Faster Training of Structural SVMs with Diverse M-Best Cutting-Planes

Abner Guzman-Rivera
University of Illinois
aguzman5@illinois.edu

Pushmeet Kohli
Microsoft Research Cambridge
pkohli@microsoft.com

Dhruv Batra
Virginia Tech
dbatra@vt.edu

Abstract

Training of Structural SVMs involves solving a large Quadratic Program (QP). One popular method for solving this optimization problem is a cutting-plane approach, where the most violated constraint is iteratively added to a working-set of constraints. Unfortunately, training models with a large number of parameters remains a time consuming process. This paper shows that significant computational savings can be achieved by generating and adding multiple highly violated constraints at every iteration of the training algorithm. We show that generation of such diverse M-Best cutting-planes involves extracting diverse M-Best solutions from the loss-augmented score function. Our experiments on image segmentation and protein side-chain prediction show that the proposed approach can lead to significant computational savings, *e.g.*, $> 60\%$ reduction in the number of training iterations. Finally, our results suggest that even greater savings are possible for more expressive models involving a larger number of parameters.

1 Introduction

A number of problems in Computer Vision, Natural Language Processing and Computational Biology involve making predictions over complex but structured interdependent outputs – *e.g.*, all possible segmentations of an image or all possible English translations of a Chinese sentence. Formulations like Max-Margin Markov Networks (M^3N) [16] and Structured Support Vector Machines (SSVMs) [17] have provided principled techniques for for learning such structured-output models.

In all these settings, the learning algorithm has access to n training (input-output) pairs: $\{(\mathbf{x}_i, \mathbf{y}_i) \mid \mathbf{x}_i \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y}\}$ and the goal is to learn a mapping $f : \mathcal{X} \to \mathcal{Y}$ from the input space $\mathcal{X}$ to the output space $\mathcal{Y}$, such that it minimizes a (regularized) task-dependent loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$, where $\ell(\mathbf{y}_i, \bar{\mathbf{y}}_i)$ denotes the cost of predicting output $\bar{\mathbf{y}}_i$ when the correct prediction is $\mathbf{y}_i$.

Cutting-Plane Training. This learning problem is generally formulated as a constrained Quadratic Program (QP) [9, 17] with exponentially many constraints. For instance, 1-slack SSVMs [9] involve $|\mathcal{Y}|^n$ constraints, one for each possible n-tuple of labels $(\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_n) \in \mathcal{Y}^n$. If the most violated constrained can be identified efficiently, a cutting-plane approach [10] may be used to solve this QP.

A cutting-plane algorithm maintains a working-set of constraints and alternates between: 1) solving for the optimum solution under the current working-set and, 2) adding the most violated constraint to the working-set by calling the max-violation-oracle. Unfortunately, models for many real world problems have a large number of parameters and require many such alternations.

Contribution. This paper shows that significant computational savings can be achieved in training SSVMs by generating and adding a diverse set of highly violated constraints (cutting-planes) at every training iteration. Fig. 1 illustrates the idea. One key observation of our work is that for multiple constraints to be useful and speed up convergence, they should satisfy the following desiderata:

1. **Marginal Relevance.** Each constraint should be relevant and informative w.r.t. the current approximation (*i.e.*, be highly violated) and also have *marginal* relevance over the other

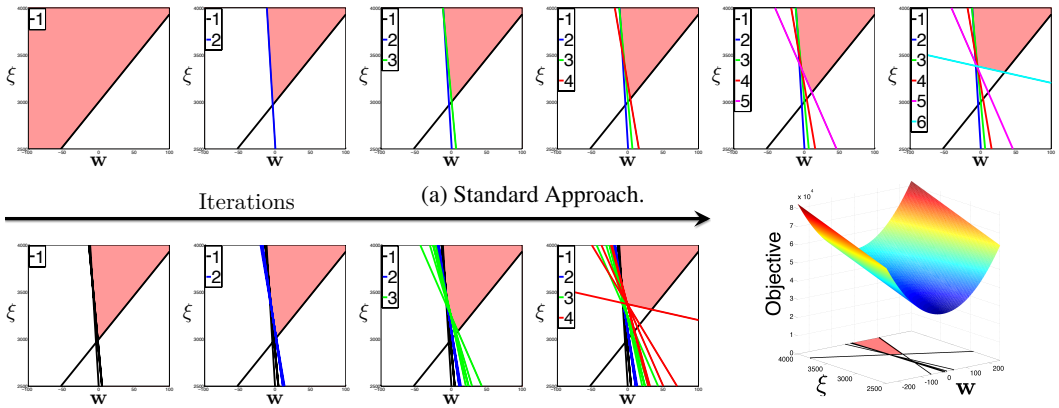


Figure 1: Illustration of the proposed approach with the feasibility region approximation at the current iteration in pink. (a) When adding a single cutting-plane at every iteration, 6 iterations are necessary to approximate the feasibility region to the desired precision. (b) In contrast, by adding 5 cutting-planes at every iteration only 4 iterations are necessary. (c) Depiction of the learning QP and approximated feasibility region.

constraints being added at the current iteration (*i.e.*, we need a *diverse* set of constraints).

2. **Efficiently Computable.** Finding a set of violated constraints should be fast enough so as to not offset the savings resulting from the reduction in the number of training iterations.

We show that generating sets of constraints (*i.e.*, cutting-planes) satisfying the above desiderata involves extracting diverse M-Best solutions from the loss-augmented score – for this, we use an efficient algorithm [4]. Our experiments on image segmentation and protein side-chain prediction show that the proposed approach can lead to significant computational savings, *e.g.*, $> 60\%$ reduction in the number of training iterations. Finally, our results suggest that even greater savings are possible for more expressive models involving a larger number of parameters.

2 Preliminaries: Training SSVMs

This section establishes notation and revisits cutting-plane training of structural SVMs.

Notation. Let $[n]$ be the shorthand for the set $\{1, 2, \ldots, n\}$. We use $\mathbf{y}$ to denote a structured output and $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_{|\mathbf{Y}|})$ for a tuple of structured outputs.

Given a training dataset of input-output pairs $S = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n) \,|\, \mathbf{x}_i \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y}\}$, we are interested in learning a mapping $f : \mathcal{X} \to \mathcal{Y}$ from an input space $\mathcal{X}$ to a structured output space $\mathcal{Y}$, where $|\mathcal{Y}|$ is finite but typically exponentially large (*e.g.*, the set of all segmentations of an image).

Structured Support Vector Machines (SSVMs). In an SSVM setting, the mapping is defined as $f(x) = \operatorname{argmax}_{y \in \mathcal{Y}} \mathbf{w}^T \boldsymbol{\Psi}(\boldsymbol{x}, \mathbf{y})$, where $\boldsymbol{\Psi}(\boldsymbol{x}, \mathbf{y})$ is a joint feature map: $\boldsymbol{\Psi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$. The quality of the prediction $\hat{\mathbf{y}}_i = f(\mathbf{x}_i)$ is measured by a task-specific loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$, where $\ell(\mathbf{y}_i, \hat{\mathbf{y}}_i)$ denotes the cost of predicting $\hat{\mathbf{y}}_i$ when the correct label is $\mathbf{y}_i$. Since the task-loss ℓ is non-continuous and non-convex in $\mathbf{w}$, typically a convex surrogate loss that upper bounds ℓ is optimized instead (*e.g.*, the hinge upper-bound [17]).

Optimization Problem 1 (OP1). The regularized hinge-loss SSVM learning problem can be formulated as a QP with exponentially many constraints. In this paper, we work with the margin-rescaling variant of the 1-slack formulation of Joachims *et al.* [9]:

$$\min_{\mathbf{w}, \xi \geq 0} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi \tag{1a}$$

$$s.t. \quad \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} \Big[\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) - \boldsymbol{\Psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i)\Big] \geq \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi \quad \forall \bar{\mathbf{Y}} \in \mathcal{Y}^n \tag{1b}$$

Note that 1-slack SSVMs involve $|\mathcal{Y}|^n$ constraints, one for each possible n-tuple of labels $\bar{\mathbf{Y}} = (\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_n) \in \mathcal{Y}^n$, but there is a single slack variable ξ shared across all constraints.

Cutting-Plane Training of SSVMs. Algorithm 1 provides a cutting-plane approach to solving OP1. At every iteration, the algorithm computes the solution over the current working-set $\mathcal{W}$ (Line 4) and

Algorithm 1 Training Structural SVMs (margin-rescaling) via the 1-Slack Formulation OP1.

1: Input: $S = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, C, ϵ
2: $\mathcal{W} \leftarrow \emptyset$
3: **repeat**
4: $\quad (\mathbf{w}, \xi) \leftarrow \operatorname{argmin}_{\mathbf{w}, \xi \geq 0} \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi$
$\quad\quad s.t. \quad \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} \left[\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) - \boldsymbol{\Psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i)\right] \geq \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi_i \quad \forall \bar{\mathbf{Y}} \in \mathcal{W}$
5: $\quad$ **for** $i = 1, \ldots, n$ **do**
6: $\quad\quad \hat{\mathbf{y}}_i \leftarrow \operatorname{argmax}_{\mathbf{y}} \left\{\ell(\mathbf{y}_i, \mathbf{y}) + \mathbf{w}^T\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y})\right\}$
7: $\quad$ **end for**
8: $\quad \mathcal{W} \leftarrow \mathcal{W} \cup \{(\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_n)\}$
9: **until** $\frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{y}_i, \hat{\mathbf{y}}_i) - \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} \left[\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) - \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i)\right] \leq \xi + \epsilon$
10: **return** $(\mathbf{w}, \xi)$

then finds the most violated constraint (Lines 5-7) to add to $\mathcal{W}$ (Line 8). The algorithm stops when the most violated constraint is violated less than a desired precision ϵ (Line 9).

Joachims *et al*. [9] showed that Algorithm 1 converges in polynomial time, in fact the number of iterations is independent of the size of the training-set. Formally:

Theorem 1. *Iteration Complexity of Algorithm 1. For any $0 < C$, $0 < \epsilon < 4R^2C$ and any training sample $S = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, Algorithm 1 terminates after at most*

$$\left\lceil \log_2\left(\frac{\ell}{4R^2C}\right) \right\rceil + 2\left\lceil \frac{8R^2C}{\epsilon} \right\rceil$$

iterations, where $R^2 = \max_{i,\bar{\mathbf{y}}} \left\|\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) - \boldsymbol{\Psi}(\mathbf{x}_i, \bar{\mathbf{y}})\right\|^2$ and $\ell = \max_{i,\bar{\mathbf{y}}} \ell(\mathbf{y}_i, \bar{\mathbf{y}})$.

Proof. See proof of Theorem 5 in [9]. □

3 Proposed Approach

An intuitive approach to speed up Algorithm 1 is to add multiple constraints in each iteration. We will show that significant computational savings are possible if the additional constraints are highly violated and diverse. Finding the most violated constraint is equivalent to performing MAP inference on a loss-augmented score for each training instance. Similarly, finding multiple violated constraints involves generating multiple diverse solutions to the loss-augmented score of the training instances.

Techniques for producing multiple solutions in probabilistic models can be broadly characterized into two groups: M-best MAP algorithms [7, 13, 14, 19] that find the top M most probable solutions and Sampling-based algorithms [2, 15, 18]. Both of these groups fall short for our task. M-Best MAP algorithms do not place any emphasis on diversity and tend to produce solutions that are minor perturbations of each other. Thus, the resulting cutting-planes are unlikely to be of much value in speeding up convergence. Sampling-based approaches typically exhibit long wait-times to transition from one mode to another, which is required for obtaining diversity.

3.1 Generating Diverse M-Best Solutions on Loss-Augmented Score

To enforce diversity, we use the DivMBest algorithm of Batra *et al*. [4], which computes a set of diverse M-Best solutions in discrete probabilistic models. We briefly describe their approach here.

For the sake of illustration, consider a discrete Markov Random Field (MRF). Specifically, let $\mathbf{y} = \{y_1, \ldots, y_p\} \in \mathcal{Y}$ be a set of discrete random variables, each taking value in a finite label set, *i.e.*, $y_u \in \mathcal{Y}_u$. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph defined over the output variables, *i.e.*, $\mathcal{V} = [p]$, $\mathcal{E} \subseteq \binom{\mathcal{V}}{2}$ and let y_{uv} be shorthand for the tuple (y_u, y_v). It is known that for decomposable loss functions, the loss-augmented score for any configuration $\mathbf{y}$ can be expressed as a sum of terms that decompose along the graph-structure. Thus, loss-augmented inference corresponds to a MAP inference problem:

$$\max_{\mathbf{y} \in \mathcal{Y}} \ S(\mathbf{y}) = \max_{\mathbf{y} \in \mathcal{Y}} \sum_{u \in \mathcal{V}} \theta_u(y_u) + \sum_{(u,v) \in \mathcal{E}} \theta_{uv}(y_{uv}). \tag{2}$$

Algorithm 2 Generalization of Algorithm 1 adding M constraints at every iteration.

1: Input: $S = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_y, \mathbf{y}_n)\}$, C, ϵ, M, $\mathbf{K}$, $\boldsymbol{\lambda}_0$
2: $\mathcal{W} \leftarrow \emptyset$; $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda}_0$
3: **repeat**
4: $\quad (\mathbf{w}, \xi) \leftarrow \operatorname{argmin}_{\mathbf{w}, \xi \geq 0} \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\xi$
$\quad\quad s.t. \quad \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} \left[\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) - \boldsymbol{\Psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i)\right] \geq \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi_i \quad \forall \bar{\mathbf{Y}} \in \mathcal{W}$
5: $\quad$ **for** $i = 1, \ldots, n$ **do**
6: $\quad\quad \tilde{\mathbf{Y}}_i = (\tilde{\mathbf{y}}_i^{(1)}, \ldots, \tilde{\mathbf{y}}_i^{(M)}) \leftarrow \text{DivMBest}(\ell(\mathbf{y}_i, \cdot) + \mathbf{w}^T\Psi(\mathbf{x}_i, \cdot), M, \boldsymbol{\lambda})$
7: $\quad$ **end for**
8: $\quad \boldsymbol{\lambda} \leftarrow \text{Update}\boldsymbol{\lambda}(M, \mathbf{K}, \boldsymbol{\lambda}, \tilde{\mathbf{Y}}_1, \ldots, \tilde{\mathbf{Y}}_n)$
9: $\quad (\hat{\mathbf{Y}}^{(1)}, \ldots, \hat{\mathbf{Y}}^{(M)}) \leftarrow \text{Combine}(M, \tilde{\mathbf{Y}}_1, \ldots, \tilde{\mathbf{Y}}_n)$
10: $\quad \mathcal{W} \leftarrow \mathcal{W} \cup \left\{\hat{\mathbf{Y}}^{(1)}, \ldots, \hat{\mathbf{Y}}^{(M)}\right\}$
11: **until** $\frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{y}_i, \hat{\mathbf{y}}_i^{(1)}) - \frac{1}{n}\mathbf{w}^T \sum_{i=1}^{n} \left[\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) - \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(1)})\right] \leq \xi + \epsilon$
12: **return** $(\mathbf{w}, \xi)$

We assume availability of a function $\Delta(\tilde{\mathbf{y}}, \tilde{\mathbf{y}}')$ quantifying dissimilarity between solutions $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{y}}'$. Let $\tilde{\mathbf{y}}^{(m)}$ denote the m^{th}-best solution. Thus, $\tilde{\mathbf{y}}^{(1)}$ is the MAP, $\tilde{\mathbf{y}}^{(2)}$ is the second DivMBest solution and so on. Batra *et al*. [4] proposed the following formulation for finding the m^{th} solution:

$$\tilde{\mathbf{y}}^{(m)} = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \sum_{u \in \mathcal{V}} \theta_u(y_u) + \sum_{(u,v) \in \mathcal{E}} \theta_{uv}(y_{uv}) \tag{3a}$$

$$s.t. \quad \Delta(\mathbf{y}, \tilde{\mathbf{y}}^{(m')}) \geq k_{m'} \quad \forall m' \in [m-1] \tag{3b}$$

In order to solve this problem, [4] proposed to to use the Lagrangian relaxation of (3), formed by *dualizing* the dissimilarity constraints $\Delta(\mathbf{y}, \tilde{\mathbf{y}}^{(m')}) \geq k_{m'}$:

$$f(\boldsymbol{\lambda}) = \max_{\mathbf{y} \in \mathcal{Y}} S_\Delta(\mathbf{y}) = \max_{\mathbf{y} \in \mathcal{Y}} \sum_{A \in \mathcal{V} \cup \mathcal{E}} \theta_A(y_A) + \sum_{m'=1}^{m-1} \lambda_{m'} \left(\Delta(\mathbf{y}, \tilde{\mathbf{y}}^{(m')}) - k_{m'}\right) \tag{4}$$

We see that the Lagrangian relaxation maximizes a Δ-augmented score. For some classes of Δ-functions (*e.g.*, Hamming Dissimilarity), we can solve the Δ-augmented score maximization problem using the *same algorithms* used for finding the MAP.

3.2 Generating Diverse M-Best Cutting-Planes

Our proposed approach is summarized in Algorithm 2 and is parametrized by M, the number of constraints to add to the working-set at every iteration – note that Algorithm 1 is a special case of Algorithm 2 with M=1. Algorithm 2 finds M diverse loss-augmented solutions for each example (Line 6) and uses these solutions to generate diverse M-Best cutting-planes to be added to the working-set (Lines 9-10). Update$\boldsymbol{\lambda}$ (Line 8) controls the amount of diversity in the loss-augmented solutions while Combine (Line 9) produces a set of M cutting-planes given the M loss-augmented solutions from all n examples.

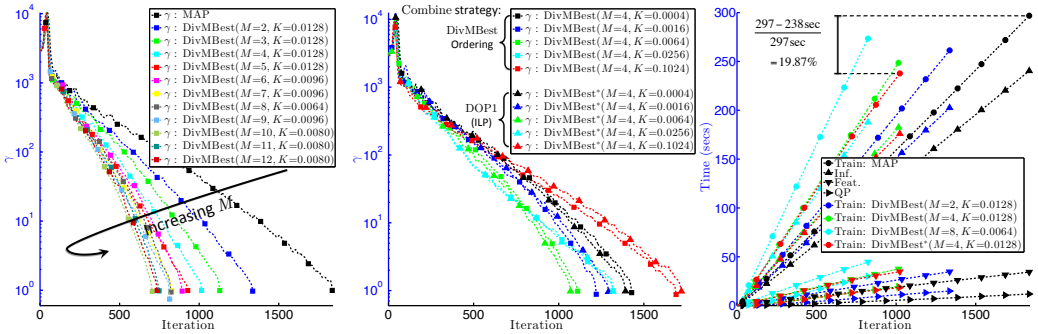
Diversity Requirements (Update$\boldsymbol{\lambda}$**).** Note that the amount of diversity in the loss-augmented solutions is controlled by parameter $k_{m'}$ in (3b). However, the amount of diversity appropriate for faster convergence is problem dependent and not known a priori.

To address this issue, Algorithm 2 adaptively controls the amount of diversity in the solutions. Let $\bar{K}_j$ be the *cumulated-diversity* observed in all $(j+1)^{th}$ solutions w.r.t. to all j^{th} solutions,

$$\bar{K}_j = \frac{1}{\alpha_\Delta} \sum_{i=1}^{n} \Delta(\tilde{\mathbf{y}}_i^{(j)}, \tilde{\mathbf{y}}_i^{(j+1)}) \tag{5}$$

where Δ is the dissimilarity function used by DivMBest and α_Δ is a normalization constant.

Then, $\mathbf{K}$ functions as a setpoint for procedure Update$\boldsymbol{\lambda}$ which evaluates the actual cumulated-diversity obtained at the current iteration and updates $\boldsymbol{\lambda}$ (the vector of Lagrange multipliers) to increase or decrease diversity at the next iteration.



(a) γ vs it. @ multiple M. (b) γ vs it. @ multiple K. (c) Time vs it. @ multiple M.

Figure 2: Violation of most violated constraint, γ, and execution times vs iterations until convergence for Algorithm 2 on foreground-background segmentation.

Combining Solutions into Constraints (Combine)**.** To generate M 1-slack constraints we need to combine features corresponding to solutions from all training examples – given M DivMBest solutions for each example, there are M^n possible constraints. Further, one must attend to the diversity of the resulting cutting-planes w.r.t. each other: Features appearing together in a constraint must be such that their "diversities" do not cancel-out. Additionally, in order to preserve the correctness and convergence properties of the original algorithm, our approach lets $\hat{\mathbf{Y}}^{(1)}$ correspond to the standard most violated constraint. For the the remaining $M-1$ additional cutting-planes, we explore the following choices:

1. DivMBest−Ordering: $\hat{\mathbf{Y}}^{(j)} \leftarrow (\tilde{\mathbf{y}}_1^{(j)}, \ldots, \tilde{\mathbf{y}}_n^{(j)})$ for $j \in [M]$. That is, we combine all m^{th} solutions together to obtain the m^{th} constraint.
2. DOP1−ILP (Heuristic): Informed by insight in the proof of Theorem 1, this strategy involves an optimization procedure that seeks to maximize the attainable increase in the objective of the Dual of OP1 given the new constraints. For lack of space, we relegate details to the extended version of this paper [8].

4 Experiments

Setup. We tested Algorithm 2 on image foreground-background segmentation and protein side-chain prediction. For both problems we tuned parameter C on validation data. We performed grid-search on $\mathbf{K}$ and found the algorithm to be fairly robust to the choice of $\boldsymbol{\lambda}_0$.

Our experiments show that the number of cutting-plane iterations can be reduced substantially, *i.e.*, up to $\sim$62% in the case of foreground-background segmentation. However, given the overhead of additional constraint generation, the greatest (time) speedup, $\sim$28%, was obtained under a more modest reduction, $\sim$33%, in the number of iterations.

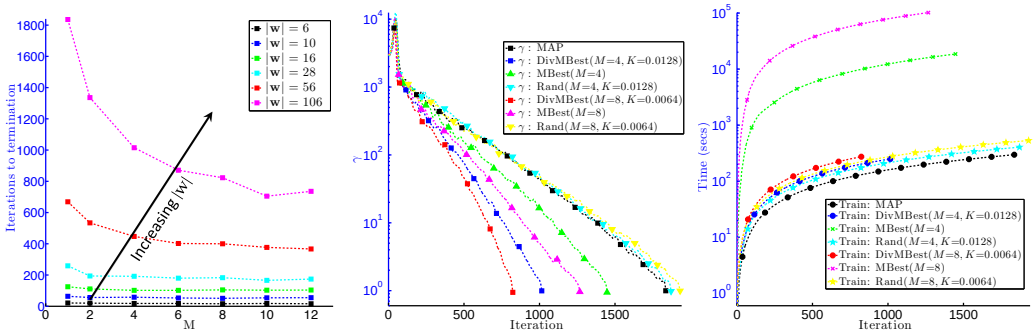
Baselines are obtained by replacing the oracle call (Line 6) in Algorithm 2 with: 1) MAP inference. 2) MBest MAP inference. 3) Rand: A sampling procedure that relabels nodes via local sampling while observing the cumulated-diversity specified by setpoint $\mathbf{K}$.

4.1 Foreground-Background Segmentation

Dataset. We used the co-segmentation dataset, iCoseg, of Batra *et al*. [3]. iCoseg consists of 37 groups of related images mimicking typical consumer photograph collections. Each group may be thought of as an "event" (*e.g.*, a baseball game, a safari, *etc*.). The dataset provides pixel-level ground-truth foreground-background segmentations for each image.

Model and Features. The task is modeled by a binary pairwise MRF where each node corresponds to a superpixel [1] in the image. We extracted up to 51 features at each superpixel. Edge features were computed for each pair of adjacent superpixels and correspond to a standard Potts model and a contrast sensitive Potts model. Edge weights were constrained so as to obtain supermodular potentials. Thus, the resulting models are amenable to inference with graph-cuts [5, 12].

Effect of M on convergence. Let γ be the amount by which the most violated constraint is violated, *e.g.*, the algorithm stops when $\gamma \leq \epsilon$. Fig. 2a plots γ vs iterations for a range of values of M. We



(a) Effect of M @ multiple $|\mathbf{w}|$. (b) γ vs it. @ baselines. (c) Time vs it. @ baselines.

Figure 3: (a) Effect of M on number of iterations to termination for different problem dimensionalities. (b-c) Convergence and execution time vs iterations against baselines on foreground-background segmentation.

observe that the greatest reduction in the number of iterations, $\frac{1836-705}{1836} \approx 61.60\%$, was achieved for $M{=}10$ and note that a further increase in M resulted in a slight reversal of gains.

Effect of K on convergence. Fig. 2b shows the effect of the diversity setpoint $\mathbf{K}$ on convergence. We observe that a mid-range value leads to fastest convergence.

This plot also compares two of the feature combination strategies suggested before. We note that the simple $\mathrm{DivMBest}{-}\mathrm{Ordering}$ strategy performs comparably to the more complex $\mathrm{DOP1}{-}\mathrm{ILP}$.

Convergence time. Fig. 2c compares execution times for different values of M. We plot total train time as well as the time contributions of inference (*e.g.*, DivMBest); feature computation; and QP optimization. An "annealed" execution, which lowers $\mathbf{K}$ and M during execution, is also included in the plot (starred curve). This curve obtained the greatest speedup with an iteration reduction of $\frac{1836-1023}{1836} \approx 44.28\%$ and a running time reduction of $\frac{297-238\ secs}{297\ secs} \approx 19.87\%$.

Effect of Problem Dimensionality, $|\mathbf{w}|$. We investigate the behavior of the proposed approach as the number of features in the learning problem is varied. Fig. 3a shows that as the dimensionality of the problem $|\mathbf{w}|$ increases, a higher value of M continues to produce iteration reductions. This suggest we may expect greater computational savings on problems of higher dimensionality.

Comparison against Baselines. In Fig. 3b we see that Rand produced an increase in the number of training iterations. MBest obtained about half the decrease in the number of iterations obtained by DivMBest. However, Fig. 3c reveals that MBest is close to *three orders of magnitude* slower.

4.2 Protein Side-Chain Prediction

Model and Dataset. Given a protein backbone structure, the task here is to predict the amino acid side-chain configurations. We used the dataset and modeling of [6] and TRW-S [11] for inference. Further details and plots may be found in the extended version of the paper [8].

Results. The greatest speedup was obtained for $M{=}4$ with an iteration reduction of $\frac{102-68}{102} \approx 33.33\%$ and a running time reduction of $\frac{14749-10585\ secs}{14749\ secs} \approx 28.23\%$.

5 Conclusions

We investigated the effect of adding multiple highly violated constraints in the context of cutting-plane training of structural SVMs. We noted that significant improvements in the convergence of the training algorithm are possible if the added constraints are: 1) highly violated, 2) diverse and, 3) efficiently computable.

We presented an an efficient algorithm for generating such constraints. Our experiments show that the proposed method leads to a significant, $>60\%$, reduction in the number of iterations to convergence. While a more modest improvement in training time was achieved, $\sim 24\%$, there are reasons to anticipate greater speedups in applications with higher feature dimensionality and also as we move towards multi-core architectures.

Acknowledgments: AGR was supported by the C2S2 Focus Center (under the SRC's Focus Center Research Program).

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Ssstrunk. SLIC Superpixels Compared to State-of-the-art Superpixel Methods. *PAMI*, (To Appear) 2012. 5

[2] A. Barbu and S.-C. Zhu. Generalizing Swendsen-Wang to Sampling Arbitrary Posterior Probabilities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27:1239–1253, August 2005. 3

[3] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen. iCoseg: Interactive Co-segmentation with Intelligent Scribble Guidance. In *CVPR*, 2010. 5

[4] D. Batra, P. Yadollahpour, A. Guzman-Rivera, and G. Shakhnarovich. Diverse M-Best Solutions in Markov Random Fields. In *ECCV*, 2012. 2, 3, 4

[5] Y. Boykov, O. Veksler, and R. Zabih. Efficient Approximate Energy Minimization via Graph Cuts. *PAMI*, 20(12):1222–1239, 2001. 5

[6] O. S.-F. Chen Yanover and Y. Weiss. Minimizing and Learning Energy Functions for Side-Chain Prediction. *Journal of Computational Biology*, 15(7):899–911, 2008. 6

[7] M. Fromer and A. Globerson. An LP View of the M-best MAP problem. In *NIPS*, 2009. 3

[8] A. Guzman-Rivera, P. Kohli, and D. Batra. Faster Training of Structural SVMs with Diverse M-Best Cutting-Planes, 2013. Technical Report. http://ttic.uchicago.edu/~dbatra/publications/assets/gkb_divmcuts.pdf. 5, 6

[9] T. Joachims, T. Finley, and C.-N. Yu. Cutting-Plane Training of Structural SVMs. *Machine Learning*, 77(1):27–59, 2009. 1, 2, 3

[10] J. Kelley, J. E. The Cutting-Plane Method for Solving Convex Programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):pp. 703–712, 1960. 1

[11] V. Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *PAMI*, 28(10):1568–1583, 2006. 6

[12] V. Kolmogorov and R. Zabih. What Energy Functions can be Minimized via Graph Cuts? *PAMI*, 26(2):147–159, 2004. 5

[13] E. R. Natalia Flerova and R. Dechter. Bucket and mini-bucket Schemes for M Best Solutions over Graphical Models. In *IJCAI Workshop on Graph Structures for Knowledge Representation and Reasoning*, 2011. 3

[14] D. Nilsson. An efficient algorithm for finding the M most probable configurations in probabilistic expert systems. *Statistics and Computing*, 8:159–173, 1998. 10.1023/A:1008990218483. 3

[15] J. Porway and S.-C. Zhu. C^4: Exploring Multiple Solutions in Graphical Models by Cluster Sampling. *PAMI*, 33(9):1713 –1727, 2011. 3

[16] B. Taskar, C. Guestrin, and D. Koller. Max-Margin Markov Networks. In *NIPS*, 2003. 1

[17] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large Margin Methods for Structured and Interdependent Output Variables. *JMLR*, 6:1453–1484, 2005. 1, 2

[18] Z. Tu and S.-C. Zhu. Image Segmentation by Data-Driven Markov Chain Monte Carlo. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24:657–673, May 2002. 3

[19] C. Yanover and Y. Weiss. Finding the M Most Probable Configurations Using Loopy Belief Propagation. In *NIPS*, 2003. 3