Adaptive Stochastic Dual Coordinate Ascent for Conditional Random Fields

Rémi Le Priol remi.lp.17@gmail.com
Ahmed Touati ahmed.touati@umontreal.ca
Simon Lacoste-Julien simon.lacoste-julien@umontreal.ca
MILA & DIRO - Université de Montréal

Abstract

This work investigates training Conditional Random Fields (CRF) by Stochastic Dual Coordinate Ascent (SDCA). SDCA enjoys a linear convergence rate and a strong empirical performance for independent classification problems. However, it has never been used to train CRF. Yet it benefits from an exact line search with a single marginalization oracle call, unlike previous approaches. In this paper, we adapt SDCA to train CRF and we enhance it with an adaptive non-uniform sampling strategy. Our preliminary experiments suggest that this method matches state-of-the-art CRF optimization techniques.

The CRF model [1] is a common tool in natural language processing and computer vision for structured prediction. The optimization of this model is notoriously challenging. Schmidt et al. [2] describes a practical implementation of the stochastic average gradient (SAG) algorithm [3] for CRFs and proposes a non-uniform sampling scheme that boosts performance. This algorithm (SAG-NUS) is currently the state of the art for CRFs optimization and we refer to Schmidt et al. [2] for a review of competing methods.

Deterministic methods such as L-BFGS [4, 5] have linear convergence rate but the cost per iteration is large. On the other hand, the Online Exponentiated Gradient method (OEG) [6] and SAG are both members of a family of algorithms with cheap stochastic updates and linear convergence rates. They are called variance reduced algorithms, because their common point is to use memory to reduce the variance of the gradient as they get closer from the optimum. Johnson and Zhang [7] coined the name with SVRG and Defazio et al. [8] unified the family.

SDCA is one member of this family that has not been applied yet to CRFs. Similar to OEG, it works on the marginalized dual variable, with two advantages. First, its linear convergence guarantee bears on the duality gap, whereas OEG's bears on the dual objective. Second, it can perform an exact line search using only one marginalization oracle call. In this paper we present a new adaptive non-uniform sampling strategy for SDCA, that can be fitted to the CRF setting. We compare this algorithm with SAG-NUS on the OCR dataset [9], matching its performance.

1 Conditional Random Fields

Primal problem. CRFs model the conditional probability of a structured output $y \in \mathcal{Y}$ given an input $x \in \mathcal{X}$ with an exponential family of sufficient statistic $F(x, y) \in \mathbb{R}^d$ and parameters $w \in \mathbb{R}^d : p(y|x; \boldsymbol{w}) \propto \exp(\boldsymbol{w}^\top F(x, y))$. The feature vector F decomposes as a sum over cliques $c \in \mathcal{C}$ of y's graphical model $F(x, y) = \sum_c F_c(x, y_c)$.

The parameter is learned by minimizing the l_2-regularized negative log-likelihood. We write it to fit into the description of SDCA for multi-class classification from Shalev-Shwartz and Zhang [10]. Denote A_i the $d \times |\mathcal{Y}_i|$ matrix whose columns are the *corrected features* $\{\psi_i(y) := F(x_i, y_i) - F(x_i, y)\}_{y \in \mathcal{Y}_i}$. Denote also $\phi_i(z) = \log\left(\sum_{y \in \mathcal{Y}_i} \exp(z_y)\right)$ the log-partition function. The negative log-likelihood can be written $-\log(p(y_i|x_i; \boldsymbol{w}) = \phi_i(-A_i^\top \boldsymbol{w})$. The problem at stakes is:

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \mathcal{P}(w) = \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \phi_i(-A_i^\top \boldsymbol{w}). \tag{1}$$

Dual Formulation. The above minimization problem (1) has an equivalent *convex dual* problem [11]. Denote $\Delta_{|\mathcal{Y}_i|}$ the $|\mathcal{Y}_i|$-dimensional probability simplex. Denote $\alpha_i \in \Delta_{|\mathcal{Y}_i|}$ the set of dual variables for a

OPTML 2017: 10th NIPS Workshop on *Optimization for Machine Learning* (NIPS 2017).

given x_i. The dual problem is

$$\max_{\boldsymbol{\alpha}=(\alpha_1,\dots,\alpha_n)\in\Delta_{|\mathcal{Y}_1|}\times\dots\times\Delta_{|\mathcal{Y}_n|}} \mathcal{D}(\boldsymbol{\alpha}) := -\frac{1}{2\lambda}\|\frac{1}{n}\sum_i A_i\alpha_i\|^2 + \frac{1}{n}\sum_{i=1}^n H(\alpha_i), \tag{2}$$

where $H(\alpha_i) := -\sum_{y\in\mathcal{Y}_i}\alpha_i(y)\log(\alpha_i(y))$ is the entropy of the probability distribution α_i. If we define the *conjugate weight* function $\hat{w}$ as follows:

$$\hat{w}(\boldsymbol{\alpha}) := \frac{1}{n\lambda}\sum_i A_i\alpha_i = \frac{1}{\lambda n}\sum_{i=1}^n \mathbb{E}_{y\sim\alpha_i}[\psi_i(y)],$$

then it is known that $\hat{w}(\boldsymbol{\alpha}^\star) = \boldsymbol{w}^\star$ where $\boldsymbol{w}^\star$ and $\boldsymbol{\alpha}^\star$ are respectively the optimal primal parameters and the optimal dual parameters. And if we define the *conjugate* probabilities $\hat{\alpha}_i$ as follows:

$$\forall i, \quad \hat{\alpha}_i(\boldsymbol{w}) := \nabla_z\phi_i(-A_i^\top\boldsymbol{w}) = p(.|x_i;\boldsymbol{w}), \tag{3}$$

then we get another optimality condition $\hat{\alpha}(\boldsymbol{w}^\star) = \boldsymbol{\alpha}^\star$. Hence the optimization problem (1) can be seen as a fixed point problem for the function $\hat{w}\circ\hat{\alpha}$. Optimization algorithms can often be interpreted as a back and forth between the conjugate variables w and $\hat{w}(\hat{\alpha}(\boldsymbol{w}))$ (SAG) or α and $\hat{\alpha}(\hat{w}(\boldsymbol{\alpha}))$ (SDCA).

Duality gaps. Note that we always have that $\mathcal{P}(\boldsymbol{w}) \geq \mathcal{D}(\boldsymbol{\alpha})$, with equality at the optimum. The *duality gap* is defined by $g(\boldsymbol{w},\boldsymbol{\alpha}) = \mathcal{P}(\boldsymbol{w}) - \mathcal{D}(\boldsymbol{\alpha})$. One can verify that:

$$g(\boldsymbol{w},\hat{\alpha}(\boldsymbol{w})) = \frac{\lambda}{2}\|\boldsymbol{w}-\hat{w}(\hat{\alpha}(\boldsymbol{w}))\|^2 = \frac{1}{2\lambda}\|\nabla\mathcal{P}(\boldsymbol{w})\|^2 \tag{4}$$

$$g(\hat{w}(\boldsymbol{\alpha}),\boldsymbol{\alpha}) = \frac{1}{n}\sum_i D_{KL}(\alpha_i||\hat{\alpha}_i(\hat{w}(\boldsymbol{\alpha})) \tag{5}$$

The duality gap (5) is typically used to monitor the optimization, but we argue that it also provides additional information which can accelerate SDCA's convergence. Therefore, we propose a non-uniform sampling scheme based on the duality gap in the following sections.

2 Proximal Stochastic Dual Coordinate Ascent

Prox-SDCA, option II [10] is a primal-dual method to solve (1). At each step, we update a random block α_i to maximize the dual objective $\mathcal{D}(\boldsymbol{\alpha})$:

$$\alpha_i \leftarrow \alpha_i + \gamma\delta_i = (1-\gamma)\alpha_i + \gamma\hat{\alpha}_i(\hat{w}(\boldsymbol{\alpha})) \tag{6}$$

where $\delta_i = \hat{\alpha}_i(\hat{w}(\boldsymbol{\alpha})) - \alpha_i$ is a guaranteed ascent direction for $\mathcal{D}(\boldsymbol{\alpha})$, and $\gamma\in[0,1]$ is a step-size which is either fixed or found via line search.

In the CRF setting, the dual variable α_i is exponentially large in the input size x_i. It might not even fit in memory. Instead, the standard approach used in OEG and SAG is to consider the marginal probabilities $(\mu_c)_{c\in\mathcal{C}}$ on the cliques of the graphical model. Similarly, we replace $\boldsymbol{\alpha}$ by $\boldsymbol{\mu} = (\mu_1,\cdots\mu_n)$, where $\mu_i\in\prod_c\Delta_{|c|}$ is the concatenation of all the clique marginal vectors for the sample i. The associated weight vector can be expressed as function of $\boldsymbol{\mu}$ thanks to the separability of the features (see (7) below). Now, assume that the graph has a *junction tree* structure $T = (\mathcal{C},\mathcal{S})$, where C are the maximal cliques and S their separations. We can then run message passing on the junction tree to infer new marginals given weights $\boldsymbol{w}$: $\hat{\mu}_i(\boldsymbol{w}) = p(y_c = .|x_i;\boldsymbol{w})$. We can also now recover the joint probability $\alpha_i(y)$ as a function of its marginals $\mu_{i,c}$:

$$\hat{w}(\boldsymbol{\mu}) = \frac{1}{\lambda n}\sum_i\sum_c \mathbb{E}_{\mu_{i,c}}[\psi_{i,c}] \tag{7}$$

$$\alpha_i(y) = \frac{\prod_{c\in\mathcal{C}}\mu_{i,c}(y_c)}{\prod_{s\in\mathcal{S}}\mu_{i,s}(y_s)} \tag{8}$$

Equation (8) in turn allows us to compute the entropy and the divergences of the joints, using only the marginals. Both of these are key quantities for the Algorithm 1.

$$H(\alpha_i) = \sum_c H(\mu_{i,c}) - \sum_s H(\mu_{i,s}) := \mathcal{H}(\mu_i) \tag{9}$$

$$D_{KL}(\alpha_i||\alpha_i') = \sum_c D_{KL}(\mu_{i,c}||\mu_{i,c}') - \sum_s D_{KL}(\mu_{i,s}||\mu_{i,s'}) := \mathcal{D}(\mu_i||\mu_i') \tag{10}$$

Algorithm 1 Prox-SDCA for CRF

$\forall i$, set $\mu_i^{(0)}$ in the interior of $\prod_c \Delta_{|c|}$ and $\boldsymbol{w}^{(0)} := \hat{w}(\boldsymbol{\alpha}^{(0)}) = \frac{1}{\lambda n} \sum_i \sum_c \mathbb{E}_{\mu_{i,c}^{(0)}}[\psi_{i,c}]$
Let $\forall i \quad g_i = 1$ (optional)
for $k = 0 \dots K$ **do**
  Pick i at random in $\{1, \dots, n\}$ (optionally, proportional to g_i)
  Compute $\forall c, \mu'_{i,c}(y_c) := p(y_c|x_i; w^{(k)})$ (marginalization oracle)
  Let $g_i = \mathcal{D}(\mu_i || \mu'_i)$ (optional, individual duality gap)
  Let $\delta_i = \mu'_i - \mu_i^{(k)}$ (ascent direction)
  Let $v_i = \frac{1}{\lambda n} \sum_c \mathbb{E}_{\mu_{i,c}}[F_c(x_i, .)] - \mathbb{E}_{\mu'_{i,c}}[F_c(x_i, .)]$ (primal direction)
  Solve $\gamma^* = \arg\max_{\gamma \in [0,1]} \mathcal{H}(\mu_i^{(k)} + \gamma \delta_i) - \frac{\lambda n}{2} \|\boldsymbol{w}^{(k)} + \gamma v_i\|^2$ (Line Search)
  Update $\mu_i^{(k+1)} := \mu_i^{(k)} + \gamma^* d_i$
  Update $\boldsymbol{w}^{(k+1)} := \boldsymbol{w}^{(k)} + \gamma^* v_i$

3 Implementation

Initialization. Schmidt et al. [2] improves the starting point of OEG by setting μ_i to the result of marginalizing a CRF with cliques' potentials set to a large value (such as 10) on the correct labels, and zeros for the others clique assignments. This initialization is beneficial for dual methods.

Line Search. The ill-definition of the entropy next to the border of the simplex leads to numerical instability. The implementation of quadratic methods such as Newton-Raphson is tricky. Most values should be treated in the log-space. On average, the line search can be done with 2 iterations to reach precision 10^{-2}. Each iteration has a cost proportional to the size of the marginals : $\sum_c |\mathcal{Y}_c|$. This is not really free, although it can be parallelized.

Non-Uniform Sampling. An adaptive non-uniform sampling scheme for SDCA has been proposed by Csiba et al. [12] to train a binary classifier. Their idea is to sample *almost* proportionally to the norm of the dual ascent direction. Yet as the norm is not separable over cliques, this scheme cannot be applied to CRFs. Inspired by the approach followed by Osokin et al. [13] to improve the Block-Coordinate Frank-Wolfe algorithm [14], we propose to bias sampling towards examples whose duality gaps are large. Thanks to (10), the duality gaps are efficiently computable from the marginals and thus we can apply this scheme to CRFs. In terms of performance, our experiments on non-structured logistic regression revealed that both methods lead to similar improvements of the convergence rate. On the OCR task, non-uniform sampling improves substantially the convergence of SDCA. Without this improvement, SDCA is on par with OEG.
All adaptive NUS schemes keep track of outdated values (local Lipschitz constant, ascent norm or individual gap) to estimate the current one. These estimates become stale as we perform more iterations. In fact, there is no convergence-guarantee for BCFW or SAG-NUS when one samples proportionally to past estimates. To fight this trend, SAG-NUS uses a mixed scheme: it samples uniformly half the time and non-uniformly the other half. BCFW-NUS opts for a full batch update of the gaps every 10 epochs or so, providing a convergence certificate at the same time. Surprisingly, our experiments on OCR seem to indicate that the staleness of individual gaps estimates does not affect the convergence. Interestingly enough, mixed scheme and fully non-uniform scheme, with or without full batch updates, all have the same convergence behavior, as soon as we sample non-uniformly often enough.

4 Experiments

Comparison with OEG and SAG. We implemented SDCA to train a classifier on the optical character recognition (OCR) dataset [9]. We compare our method with SAG and OEG implemented by Schmidt et al. [2], with the same experimental setup ($\lambda = 1/n$). We plot the sub-optimality as a function of the number of marginalization oracle calls in Figure 1 (left). It is a standard implementation-independent way to assess the convergence of different structured prediction algorithms, given the substantial cost of the oracle. In these terms, SDCA-NUS is quite competitive with SAG-NUS* (improved version of SAG with a line-search skipping scheme). SAG, OEG and SDCA share the same memory requirements.

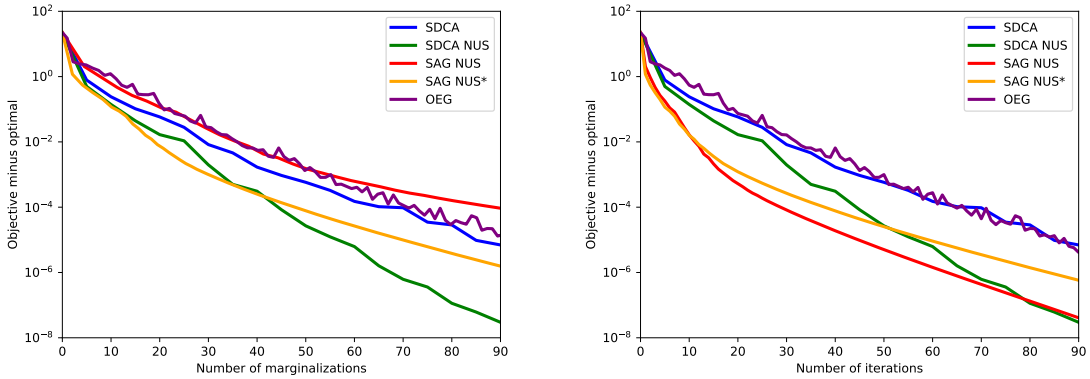

Figure 1: Primal suboptimality as a function of the number of marginalization oracle calls divided by n (left) or number of parameter updates divided by n (right).

This comparison is not completely fair to SAG-NUS though. Each step of the line search that we use has actually a cost proportional to the size of the marginals. The complexity of the marginalization oracle (sum-product on the junction tree) has actually the same scaling, with a comparable constant factor. However, each step of our line search can be computed in parallel while the sum-product is purely sequential. As an alternative perspective which is more fair to SAG, we thus also plot the sub-optimality as a function of the number of parameter updates. Then SAG beats SDCA by a margin, except in the high precision regime.

Influence of the line search precision. One of the hyper-parameters of SDCA is the precision we want on the optimum of the line search. In fact, setting a high precision does not improve the convergence rate at all. Yet it makes the algorithm slower as asking for a precision 10^{-8} takes 4 steps of Newton versus 2 for a precision 10^{-2}. This is expected since the accuracy of the optimization comes from the iterates $\boldsymbol{\alpha}$ and $\hat{\alpha}(\hat{w}(\boldsymbol{\alpha}))$ getting closer from each other.

5 Discussion

In this work, we explore using SDCA for training CRFs for the first time. We highlight technical details to make a practical implementation. Our non-uniform sampling strategy improves experimental performance. We wish to investigate the theoretical properties of this strategy. Our preliminary experiments on OCR indicate that SDCA is similar to SAG in terms of oracle calls, though other sequential datasets will be tested. As SDCA only requires a marginalization oracle per line-search, the implementation could become advantageous over SAG when the marginalization oracle becomes much more expensive than evaluating the entropy function from the marginals. Examples for this scenario are: when a parallel implementation is used for the entropy computation; or when the marginalization oracle uses an iterative approximate inference algorithms such as TRW BP whereas an approximation of the entropy is direct from the marginals [15].

References

[1] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

[2] Mark Schmidt, Reza Babanezhad, Mohamed Ahmed, Aaron Defazio, Ann Clifton, and Anoop Sarkar. Non-uniform stochastic average gradient method for training conditional random fields. In *Artificial Intelligence and Statistics*, 2015.

[3] Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems*, 2012.

[4] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, 2003.

[5] Hanna Wallach. Efficient training of conditional random fields. Master's thesis, University of Edinburgh, 2002.

[6] Michael Collins, Amir Globerson, Terry Koo, Xavier Carreras, and Peter L Bartlett. Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *Journal of Machine Learning Research*, 2008.

[7] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, 2013.

[8] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, 2014.

[9] Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*, 2004.

[10] Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, 2016.

[11] Guy Lebanon and John D Lafferty. Boosting and maximum likelihood for exponential models. In *Advances in Neural Information Processing Systems*, 2002.

[12] Dominik Csiba, Zheng Qu, and Peter Richtárik. Stochastic dual coordinate ascent with adaptive probabilities. In *International Conference on Machine Learning*, 2015.

[13] Anton Osokin, Jean-Baptiste Alayrac, Isabella Lukasewitz, Puneet Dokania, and Simon Lacoste-Julien. Minding the gaps for block Frank-Wolfe optimization of structured SVMs. In *International Conference on Machine Learning*, 2016.

[14] Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *International Conference on Machine Learning*, 2013.

[15] Rahul G. Krishnan, Simon Lacoste-Julien, and David Sontag. Barrier Frank-Wolfe for marginal inference. In *Advances in Neural Information Processing Systems*, 2015.