P \end{cases}$$

where $P = \left\{ y \in \mathbb{R}^N : \forall y_i \in [0, 1] \right\}$, y_i is the i^{th} component of y. Let $z = 1 - Kx, x \in \mathbb{R}^d$ and $K \in \mathbb{R}^{N \times d}$. Then according to $f(z) = \max\limits_{y} \langle z, y \rangle - f^*(y)$, $f(1 - Kx)$ can be defined as,

$$f(1 - Kx) = \max_{y} \langle 1 - Kx, y \rangle - f^*(y).$$

Therefore in the primal-dual model, F^* should be formulated as $-\sum\limits_{i=1}^{N} y_i + f^*(y)$.

Before summarizing PD CP, we introduce the proximal operator. Let $G : X \to \mathbb{R} \cup \{+\infty\}$ be a proper convex, lower-semicontinuous function. The proximal operator of G with parameter τ is defined by

$$\begin{aligned} \operatorname{prox}_{\tau G}(v) &= (I + \tau \partial G)^{-1}(v) \\ &= \arg\min_{x \in X} (\tau G(x) + \frac{\|x - v\|_2^2}{2}) \end{aligned}$$

Since Euclidean norm is strong convex, $\operatorname{prox}_{\tau G}(v)$ is unique. PD CP proceeds by iteratively maximizing with respect to the dual variable and minimizing with respect to the primal variable by proximal operators.

Now we summarize PD CP as follows,

- Initialization: $\tau\sigma \leq \frac{1}{\|K\|^2}$,$\theta \in [0,1]$,$(x^0, y^0) \in X \times Y$,$\overline{x}^0 = x^0, \lambda \in \mathbb{R}$
- Iterations $n \geq 0$: Update x^n, y^n as follows,

$$\begin{aligned} y^{n+1} &= (I + \sigma\partial F^*)^{-1}(y^n + \sigma K\overline{x}^n) \\ x^{n+1} &= (I + \tau\lambda\partial G)^{-1}(x^n - \tau K^* y^{n+1}) \\ \overline{x}^{n+1} &= x^{n+1} + \theta(x^{n+1} - x^n) \end{aligned}$$

PD CP conducts proximal operators of σF^* and $\tau\lambda G$ respectively and then PD CP makes its scheme semi-implicit by letting $\overline{x}^{n+1} = x^{n+1} + \theta(x^{n+1} - x^n)$. This operation equals to making one more step in the direction of $x^{n+1} - x^n$. We refer to [3] for more information. The condition of the convergence of PD CP is $\tau\sigma \leq \frac{1}{\|K\|^2}$.

2.3 Heuristics for the primal dual algorithm

In this section, we introduce two heuristic observations of the primal-dual algorithm proposed by Chambolle and Pock [3].

2.4 Heuristics for the ratio of the primal step size to the dual step size

In PD CP, we define $a = \sqrt{\tau/\sigma}$. How to choose a to achieve the best performance is still an unsolved problem. However, Chambolle and Pock [3] show that

$$\begin{aligned} &[\langle Kx_N, \widehat{y}\rangle - F^*(\widehat{y}) + G(x_N)] - [\langle K\widehat{x}, y_N\rangle - F^*(y_N) + G(\widehat{x})] \\ &\qquad \leq \frac{1}{N}\left(\frac{\left\|\widehat{y} - y^0\right\|^2}{2\sigma} + \frac{\left\|\widehat{x} - x^0\right\|^2}{2\tau}\right) \end{aligned} \tag{2}$$

where $x_N = (\sum_{n=1}^{N} x^n)/N$, $y_N = (\sum_{n=1}^{N} y^n)/N$ and $(\widehat{x}, \widehat{y})$ is the saddle point. The RHS of Equation (2) is non-negative because

$$\begin{aligned} [\langle Kx_N, \widehat{y}\rangle - F^*(\widehat{y}) + G(x_N)] &\geq [\langle K\widehat{x}, \widehat{y}\rangle - F^*(\widehat{y}) + G(\widehat{x})] \\ &\geq [\langle K\widehat{x}, y_N\rangle - F^*(y_N) + G(\widehat{x})]\,. \end{aligned}$$

And when (x_N, y_N) is a saddle point, the LHS of Equation (2) equals zero. To minimize the upper bound of $[\langle Kx_N, \widehat{y}\rangle - F^*(\widehat{y}) + G(x_N)] - [\langle K\widehat{x}, y_N\rangle - F^*(y_N) + G(\widehat{x})]$, we plug $\tau = \frac{1}{\|K\|^2\sigma}$, the largest value that guarantees convergence, into the RHS of Equation (2) and get

$$\frac{1}{N}\left(\frac{\left\|\widehat{y} - y^0\right\|^2}{2\sigma} + \frac{\left\|\widehat{x} - x^0\right\|^2 \|K\|^2 \sigma}{2}\right). \tag{3}$$

Equation (3) is a convex function of σ. We take the derivative of Equation (3) with respect to σ,

$$\sigma = \frac{\left\|\widehat{y} - y^0\right\|}{\left\|\widehat{x} - x^0\right\| \|K\|}.$$

Thus we can conclude that $\frac{1}{N}\left(\frac{\left\|\widehat{y}-y^0\right\|^2}{2\sigma} + \frac{\left\|\widehat{x}-x^0\right\|^2}{2\tau}\right)$ reaches its minimum when $a = \sqrt{\frac{\tau}{\sigma}} = \frac{\left\|\widehat{x}-x^0\right\|}{\left\|\widehat{y}-y^0\right\|}$. However, $\widehat{x}$ and $\widehat{y}$ are not available because they are what we want to calculate.

2.5 Heuristics for the adaption of step sizes

We observe that the convergence condition [3] $\tau\sigma \leq \frac{1}{\|K\|^2}$ can be relaxed to accelerate the algorithm. We refer to the resulting scheme as Online PD CP. Although Online PD CP converges in the experiments of this paper, we do not prove its convergence theoretically. Online PD CP try to seek a larger step size. Once it finds one, it is faster than PD CP in the experiments of this paper. The difference between PD CP and Online PD CP is that Online PD CP starts with a larger step size ($\tau\sigma > \frac{1}{\|K\|^2}$) and decreases it according to a certain rule. We employ the following scheme,

$$\begin{cases} \widetilde{L}^{n+1} & = \frac{\left\langle K(x^n - x^{n-1}), y^{n+1} - y^n \right\rangle}{\|x^{n-1} - x^n\| \|y^{n+1} - y^n\|} \\ L^{n+1} & = \max\left(L^n, \widetilde{L}^{n+1}\right) \\ \tau^{n+1} & = \frac{a}{L^{n+1}}, \sigma^{n+2} = \frac{1}{aL^{n+1}} \end{cases} \tag{4}$$

Thus how to choose a proper L is the main concern of Online PD CP. As shown in Equation (4), we let $L^{n+1} = \max\left(L^n, \widetilde{L}^{n+1}\right)$. If $L^n < \widetilde{L}^{n+1}$, we increase L^{n+1} to $\widetilde{L}^{n+1}$. Thus, $\|K\|$ is a upper bound of L. Chambolle and Pock [3] proves the convergence when $L = \|K\|$.

Another observation is that the larger step size may lead to a large L. If $\widetilde{L}^{n+1}$ is smaller than L^n, Online PD CP does not update L^{n+1}. It is a sign of convergence and stability. That is, inappropriate large step sizes lead to divergence and a large $\widetilde{L}$ which may be close to $\|K\|$. This obeys the principle of Online PD CP. To explore more possible step sizes, we decrease the step size to an appropriate degree rather than choose the maximum between L^n and $\widetilde{L}^{n+1}$. This is the reason that we smooth L. We can devise different rules to smooth L. For example, we let $L = (L + \kappa \max(L, L^{n+1}))/(1+\kappa), \kappa > 0$. We set $\kappa = 0.618$ for all experiments. The other use of Online PD CP is the case that $\|K\|$ is non-calculable, e.g., K is not known explicitly.

3 The machine learning problems

Machine learning problems in this paper can be formulated as a convex minimization problem consisting of a loss function F and a regularizer G. We summarize machine learning problems in Table 2. In each row of table 2, the last two columns show the loss function and regularizer we used in the machine learning problem given by column 1. For more information about each machine learning problem, refer to the literature given by column 2.

In Table 2, $Q, H, \hat{H}$ are positive-semidefinite matrices and δ is an indicator function. In experiment of Kernel SVM, we calculate the dual form of the primal form [4] such that F becomes a smooth convex function as shown in row 5 of Table 2. And we solve this dual form by Fobos, FISTA and PD CP. Because this dual form is a constrained optimization problem which is not easy to be solved by OSGA, we use OSGA to solve the primal form [4] as shown in row 4 of Table 2.

Machine learning problem	Ref.	F	G
Dimensionality Reduction[1]	[6]	square	$\ell_{2,1}$
Linear SVM[2]	[8]	hinge	x^TQx
Kernel SVM[2]	[4]	hinge	x^THx
Kernel SVM [2]	-	$x^T\hat{H}x$	$-\sum x_i + \delta(x)$
Feature Selection[3]	[9]	absolute loss	group lasso
Multi-Task Learning [1]	[10]	ϵ-insensitive	$\ell_{1,\infty}$
Matrix Factorization[4]	[10]	hinge	trace norm

Table 2: Machine learning problems

4 Results

4.1 Experimental settings

When ∇F is with a Lipschitz constant L, the convergence rate of Fobos can be $O(1/n)$ [1]. Thus, we only compare Fobos with FISTA, OSGA and PD CP in the machine learning problems where F and G are both non-smooth. In all experiments, we initialize the primal variable and the dual variable to a null vector. All algorithms were implemented in Matlab and executed on a 2.66 GHz CPU, running a 64 Bit Windows system.

4.2 Covergence and time comparison

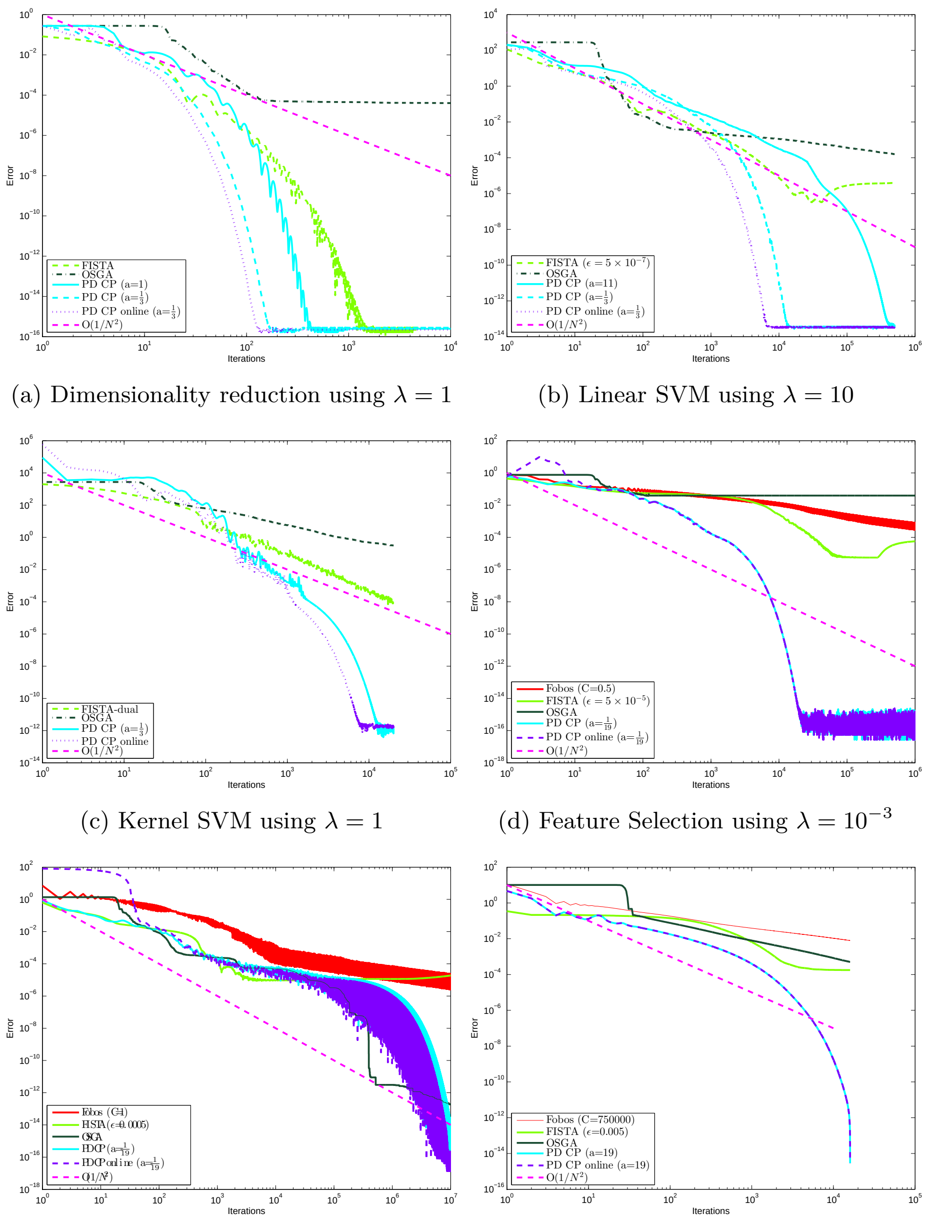
Figure 1 compares the practical convergence rate. During the different ranges of iterations, solvers can have different performances. For example, normally FISTA is faster within the first ten iterations but only reaches the optimal solution in the experiment of dimensionality reduction. When loss function and regularizer are both smooth as in Kernel SVM, FISTA shows exactly convergence rate $O(1/n^2)$. If we prolong the line of FISTA in Figure 1c, we can see FISTA needs about $10^{7.8}$ iterations to reach the optimal solution. However, PD CP only needs about 10^4 iterations. For the last three experiments where loss functions and regularizers are both non-smooth, Fobos shows a convergence rate $O(1/\sqrt{n})$. But for all experiments, PD CP is the fastest to reach the optimal solution. Although the performances of PD CP with different values of a are different, they show a similar practical convergence rate as shown in Figure 1a and 1b. PD CP has a much better practical convergence rate which is even better than $O(1/n^2)$. The experimental results show that FISTA is less capable to handle the case of two non-smooth terms. Furthermore, we may not get the optimal values by FISTA since we use the smoothing techniques. In all experiments, Online PD CP is better than or equal with PD CP. Refer the supplementary material for more results of machine learning problems. From Table 3, we can observe PD CP is still very competitive. Only in Multi-Task Learning, the running time per iteration of PD CP is slower than OSGA's. However, by observing Figure 1e, PD CP needs much less number of iterations to approach the optimal solution. Overall, PD CP have the superior performance among all machine learning problems considered in this paper.

[1] MNIST is available at http://yann.lecun.com/exdb/mnist/.

[2] 'svmguide1' is available at http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/.

[3] MEMset Donar is available at http://genes.mit.edu/burgelab/maxent/ssdata/.

[4] '100K MovieLens' is available at http://www.grouplens.org/node/12.



(a) Dimensionality reduction using $\lambda = 1$

(b) Linear SVM using $\lambda = 10$

(c) Kernel SVM using $\lambda = 1$

(d) Feature Selection using $\lambda = 10^{-3}$

(e) Multi-Task Learning using $\lambda = 10^{-3}$

(f) Matrix Factorization using $\lambda = 10^{-5}$

Figure 1: Comparison of convergence rate

5 Conclusion

This paper compares the performance of different optimization algorithms applied to six benchmark problems of machine learning. The primal dual algorithm [3] has the best perform with a fast empirical convergence rate in all problems concerned in this paper. Moreover, we give two heuristic suggestions for the primal dual algorithm. We hope that machine learning problems can make good use of the progress in optimization. When we use an optimization algorithm to solve a machine learning problem, we need to set the value of parameters both in machine learn problem and optimization algorithm. Our future concern is how to set them automatically. Our experiments show that PD CP is an efficient and robust solver for machine learning problems. Future work is to give theoretical explanation about the empirical convergence rate of PD CP and the convergence of Online PD CP.

per iteration (s)	Dimensionality Reduction	Linear SVM	Kernel SVM
PD CP	4.164×10^{-4}	11.1427×10^{-3}	0.1009
OSGA	6.25×10^{-4}	12.1379×10^{-3}	0.3
FISTA	5×10^{-4}	21.5387×10^{-3}	0.1052
per iteration (s)	Feature Selection	Multi-Task Learning	Matrix Factorization
PD CP	1.7648×10^{-2}	0.6214×10^{-3}	4.69386
OSGA	4.4×10^{-2}	0.3381×10^{-3}	16.80799
FISTA	6.4×10^{-2}	0.8014×10^{-3}	11.113636

Table 3: Running time

References

[1] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2(1):183–202, March 2009.

[2] Kristin P. Bennett and Emilio Parrado-Hernández. The interplay of optimization and machine learning research. *J. Mach. Learn. Res.*, 7:1265–1281, December 2006.

[3] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[4] Olivier Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19:1155–1178, 2007.

[5] John Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *J. Mach. Learn. Res.*, 10:2899–2934, December 2009.

[6] Junbin Gao, Qinfeng Shi, and Tibério S. Caetano. Dimensionality reduction via compressive sensing. *Pattern Recogn. Lett.*, 33(9):1163–1170, July 2012.

[7] Arnold Neumaier. Osga: A fast subgradient algorithm with optimal complexity. Unpublished manuscript, 2014.

[8] Ofir Pele, Ben Taskar, Amir Globerson, and Michael Werman. The pairwise piecewise-linear embedding for efficient non-linear classification. In *ICML*, 2013.

[9] Haiqin Yang, Zenglin Xu, Irwin King, and Michael R. Lyu. Online learning for group lasso. In *ICML*, 2010.

[10] Tianbao Yang, Mehrdad Mahdavi, Rong Jin, and Shenghuo Zhu. An efficient primal-dual prox method for non-smooth optimization. *Mach Learn*, 2014.