Supplementary Material for Paper "High Order Regularization for Semi-Supervised Learning of Structured Output Problems"

Yujia Li[1] YUJIALI@CS.TORONTO.EDU
Richard Zemel[1,2] ZEMEL@CS.TORONTO.EDU
[1]Department of Computer Science, University of Toronto, Toronto, ON, Canada
[2]Canadian Institute for Advanced Research, Toronto, ON, Canada

In this supplementary material we present:

- Dual decomposition inference for optimizing Equation 12 in the main paper.
- A leave-one-out algorithm for optimizing Equation 9 in the main paper for PASCAL loss.
- Remark on the connection between our method and Constraint Driven Learning (CODL).
- More details about the bird datasets.
- More details about hyper parameter settings.
- More experiment results.

1. Optimizing Equation 12 using Dual Decomposition

Eq.12 is a special case of the following more general optimization problem

$$\min_{\mathbf{y}} f^u(\mathbf{y}) + f^p(\mathbf{y}) + h(1^\top \mathbf{y})$$

where $\mathbf{y} \in \{0,1\}^P$, f^u, f^p and h are unary, pairwise and cardinality potentials respectively. To see this, note that in Eq.12 f is a sum of unary and pairwise potentials and $\Delta(\mathbf{y}_i, \mathbf{y}_j)$ is a sum of pairwise terms. This is hard to optimize due to the interaction between the pairwise potential and high order cardinality potential.

In dual decomposition, we decompose the original problem into two subproblems that are more tractable. We define $A(\mathbf{y}) = \mu f^u(\mathbf{y}) + f^p(\mathbf{y})$ and $B(\mathbf{y}) = (1-\mu) f^u(\mathbf{y}) + h(\mathbf{y})$, where μ is a fixed constant, e.g. 0.5, then the original objective is $A(\mathbf{y}) + B(\mathbf{y})$.

For any $\boldsymbol{\lambda} \in \mathbb{R}^P$, we have a lower bound on the original objective,

$$\mathcal{L}(\lambda) = \min_{\mathbf{y}} \left\{ A(\mathbf{y}) + \boldsymbol{\lambda}^\top \mathbf{y} \right\} + \min_{\mathbf{y}} \left\{ B(\mathbf{y}) - \boldsymbol{\lambda}^\top \mathbf{y} \right\}$$

As $\boldsymbol{\lambda}^\top \mathbf{y}$ is just a sum of very simple unary potentials, each of the subproblems here are easy to solve. For the first one, graph cuts can be used to find exact optimum, and for the second one we can use methods described in (Gupta et al., 2007).

We then maximize this lower bound over $\boldsymbol{\lambda}$, to make it as tight as possible and hence approach the optimum of the original problem. We can compute the subgradient of the lower bound with respect to $\boldsymbol{\lambda}$,

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} = \hat{\mathbf{y}}^A - \hat{\mathbf{y}}^B$$

where $\hat{\mathbf{y}}^A$ is the optimal $\mathbf{y}$ for the first subproblem and $\hat{\mathbf{y}}^B$ is the optimal $\mathbf{y}$ for the second subproblem. In our experiments we follow this subgradient to optimize the lower-bound, but a wide range of other optimization techniques can be applied here as well.

Once the optimization terminates, we have to decode the final $\mathbf{y}^*$ as the solution to the two subproblems may not agree. For this we can calculate the original objective for all $\hat{\mathbf{y}}^A$ and $\hat{\mathbf{y}}^B$'s encountered during the optimization and choose one that has the smallest objective value. Other heuristics can be applied here as well.

2. Optimizing Equation 9 for PASCAL Loss

For a single class, let $\mathbf{y} \in \{0,1\}^P$ denote the prediction for each pixel whether it belongs to that class, and let $\mathbf{y}^*$ be the ground truth. The PASCAL loss is defined as

$$\Delta(\mathbf{y}, \mathbf{y}^*) = 1 - \frac{\sum_i \mathbf{I}[y_i = 1 \text{ and } y_i^* = 1]}{\sum_i \mathbf{I}[y_i = 1 \text{ or } y_i^* = 1]}.$$

(Tarlow & Zemel, 2012) describes an efficient method to compute MAP for high order factors such as the PASCAL loss, with the true label fixed. Here we use this as a subroutine in our optimization.

Since the graph term in Eq. 9 is a sum of many high order factors, direct optimization is very hard, and even messages

are hard to compute. We therefore use a leave-one-out algorithm instead. This algorithm iterates through all j's one by one. For each j, all $\mathbf{y}_{j'}$ for $j' \neq j$ are fixed and we optimize over $\mathbf{y}_j$ only. For a single j, the corresponding optimization problem has the form of

$$\min_{\mathbf{y}_j} \quad \lambda \sum_{i:s_{ij}>0} s_{ij}\Delta(\mathbf{y}_i, \mathbf{y}_j) - \mu f(\mathbf{x}_j, \mathbf{y}_j, \mathbf{w})$$

This is a sum of unary and pairwise potentials in f plus a set of PASCAL loss high order potentials.

We can again use dual decomposition to do the optimization. There are two types of subproblems: (1) unary potentials + pairwise potentials, which can be optimized using graph cuts; (2) unary potentials + one PASCAL loss potential, which can be optimized by invoking the optimization subroutine.

This optimization for a single j can also be done by message passing, as messages for the PASCAL loss can be efficiently computed as described in (Tarlow & Zemel, 2012).

If for each j this optimization can decrease the objective, then Step 1 of our proposed algorithm in Section 3.2 will monotonically decrease the objective in Eq.5, therefore our algorithm is still guaranteed to converge.

3. Remark on the Connection between Our Method and Constraint Driven Learning (CODL)

The Constraint Driven Learning (CODL) (Chang et al., 2007) is similar to our algorithm described in Section 3.2 in the main paper, which is also an alternating optimization method, with some notable and important differences:

1. Few people have explored the use of high order regularizers in CODL, and the optimization in Step 1 is usually done by heuristic search rather than using efficient discrete optimization algorithms.

2. In Step 2, CODL uses the inferred labels as true labels while we use them in the constraint relaxation penalty.

3. The CODL learning algorithm does not correspond to the optimization of a unified objective function. As we derive the algorithm from a joint optimization problem, it is possible to develop variants other than the coordinate descent currently used.

4. More Details about the Bird Datasets

We obtained the PASCAL VOC "bird" dataset by first restricting the image based on the bounding box containing the bird, and then labeling all bird pixels as 1 and all other pixels as 0, resulting in 214 bird images with segmentations.

We selected images from the pool of unlabeled images by utilizing the Histogram of Oriented Gradients (HOG)(Dalal & Triggs, 2005) image features as the distance measure and choosing the set closest to the labeled images. We choose 500 images from CIFAR-10 for the horse segmentation task, and 600 images from CUB for the bird segmentation task according to this criterion.

We obtained labels for CUB dataset, from the rough segmentations provided in CUB. The rouph segmentations provides a localization of the object but are not very precise around the boundary. Usually the rouph segmentations will include a significant amount of background in the foreground mask. To refine this, we fixed the pixel labels in the interior of the foreground area and the background area, and try to relabel the boundary pixels. More specifically, we used GrabCut (Rother et al., 2004), by alternating appearance model fitting and segmentation label updates. The fixed foreground and background areas are used to train the initial appearance model. After that, an extra hole filling operation from mathematical morphology is used to post process the results. These segmentations generated in this way capture most of the details of bird silhouettes. Fig. 1 shows some sample images and generated segmentation masks for the CUB dataset.

5. More Details about Hyper Parameter Settings

We formulate μ as μ'/U where U is the number of unlabeled examples. We found that μ' is not very sensitive to different datasets, and we used the same $\mu' = 100$ for all splits of all datasets. More tuning of this parameter for different datasets may result in even better performance.

The parameter γ for cardinality potentials is fixed to 1 for all experiments.

Parameter λ and learning rate, momentum, etc. for the neural networks are tuned using the validation set. We found that λ is more sensitive to datasets than μ' and γ, but a wide range of λ works quite well.

6. More Experiment Results

Some segmentation results for the intial model, and the models trained with self-training, our graph based method and our graph based method + cardinality regularizer are shown in Fig. 2. These examples are randomly chosen with models trained with 40 labeled images on one split of the corresponding datasets.

The effect of using cardinality regularizers is most obvious

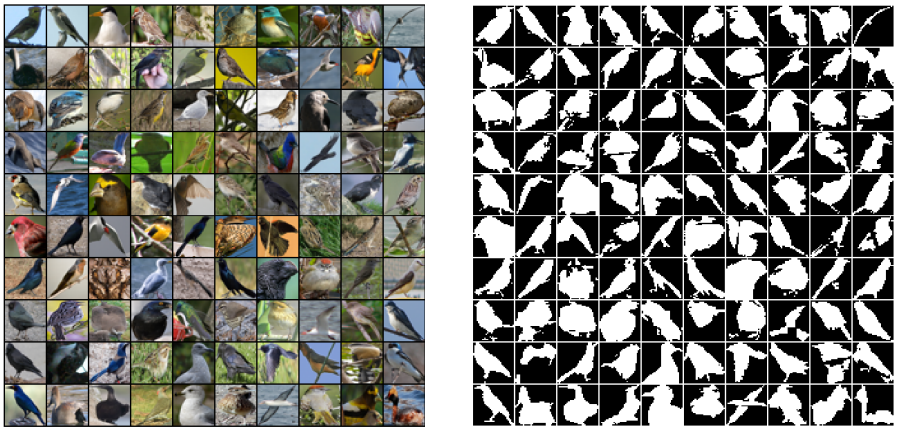

Figure 1. Samples from the CUB dataset and generated segmentations.

from the horse segmentation results. On the bird dataset, there is a significant difference between methods that use graph sturcture ("Graph" and "Graph-Card") and those do not ("Initial" and "Self-Train").

References

Chang, Ming-Wei, Ratinov, Lev, and Roth, Dan. Guiding semi-supervision with constraint-driven learning. In *ACL*, 2007.

Dalal, Navneet and Triggs, Bill. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

Gupta, Rahul, Diwan, Ajit A, and Sarawagi, Sunita. Efficient inference with cardinality-based clique potentials. In *Proceedings of the 24th International Conference on Machine Learning*, 2007.

Rother, Carsten, Kolmogorov, Vladimir, and Blake, Andrew. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 2004.

Tarlow, Daniel and Zemel, Richard S. Structured output learning with high order loss functions. In *Proceedings of the 15th Conference on Artificial Intelligence and Statistics*, 2012.

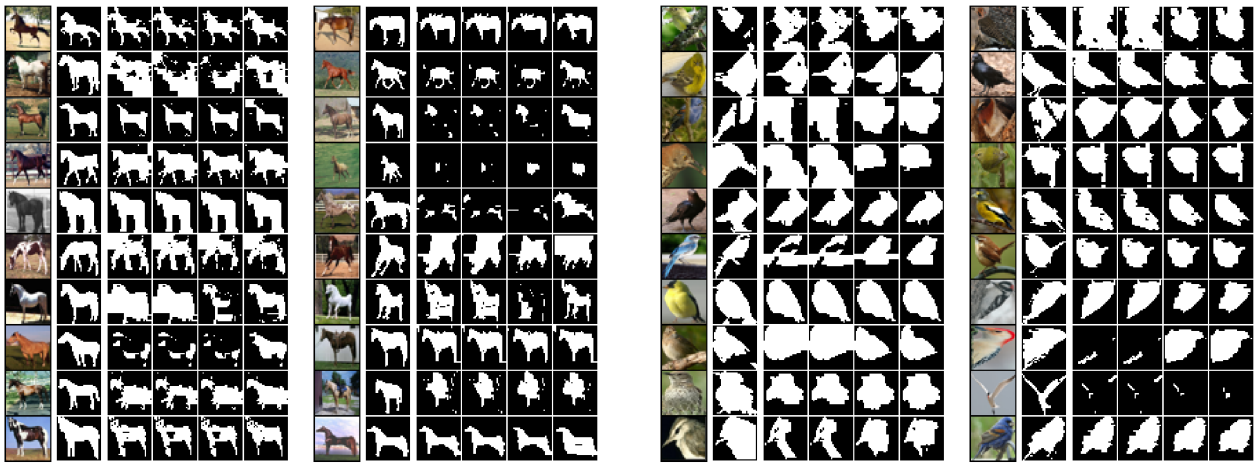

Figure 2. Example segmentation results for horse and bird images. In each row, the first two columns contain the original image and the ground truth segmentation. The next four columns are results obtained using “Initial”, “Self-Training”, “Graph”, “Graph-Card”.