SampleRank: Learning Preferences from Atomic Gradients

Michael Wick, Khashayar Rohanimanesh, Aron Culotta[*], Andrew McCallum
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003
{*mwick,khash,culotta,mccallum*}*@cs.umass.edu*

Abstract

Large templated factor graphs with complex structure that changes during inference have been shown to provide state-of-the-art experimental results on tasks such as identity uncertainty and information integration. However, learning parameters in these models is difficult because computing the gradients require expensive inference routines. In this paper we propose an online algorithm that instead learns preferences over hypotheses from the gradients between the atomic steps of inference. Although there are a combinatorial number of ranking constraints over the entire hypothesis space, a connection to the frameworks of sampled convex programs reveals a polynomial bound on the number of rankings that need to be satisfied in practice. We further apply ideas of passive aggressive algorithms to our update rules, enabling us to extend recent work in confidence-weighted classification to structured prediction problems. We compare our algorithm to structured perceptron, contrastive divergence, and persistent contrastive divergence, demonstrating substantial error reductions on two real-world problems (20% over contrastive divergence).

1 Introduction

The expressive power of probabilistic programming languages (Richardson and Domingos, 2006; Milch et al., 2006; Goodman et al., 2008; McCallum et al., 2009) has given rise to complex factor graphs that preclude exact training methods because traditional machine learning algorithms involve expensive inference procedures as subroutines. For example, maximum likelihood gradients require computing marginals and perceptron gradients require decoding. Furthermore, these inference routines must be invoked for each update, and therefore lie in some of the inner loops of learning. A number of approaches address this issues to various degrees (Hinton, 2002; Hal Daumé and Marcu, 2005; Tieleman, 2008). For example, LASO learns to score incomplete configurations based on a binary loss function that determines if a partial configuration could lead to the ground-truth; contrastive divergence (CD) approximates gradients by sampling in the neighborhood of the ground-truth to obtain inexpensive updates.

In this paper we present *SampleRank* (Culotta, 2008), an alternative to contrastive divergence that computes *atomic* gradients between neighboring configurations according to a loss function (for example, F1 score). This signal induces a preference over the samples, and parameters are learned to reflect these preferences. Because SampleRank is concerned only with the ranking of hypothesized samples, and not with approximating likelihood gradients, the algorithm is not required to be governed by a strict Markov chain. In particular, this allows large parameter updates to be made between intermediate samples, an immediate advantage over persistent contrastive divergence (PCD) which

[*] Southeastern Louisiana University (culotta@selu.edu)

must ensure updates between neighboring samples are sufficiently small. Therefore, SampleRank enjoys greater freedom in updating the parameters, and indeed we can apply ideas from passive aggressive algorithms (MIRA) (Crammer et al., 2006) and feature-specific confidence-weights (Dredze et al., 2008) in order to achieve even greater performance.

In our experiments, we compare SampleRank to several alternative learning algorithms and demonstrate that it reduces error over several variations of CD, PCD, and perceptron on three different datasets. We also explore different update rules including MIRA and confidence weighting, and finally compare CD and SampleRank in chains not governed by strict ergodic properties.

2 Preliminaries

A factor graph $\mathcal{G} = \langle V, \Psi \rangle$ is a bipartite representation of a probability distribution π, that decomposes into a set of factors $\Psi = \{\psi\}$. Random variables V can further be divided into observables X and hidden variables Y. Using lowercase letters (e.g., x, y) to denote values from the domains of the random variables, the conditional distribution given by the factor graph can be written: $\pi(Y = y \mid X = x; \theta) = \frac{1}{\mathcal{Z}_X} \prod_{\psi \in \Psi} \psi(y^i, x^i)$. Where $\mathcal{Z}_{\mathbf{X}}$ is the input-dependent normalizer, and factor ψ takes assignments to sets of hidden y^i variables and observables x^i as arguments. We define the feasible region $\mathcal{F} \subseteq Y$ of the factor graph to contain only non-zero-probability configurations $\mathcal{F} = \{y \in Y \mid \pi(y|x) > 0\}$.

Parameter learning in factor graphs generally involves the update rule: $\theta \leftarrow \theta + \eta\nabla$, where ∇ is a correction that is applied to the current estimate of the parameters, and η is the learning rate. For example, maximum likelihood gradients ($E_{\mathcal{D}} \langle \Phi \rangle - E_\theta \langle \Phi \rangle$) involve the $\#\mathcal{P}$-hard problem of computing marginals for feature expectations under the model, and Collins' structured perceptron gradient ($\phi(y^\star_{\mathcal{D}}) - \phi(y^\star_\theta)$) requires the $\mathcal{NP}$-hard problem of computing the MAP configuration $y^\star_\theta$.

3 SampleRank

In this section we describe our algorithm, SampleRank, which learns configuration rankings from atomic gradients. Recall that this work is motivated by the fact that most gradient methods require an expensive black-box inference routine $\mathbb{B}$ (e.g., for returning $y^\star$). Now, we will assume $\mathbb{B}$ is no longer a black-box, and we can indeed observe the underlying procedure that performs inference. Furthermore, we will assume that each step of that procedure produces a configuration pair $\langle y', y \rangle$ (as is the case with MCMC and local search methods). More precisely, let $\delta : \mathcal{F} \rightarrow \mathcal{F}$ be the nondeterministic transition function that represents a draw from a proposal distribution $\mathbb{Q} : \mathcal{F} \times \mathcal{F} \rightarrow [0, 1]$ s.t. $\sum_{y'} \mathbb{Q}(y'|y) = 1$. At each step of inference δ is invoked to yield a new configuration y' from a current configuration y. Let $\phi : Y \times X \rightarrow \mathbb{R}^{|\theta|}$ denote the sufficient statistics of a configuration, then we define the *atomic gradient* ∇ (from inference step $y' = \delta(y)$) as

$$\nabla = \phi\left(\delta(y), x\right) - \phi(y, x) = \phi(y', x) - \phi(y, x) \tag{1}$$

Let G_θ be a factor graph representation of a probability distribution π parameterized by θ, with feasible region $\mathcal{F}$; and let $\mathbb{P} : Y \times Y \rightarrow \{0, 1\}$ be a preference function:

$$\mathbb{P}(y', y) = \begin{cases} 1 & \text{if } y' \text{ is preferred} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

then SampleRank is given in Algorithm 1. Despite its apparent simplicity, SampleRank is actually quite general. In fact, SampleRank provides tremendous flexibility, enabling both proposal distributions and preference functions to be customized to a particular domain or setting. For example, in unsupervised settings, the preference function could by governed by prior knowledge (or by measuring a generative process); in supervised settings, preference functions can exploit ground-truth labels (e.g., comparing F1 scores). We have established a convergence proof in our technical report (Rohanimanesh et al., 2009) that is completely agnostic to the preference function and is applicable to nearly arbitrary (including non-ergodic) proposal distributions. Many inference algorithms of interest (such as Gibbs, and Metropolis-Hastings), are covered by the convergence results.

Algorithm 1 Atomic Gradient Method (SampleRank)

Inputs: training data $\mathcal{D}$ with factor graph $\mathcal{G} = \langle X, Y, \Psi, \theta \rangle$
Initialization: set $\theta \leftarrow \mathbf{0}$, set $y \leftarrow y_0 \in \mathcal{F}$
Output: parameters θ
for $t = 1, 2, \ldots$ until convergence **do**
  $y' \sim \mathbb{Q}(\cdot|y)$
  $\nabla \leftarrow \phi(y', x) - \phi(y, x)$
  if $[\theta \cdot \nabla > 0 \wedge \mathbb{P}(y, y')]$ **then**
    $\theta \leftarrow \theta - \eta\nabla$
  else if $[\theta \cdot \nabla \leq 0 \wedge \mathbb{P}(y', y)]$ **then**
    $\theta \leftarrow \theta + \eta\nabla$
  end if
  if chooseToAccept(y, y', θ) **then** $y \leftarrow y'$
end for

Finally, we note that although the learning rate η is traditionally a scalar, it can be adjusted by a passive aggressive method (MIRA), or be vector-valued (as in confidence weighting). We adapt these methods to our structured setting by casting each update as a binary classification problem where the configuration pair is the data instance and the preference function serves as the label. The sufficient statistics of the classification problem are then the components of the atomic gradient ∇.

3.1 Efficient Gradient Computations

Equation 1 implies that computing ∇ requires obtaining sufficient statistics from two configurations y and y', which can be expensive. However, due to the local nature of search, this can be avoided entirely. Taking advantage of the fact that sufficient statistics present in both y and y' cancel, we can compute ∇ directly as: $\nabla = \sum_{\phi \in \nu(\Delta')} \phi(y^i, x) - \sum_{\phi \in \nu(\Delta)} \phi(y^i, x)$, where Δ' is the new setting to the variables that have changed, and Δ is the previous setting to those variables before the transition. The neighborhood function $\nu(\Delta')$ returns the sufficient statistics that require these variable settings as arguments. For many commonly used transition functions (e.g., Gibbs or split-merge (Jain and Neal, 2004)), we save computing an order of n factors over the brute force method.

3.2 Sample Complexity

Although our algorithm considers a combinatorial number of ranking constraints (in the configuration space of the factor graph), SampleRank can alternatively be viewed as an instance of *randomized constraint sampling* (Farias and Roy, 2004, 2006) or *sampled convex programs (SCP)* (Calafiore and Campi, 2005), where errors are bounded on approximations to convex optimization problems involving an intractable number of constraints. The unifying idea of these frameworks is the notion of a relaxed optimization problem that considers just a manageable set of *i.i.d.* constraints. This manageable set is actually sampled from the full constraint set according to a probability distribution ρ. Solutions are then obtained by optimizing the relaxed problem over the subset of constraints. The underlying intuition of this idea is that most constraints are either (a) inactive, (b) redundant (captured by other constraints), or (c) negligible (have only a minor impact on the solution). The fundamental question that these frameworks address is how many samples are required such that the solution to the resulting relaxed optimization problem violates only a *small* subset of constraints. It has been shown (Farias and Roy, 2004) that in particular, for a problem with K variables, with a number of sampled constraints given by: $N = \mathcal{O}\left(\frac{1}{\epsilon}\left(K \ln \frac{1}{\epsilon} + \ln \frac{1}{\delta}\right)\right)$ any optimal solution to the relaxed problem with a probability at least $(1 - \delta)$ violates a set of constraints $\mathcal{V}$ with measure $\rho(\mathcal{V}) \leq \epsilon$, where $\rho(.)$ is a probability distribution over the constraint space from which *i.i.d.* sample constraints are generated.

SampleRank can be described as the following SCP:

$$\begin{cases} \text{minimize} & c^T\theta \\ \text{subject to} & \theta \cdot \phi(x, y^-) - \theta \cdot \phi(x, y^+) \leq 0, \quad \forall \langle y^-, y^+ \rangle \in \Omega_{\mathbb{Q}} \end{cases}$$

where c is a vector of importance weights, and $\Omega_{\mathbb{Q}}$ is a set of sampled constraints generated by SampleRank throughout the course of local search (e.g. MCMC) guided by a proposal distribution

$\mathbb{Q}(.))$[1]. Taking $K = |\theta|$ reveals that a reasonable model can be learned by sampling a polynomial size subset of the constraints.

4 Experiments

In this section we demonstrate how SampleRank can be used to train a conditional random field (CRF) with first-order logic features defined over sets of instances. In particular, we focus on two clustering problems: ontology alignment and noun-phrase coreference resolution. In ontology alignment, all concepts belonging to the same cluster are considered equivalent; similarly, in coreference, all mentions belonging to the same cluster are considered coreferent.

Setup:
The CRF contains variables for each possible cluster (with a factor measuring the cluster's compatibility) and variables between mention-pairs across clusters (with a factor measuring their disparity), resulting in a combinatorial number of variables and factors. For more details about this CRF, please see (Culotta et al., 2007), or our technical report (Rohanimanesh et al., 2009). For MAP inference, we use Metropolis-Hastings, where the proposal distribution randomly picks a data-point then randomly moves it to another cluster (or creates a new cluster). SampleRank treats each proposal as an atomic inference step[2]; our preference function for both problems exploits the ground-truth labels and is defined to be $\mathbb{P}(y', y) = 1$ if accuracy(y')>accuracy(y), and 0 otherwise.

Data:
For coreference experiments we use the ACE 2004 dataset, which contains 443 documents; 336 for training and 107 for testing. We run each online method over the training set (ten times), performing 4000 proposals (inference steps) per document. For the ontology experiments we use two domains from the Illinois Semantic Integration Archive (ISIA): *course catalog*, and *company profile* (for more discussion on these domains see Doan et al. (2002)).

Results:
First, we compare the BCubed F1 (in coreference) of three learning rates η: constant unit updates (f1=77.6), MIRA updates (f1=80.5), and the approximate version of confidence weighted updates (f1=81.5). Confidence weighted updates have previously been shown to improve results in classification problems, and we were pleased to see a similar improvement in a structured prediction setting. Next (Table 1), we compare SampleRank to variants of contrastive divergence, persistent contrastive divergence, and perceptron on three datasets (ACE newswire coreference, course catalog ontology alignment, and company profile ontology alignment). We observe substantial error reductions over variants of contrastive divergence (more than 20% on ACE coreference—a new state-of-the-art result); in particular, we observe even greater improvements (over CD) in chains lacking detailed balance. Columns indicated as *valid MCMC chain* use a proposer that moves a single variable and obeys detailed balance. The column indicated as *not valid MCMC chain* uses a more sophisticated proposer that adapts to the model, but does not necessarily obey detailed balance. Note how the sophisticated proposal distribution hinders performance for CD, but actually helps SampleRank.

5 Acknowledgements

This work was supported in part by the Center for Intelligent Information Retrieval, in part by SRI International subcontract #27-001338 and ARFL prime contract #FA8750-09-C-0181, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249, in part by Army prime contract number W911NF-07-1-0216 and University of Pennsylvania subaward number 103-548106, and in part by UPenn NSF medium IIS-0803847. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

[1]Note that in this particular case the choice of the importance weight vector c is unimportant (e.g., we can chose $c = \mathbf{0}$) if the goal is to find a feasible solution for θ. For a quadratic program, the optimization objective should be replaced by $\theta^T\theta$.

[2]Despite the large graph, computing the atomic gradients requires evaluating only a constant number of cluster-wise factors.

	ACE coreference				Ontology alignment	
	valid MCMC chain		not valid MCMC chain		valid MCMC chain	
Method	**F1 (B^3)**	**F1 (PW)**	**F1 (B^3)**	**F1 (PW)**	**F1 (Course)**	**Match F1 (Company)**
SampleRank	**80.1**	**45.1**	**81.5**	**51.0**	**89.8**	**82.1**
CD-1	75.1	22.4	75.1	22.4	76.9	64. 8
CD-10	76.03	33.7	73.1	19.3	72.4	67.8
PCD-10	77.9	37.3	75.7	19.5	67.9	74.6
Perceptron	—	—	—	—	69.7	60.2

Table 1: Comparison of SampleRank with other training methods

References

Calafiore, G. and Campi, M. C. (2005). Uncertain convex programs: Randomized solutions and confidence levels. *Mathematical Programming*, 102:25–46.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585.

Culotta, A. (2008). *Learning and inference in weighted logic with application to natural language processing*. PhD thesis, University of Massachusetts.

Culotta, A., Wick, M., Hall, R., and McCallum, A. (2007). First-order probabilistic models for coreference resolution. In *HLT*, pages 81–88.

Doan, A., Madhavan, J., Domingos, P., and Halevy, A. Y. (2002). Learning to map between ontologies on the semantic web. In *WWW*, page 662.

Dredze, M., Crammer, K., and Pereira, F. (2008). Confidence-weighted linear classification. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 264–271, New York, NY, USA. AC-M.

Farias, D. P. D. and Roy, B. V. (August 2004). On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3):462–478.

Farias, V. F. and Roy, B. V. (2006). Tetris: A study of randomized constraint sampling. *Probabilistic and Randomized Methods for Design Under Uncertainty, G. Calafiore and F. Dabbene, eds.*

Goodman, N. D., Mansighka, V. K., D. Roy, K. B., and Tenenbaum, J. B. (2008). A language for generative models. In *UAI*.

Hal Daumé, I. and Marcu, D. (2005). Learning as search optimization: approximate large margin methods for structured prediction. In *ICML*, pages 169–176, New York, NY, USA. ACM.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1771–1800.

Jain, S. and Neal, R. M. (2004). A split-merge markov chain monte carlo procedure for the dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13:158–182.

McCallum, A., Rohanimanesh, K., Wick, M., Schultz, K., and Singh, S. (2009). Factorie:probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems(NIPS)*, Vancouver, BC, Canda.

Milch, B., Marthi, B., and Russell, S. (2006). *BLOG: Relational Modeling with Unknown Objects*. PhD thesis, University of California, Berkeley.

Richardson, M. and Domingos, P. (2006). Markov logic networks. *Machine Learning*, 62:107–136.

Rohanimanesh, K., Wick, M., and McCallum, A. (2009). Inference and learning in large factor graphs with adaptive proposal distributions. Technical Report #UM-CS-2009-028, University of Massachusetts, Amherst.

Tieleman, T. (2008). Training restricted boltzmann machines using approximations to the likelihood gradient. In *ICML*, pages 1064–1071, New York, NY, USA. ACM.