Supplementary Material for “DivMCuts: Faster Training of Structural SVMs with Diverse M-Best Cutting-Planes”

Abner Guzman-Rivera
University of Illinois

Pushmeet Kohli
Microsoft Research Cambridge

Dhruv Batra
Virginia Tech

1 Preliminaries: Training Structural SVMs

This section reviews the notation used in the main paper and revisits cutting-plane methods for training structured-output predictors.

Notation. For any positive integer n we use $[n]$ as shorthand for the set $\{1, 2, \ldots, n\}$. We use $\mathbf{y}$ for 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 $\{(\mathbf{x}_i, \mathbf{y}_i) \,|\, \mathbf{x}_i \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y}\}$, we are interested in learning a mapping $f : \mathcal{X} \to \mathcal{Y}$ from an input space $\mathcal{X}$ to a structured output space $\mathcal{Y}$ that is finite but typically exponentially large (*e.g.*, the set of all segmentations of an image, or all English translations of a Chinese sentence).

Structural Support Vector Machines (SSVMs). In an SSVM setting, the mapping is defined as $f(x) = \text{argmax}_{y \in \mathcal{Y}} \mathbf{w}^T \mathbf{\Psi}(\boldsymbol{x}, \mathbf{y})$, where $\mathbf{\Psi}(\boldsymbol{x}, \mathbf{y})$ is a joint feature map: $\mathbf{\Psi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$. 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 typically non-convex and non-continuous, [2] proposed to optimize the hinge upper bound on ℓ.

The regularized hinge-loss SSVM learning problem can be formulated as a QP with exponentially many constraints. In this paper, we work with the 1-slack formulation of Joachims *et al.* [1] in the Margin-Rescaling variant (1).

Optimization Problem 1 (OP1). 1-slack Structural SVM (Margin-Rescaling) Training (Primal) formulation,

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

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

Note that 1-slack SSVMs involve $|\mathcal{Y}|^n$ constraints, one for each possible combination 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. Hence, the name 1-slack. The number of constraints is thus *exponentially* larger than in n-slack SSVMs (which involve only $n|\mathcal{Y}|$ constraints). However, Joachims *et al.* [1] showed that: 1) the two formulations are equivalent and, most importantly, 2) 1-slack leads to faster convergence, both in theory and practice.

The cutting-plane algorithm (Algorithm 1 in the main paper) for solving OP1 relies on the fact that the number of non-zero elements of the solution $\boldsymbol{\alpha}$ of the dual problem of OP1 is independent of the size of the training set. This key property is used for proving convergence of the cutting-plane algorithm. The Dual of OP1 is [1],

Optimization Problem 2 (DOP1).

$$\max_{\boldsymbol{\alpha}\geq 0} \quad D(\boldsymbol{\alpha}) = \sum_{\bar{\mathbf{Y}}\in\mathcal{Y}^n} \alpha_{\bar{\mathbf{Y}}}\ell(\bar{\mathbf{Y}}) - \frac{1}{2}\sum_{\bar{\mathbf{Y}}\in\mathcal{Y}^n}\sum_{\bar{\mathbf{Y}}'\in\mathcal{Y}^n} \alpha_{\bar{\mathbf{Y}}}\alpha_{\bar{\mathbf{Y}}'}H(\bar{\mathbf{Y}},\bar{\mathbf{Y}}') \tag{2a}$$

$$s.t. \quad \sum_{\bar{\mathbf{Y}}\in\mathcal{Y}^n} \alpha_{\bar{\mathbf{Y}}} = C \tag{2b}$$

where $\ell(\bar{\mathbf{Y}}) = \frac{1}{n}\sum\limits_{i=1}^{n}\ell(\mathbf{y}_i, \bar{\mathbf{y}}_i)$, and

$$H(\bar{\mathbf{Y}},\bar{\mathbf{Y}}') = \frac{1}{n^2}\Bigg[\sum_{i=1}^{n}\sum_{j=1}^{n}\boldsymbol{\Psi}(\mathbf{x}_i,\mathbf{y}_i)^T\boldsymbol{\Psi}(\mathbf{x}_j,\mathbf{y}_j) - \sum_{i=1}^{n}\sum_{j=1}^{n}\boldsymbol{\Psi}(\mathbf{x}_i,\mathbf{y}_i)^T\boldsymbol{\Psi}(\mathbf{x}_j,\bar{\mathbf{y}}_j) - \sum_{i=1}^{n}\sum_{j=1}^{n}\boldsymbol{\Psi}(\mathbf{x}_i,\mathbf{y}_i)^T\boldsymbol{\Psi}(\mathbf{x}_j,\bar{\mathbf{y}}'_j) + \sum_{i=1}^{n}\sum_{j=1}^{n}\boldsymbol{\Psi}(\mathbf{x}_i,\bar{\mathbf{y}}_i)^T\boldsymbol{\Psi}(\mathbf{x}_j,\bar{\mathbf{y}}'_j)\Bigg]. \tag{3}$$

The primal and dual solutions, $\mathbf{w}^*$ and $\boldsymbol{\alpha}^*$ respectively, are related by

$$\mathbf{w}^* = \frac{1}{n}\sum_{\bar{\mathbf{Y}}\in\mathcal{Y}^n}\alpha^*_{\bar{\mathbf{Y}}}\sum_{j=1}^{n}\big[\boldsymbol{\Psi}(\mathbf{x}_j,\mathbf{y}_j) - \boldsymbol{\Psi}(\mathbf{x}_j,\bar{\mathbf{y}}_j)\big].$$

2 On Alternative 1-Slack Constraint Generation Strategies

A 1-slack constraint is uniquely determined by a tuple of solutions for all examples, *e.g.*, $\hat{\mathbf{Y}} = (\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_n)$,

$$\frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{y}_i,\hat{\mathbf{y}}_i) - \frac{1}{n}\mathbf{w}^T\sum_{i=1}^{n}\Big[\boldsymbol{\Psi}(\mathbf{x}_i,\mathbf{y}_i) - \boldsymbol{\Psi}(\mathbf{x}_i,\hat{\mathbf{y}}_i)\Big] \leq \xi \tag{4}$$

Given tuples $\tilde{\mathbf{Y}}_i = (\tilde{\mathbf{y}}_i^{(1)}, \ldots, \tilde{\mathbf{y}}_i^{(M)})$ of diverse solutions for all examples, we need to decide which solutions to combine with each other in order to generate multiple 1-slack constraints. Obviously, there are an exponential number of possibilities.

The proof of Theorem 1 in the main paper suggests a number of ideas one could try. We describe in detail one of the heuristics we experimented with which worked well in practice. We seek to maximize the attainable increase of the dual objective (2a) after adding a set of M new constraints.

We keep only the terms in the dual objective (2a) that are dependent on the new constraints, *i.e.*, we would like to maximize the following expression (w.r.t. the new constraints),

$$\sum_{m}\alpha_{\hat{\mathbf{Y}}^{(m)}}\ell(\hat{\mathbf{Y}}^{(m)}) - \sum_{\bar{\mathbf{Y}}\in\mathcal{W}}\sum_{m}\alpha_{\bar{\mathbf{Y}}}\alpha_{\hat{\mathbf{Y}}^{(m)}}H(\bar{\mathbf{Y}},\hat{\mathbf{Y}}^{(m)}) - \frac{1}{2}\sum_{m,m'}\alpha_{\hat{\mathbf{Y}}^{(m)}}\alpha_{\hat{\mathbf{Y}}^{(m')}}H(\hat{\mathbf{Y}}^{(m)},\hat{\mathbf{Y}}^{(m')}) \tag{5}$$

where $m, m' \in [M]$, $\mathcal{W}$ is the working-set at the current iteration and $\hat{\mathbf{Y}}^{(m)}$ is the tuple of new constraints.

Of course, we do not know the value $\boldsymbol{\alpha}$ will take after the addition of the new constraints but we must somehow fix it in order to proceed. We set $\boldsymbol{\alpha}$ as follows,

$$\alpha_{\mathbf{Y}} = \frac{C}{Z}\begin{cases} \alpha_{\mathbf{Y}} & \text{if } \mathbf{Y}\in\mathcal{W} \\ \tilde{\alpha}_{\mathcal{W}} & \text{if } \mathbf{Y} \text{ is a new constraint} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $\tilde{\alpha}_{\mathcal{W}}$ is the median of $\{\alpha_{\mathbf{Y}} \mid \mathbf{Y} \in \mathcal{W}\}$ and $\frac{C}{Z}$ is a normalization constant so that 2b is satisfied (normalization, however, does not affect the optimization procedure that follows). Further, we would like for $\hat{\mathbf{Y}}^{(1)}$ to be the most-violated constraint (in order to preserve the theoretical properties of the original algorithm, *e.g.*, Theorem 1). That is, $\hat{\mathbf{Y}}^{(1)} = (\hat{\mathbf{y}}_1^{(1)}, \ldots, \hat{\mathbf{y}}_n^{(1)})$. After fixing $\boldsymbol{\alpha}$ and $\hat{\mathbf{Y}}^{(1)}$, and discarding some constant terms, eq. (5) becomes,

$$\sum_{m \in [M]_-} \tilde{\alpha}_{\mathcal{W}} \ell(\hat{\mathbf{Y}}^{(m)}) - \sum_{\bar{\mathbf{Y}} \in \mathcal{W}} \sum_{m \in [M]_-} \alpha_{\bar{\mathbf{Y}}} \tilde{\alpha}_{\mathcal{W}} H(\bar{\mathbf{Y}}, \hat{\mathbf{Y}}^{(m)}) - \sum_{m \in [M]_-} \tilde{\alpha}_{\mathcal{W}}^2 H(\hat{\mathbf{Y}}^{(1)}, \hat{\mathbf{Y}}^{(m)})$$
$$- \frac{1}{2} \sum_{m, m' \in [M]_-} \tilde{\alpha}_{\mathcal{W}}^2 H(\hat{\mathbf{Y}}^{(m)}, \hat{\mathbf{Y}}^{(m')}) \tag{7}$$

where $[M]_- = \{2, 3, \ldots, M\}$.

Referring to eq. (3) we will now expand the $H(\cdot, \cdot)$'s and drop additional terms not dependent on the new constraints. We define two symbols in the process,

$$\boldsymbol{\Psi}(\mathbf{X}, \mathbf{Y}) \triangleq \sum_{i=1}^{n} \boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) \tag{8}$$

$$\boldsymbol{\delta\Psi}(\mathbf{X}, \mathbf{Y}') \triangleq \sum_{i=1}^{n} \left[\boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i') - \boldsymbol{\Psi}(\mathbf{x}_i, \mathbf{y}_i) \right] \tag{9}$$

Hence,

$$H(\bar{\mathbf{Y}}, \hat{\mathbf{Y}}) = \sum_{i=1}^{n} \boldsymbol{\delta\Psi}(\mathbf{X}, \bar{\mathbf{Y}})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i) + \begin{matrix}\text{terms not} \\ \text{dependent on } \hat{\mathbf{Y}}\end{matrix} \tag{10}$$

$$H(\hat{\mathbf{Y}}, \hat{\mathbf{Y}}') = \sum_{i=1}^{n} \sum_{j=1}^{n} \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i)^T \boldsymbol{\Psi}(\mathbf{x}_j, \hat{\mathbf{y}}_j') - \sum_{i=1}^{n} \boldsymbol{\Psi}(\mathbf{X}, \mathbf{Y})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i)$$
$$- \sum_{i=1}^{n} \boldsymbol{\Psi}(\mathbf{X}, \mathbf{Y})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i') + \begin{matrix}\text{terms not} \\ \text{dependent on } \hat{\mathbf{Y}}, \hat{\mathbf{Y}}'\end{matrix} \tag{11}$$

Substituting the new symbols into eq. 7, dropping irrelevant terms and expanding $\ell(\hat{\mathbf{Y}})$ we arrive at,

$$\sum_{m \in [M]_-} \frac{\tilde{\alpha}_{\mathcal{W}}}{n} \sum_{i=1}^{n} \ell(\mathbf{y}_i, \hat{\mathbf{y}}_i^{(m)}) - \sum_{\bar{\mathbf{Y}} \in \mathcal{W}} \sum_{m \in [M]_-} \alpha_{\bar{\mathbf{Y}}} \tilde{\alpha}_{\mathcal{W}} \sum_{i=1}^{n} \boldsymbol{\delta\Psi}(\mathbf{X}, \bar{\mathbf{Y}})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)})$$
$$- \sum_{m \in [M]_-} \tilde{\alpha}_{\mathcal{W}}^2 \sum_{i=1}^{n} \boldsymbol{\delta\Psi}(\mathbf{X}, \hat{\mathbf{Y}}^{(1)})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)})$$
$$- \frac{1}{2} \sum_{m, m' \in [M]_-} \tilde{\alpha}_{\mathcal{W}}^2 \Bigg[\sum_{i=1}^{n} \sum_{j=1}^{n} \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)})^T \boldsymbol{\Psi}(\mathbf{x}_j, \hat{\mathbf{y}}_j^{(m')})$$
$$- \sum_{i=1}^{n} \boldsymbol{\Psi}(\mathbf{X}, \mathbf{Y})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)}) - \sum_{i=1}^{n} \boldsymbol{\Psi}(\mathbf{X}, \mathbf{Y})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m')}) \Bigg] \tag{12}$$

reordering summations and grouping terms,

$$\sum_{m\in[M]_-} \tilde{\alpha}_{\mathcal{W}} \sum_{i=1}^{n} \Big[\frac{1}{n}\ell(\mathbf{y}_i, \hat{\mathbf{y}}_i^{(m)}) - \sum_{\bar{\mathbf{Y}}\in\mathcal{W}} \alpha_{\bar{\mathbf{Y}}} \boldsymbol{\delta\Psi}(\mathbf{X}, \bar{\mathbf{Y}})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)})$$
$$- \tilde{\alpha}_{\mathcal{W}} \boldsymbol{\delta\Psi}(\mathbf{X}, \hat{\mathbf{Y}}^{(1)})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)}) + (M-1)\tilde{\alpha}_{\mathcal{W}} \boldsymbol{\Psi}(\mathbf{X}, \mathbf{Y})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)})\Big]$$
$$- \frac{1}{2} \sum_{m,m'\in[M]_-} \tilde{\alpha}_{\mathcal{W}}^2 \sum_{i=1}^{n}\sum_{j=1}^{n} \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)})^T \boldsymbol{\Psi}(\mathbf{x}_j, \hat{\mathbf{y}}_j^{(m')}) \tag{13}$$

Finally, we introduce binary variables $\rho_{i,k}^{m}$ which will take value 1 iff $\hat{\mathbf{y}}_i^{(k)} \in \hat{\mathbf{Y}}^{(m)}$, *i.e.*, the k-th solution for example i is included in constraint m. We are ready to write a binary Integer Quadratic Program (IQP) to set the $\rho_{i,k}^{m}$ variables for us,

$$\max_{\boldsymbol{\rho}} \; O(\boldsymbol{\rho}) = \sum_{m\in[M]_-} \sum_{i=1}^{n} \sum_{k\in[M]_-} \tilde{\alpha}_{\mathcal{W}} \Big[\frac{1}{n}\ell(\mathbf{y}_i, \hat{\mathbf{y}}_i^{(k)}) - \sum_{\bar{\mathbf{Y}}\in\mathcal{W}} \alpha_{\bar{\mathbf{Y}}} \boldsymbol{\delta\Psi}(\mathbf{X}, \bar{\mathbf{Y}})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(k)})$$
$$- \tilde{\alpha}_{\mathcal{W}} \boldsymbol{\delta\Psi}(\mathbf{X}, \hat{\mathbf{Y}}^{(1)})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(k)}) + (M-1)\tilde{\alpha}_{\mathcal{W}} \boldsymbol{\Psi}(\mathbf{X}, \mathbf{Y})^T \boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(k)})\Big] \rho_{i,k}^{m}$$
$$- \frac{\tilde{\alpha}_{\mathcal{W}}^2}{2} \sum_{m,m'\in[M]_-} \sum_{i=1}^{n} \sum_{k\in[M]_-} \sum_{j=1}^{n} \sum_{l\in[M]_-} \Big[\boldsymbol{\Psi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(k)})^T \boldsymbol{\Psi}(\mathbf{x}_j, \hat{\mathbf{y}}_j^{(l)})\Big] \rho_{i,k}^{m} \rho_{j,l}^{m'} \tag{14a}$$

$$s.t. \quad \sum_{k\in[M]_-} \rho_{i,k}^{m} = 1 \quad m \in [M]_-, i \in [n] \tag{14b}$$

$$\sum_{m\in[M]_-} \rho_{i,k}^{m} \leq 1 \quad k \in [M]_-, i \in [n] \tag{14c}$$

$$\rho_{i,k}^{m} \in \{0,1\} \qquad m,k \in [M]_-, i \in [n] \tag{14d}$$

Constraints (14b) indicate that a constraint must include exactly one solution for each example. Constraints (14c) indicate that solutions must appear in at most one constraint. The latter constraints are optional but we obtained better results when including them. A slight modification to the above program which we found useful is to let $k \in [M]$. That is, we allow the MAP solution to be reused once and we allow one of the solutions for each example to not be used.

Solving IQP (14) is unfortunately a time consuming process and thus not practical. For this reason, we tried the following two strategies:

1. Relaxation: We relax the problem by replacing (14d) with $\rho_{i,k}^{m} \in [0,1]$. Since the resulting quadratic program (QP) is not convex we are forced to solve to first-order optimality only. We then rounded the solutions we obtained from the relaxed problem.
2. Drop Quadratic Terms: The quadratic terms are exclusively dependent on the new constraints. Hoping that the linear terms would contain sufficient information to achieve a good setting of $\boldsymbol{\rho}$ we remove all quadratic terms obtaining an Integer Linear Program (ILP). It turns out we were able to solve the resulting ILPs (without recurring to relaxation) in reasonable time.

The latter combination strategy is referred to as DOP1-ILP in the main paper. This is the strategy we found experimentally to produce the greatest reduction in the number of iterations. However, as reported in our experiments, the much simpler and computationally cheap DivMBest-Ordering strategy achieved similar savings in the number of iterations.

References

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

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