Complex Loss Optimization via Dual Decomposition

Mani Ranjbar, Arash Vahdat and Greg Mori
School of Computing Science, Simon Fraser University
Burnaby, BC, Canada
{mra33, avahdat, mori}@cs.sfu.ca

Abstract

We describe a novel max-margin parameter learning approach for structured prediction problems under certain non-decomposable performance measures. Structured prediction is a common approach in many vision problems. Non-decomposable performance measures are also commonplace. However, efficient general methods for learning parameters against non-decomposable performance measures do not exist. In this paper we develop such a method, based on dual decomposition, that is applicable to a large class of non-decomposable performance measures. We exploit dual decomposition to factorize the original hard problem into two smaller problems and show how to optimize each factor efficiently. We show experimentally that the proposed approach significantly outperforms alternatives, which either sacrifice the model structure or approximate the performance measure, and is an order of magnitude faster than a previous approach with comparable results.

1. Introduction

Max-margin structured prediction is a common framework for a variety of vision problems. Tasks such as action recognition [27, 9], image segmentation [19], and scene understanding [4] involve reasoning about relationships between locations in a video, pixels in an image, and objects in a scene respectively. In addition, the performance measures for these applications often are non-decomposable and are not a simple sum of terms measured over individual output entities. Instead, they measure performance as a function of the entire, structured output. The focus of this paper is developing a learning approach that can handle these together, formulating a learning algorithm based on dual decomposition for handling structured prediction while optimizing against certain non-decomposable performance measures.

Capturing the structure of the output requires a model that is rich enough to absorb the dependencies between the outputs. In this paper we employ Markov networks, which are commonly used for modeling interdependent inputs and outputs in many applications.

We focus on a subset of non-decomposable loss functions, namely, multivariate non-linear performance measures[1], such as F_1 (image retrieval), area under ROC (binary classification), or intersection over union (image segmentation). However, our approach can be applied more generally, for optimizing against any loss function for which the maximum value of the loss plus a linear function can be computed efficiently. One example of such a loss is mean average precision [29].

Modeling dependencies between outputs while optimizing against a loss function has been a research topic for many years. Optimizing the expected loss in this scenario is a non-convex problem. However, Taskar et al. [22] and Tsochantaridis et al. [25] have proposed rather to optimize a convex relaxation of the expected loss. The cutting-plane algorithm has been shown to be efficient for solving this optimization [25]. The computational difficulty in structured prediction approaches is finding the subgradient, which requires solving the "most violated constraint" [25] or "loss augmented inference" [21]. It is shown that for decomposable performance measures learning is tractable when the model is a submodular Markov network or a matching [22, 25, 23]. In contrast, in this paper we focus on non-decomposable performance measures.

Joachims [10] proposed an approach to efficiently compute the most violated constraint for a large class of non-decomposable loss functions, a subset of those we consider in this paper. However, the underlying models were limited, and do not permit pairwise interactions between output labels. The method of Yue et al. [29] takes a similar approach to optimize against Mean Average Precision. Khanna et al. [3] present an algorithm in the same framework to optimize against normalized discounted cumulative gain (NDCG). Rather than solving a convex relaxation of the expected loss, McAllester et al. [13] proposed a perceptron-like training approach to directly optimize the original loss function, but still need to solve the loss augmented inference.

Four closely related pieces of work in the literature are

[1] losses that are a function of false positive and false negative counts

the approaches of Meshi et al. [14], Komodakis [11], Tarlow and Zemel [20] and Ranjbar et al. [16]. The work by Meshi el al. [14] shares a similar formulation, and proposes to solve the loss augmented inference and parameter learning simultaneously. Our approach is similar in the sense that we work on the dual of the loss augmented inference. However, our goal is to optimize against non-decomposable losses, while [14] is only concerned with decomposable loss functions. Komodakis [11] addresses learning the parameters of a Markov network with higher order potential functions. The case of decomposable Hamming loss is developed in detail, and the applicability for general losses is alluded to. Our paper precisely shows how the parameters of a Markov network can be learned given non-decomposable loss functions that operate over *all* output variables. Tarlow and Zemel [20] also attempt to learn against complex losses, but with a very different approach. In that paper a message passing strategy is proposed and the results are shown on a subset of the Pascal VOC images. Note that the ratio of positive to negative pixels is an order of magnitude larger in the selected subset, which makes the results incomparable. Ranjbar et al. [16] take a different approach to optimize against non-decomposable loss functions. There, the loss function is approximated with a piecewise planar surface and the loss augmented inference is solved independently for each piece by solving a linear program. The drawbacks of this approach are two-fold. First, its complexity increases linearly with the number of planes in the approximation. Moreover, this approach is computationally very expensive in practice since it requires solving a linear program for each plane in each iteration of the cutting plane approach. Second, this method optimizes against the approximated version of the loss function as opposed to our approach that optimizes against the actual loss.

2. Structured Prediction Learning

The goal of our learning problem is defined as finding a function $h \in \mathcal{H}$ from the hypothesis space $\mathcal{H}$ given training samples S = $((\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N))$ that optimizes the expected prediction performance on the new samples S' of size N'.

$$R^{\Delta}(h) = \int \Delta\Big((h(\boldsymbol{x}'_1), h(\boldsymbol{x}'_2), \ldots, h(\boldsymbol{x}'_{N'})), (y'_1, y'_2, \ldots, y'_{N'})\Big) dPr(S'). \quad (1)$$

In general, the loss function Δ cannot be decomposed into a linear combination of a loss function δ over individual samples. But, for simplicity, most discriminative learning algorithms (e.g. SVM) assume decomposibility and i.i.d. samples, which allows for rewriting Eq. 1 as

$$R^{\Delta}(h) = R^{\delta}(h) = \int \delta(h(\boldsymbol{x}'), y') dPr(\boldsymbol{x}', y'). \quad (2)$$

Instead of solving the estimated risk in Eq. 2, learning algorithms approximate that with empirical risk $\hat{R}^{\delta}$ defined as

$$\hat{R}^{\delta}(h) = \frac{1}{N} \sum_{i=1}^{N} \delta(h(\boldsymbol{x}_i), y_i). \quad (3)$$

For non-decomposable loss functions, such as F_1 score or intersection over union, optimizing Eq. 2 does not provide the desired answer. Rather, we are interested in finding an algorithm that can directly optimize the empirical risk based on the sample loss,

$$\hat{R}^{\Delta}_S(h) = \Delta\big((h(\boldsymbol{x}_1), h(\boldsymbol{x}_2), \ldots, h(\boldsymbol{x}_N)), (y_1, y_2, \ldots, y_N)\big). \quad (4)$$

Finding an $h \in \mathcal{H}$ that optimizes Eq. 4 for an arbitrary loss function Δ can be computationally challenging. In structured prediction, instead of having a mapping function $h : \mathcal{X} \rightarrow \mathcal{Y}$ from a single example $\boldsymbol{x}$ to its label y, where $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$, all examples are considered at once to learn a mapping function $\bar{h} : \mathcal{X} \times \cdots \times \mathcal{X} \rightarrow \bar{\mathcal{Y}}$, where $\bar{\mathcal{Y}} \in \{\mathcal{Y}\}^N$. We define $\bar{\boldsymbol{x}} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$, and $\boldsymbol{y} = (y_1, \ldots, y_N)$.

We can define the best labeling using a linear discriminant function

$$\hat{h}(\bar{\boldsymbol{x}}) = \arg\max_{\boldsymbol{y}' \in \bar{\mathcal{Y}}} \boldsymbol{w}^T \Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}'). \quad (5)$$

Here, the function Ψ measures the compatibility of the data points and their assigned labels.

One way of incorporating a loss function Δ in the SVM formulation is *Margin Rescaling* [24], which can be written as an unconstrained optimization problem [5],

$$\min_{\boldsymbol{w}} \frac{\rho}{2}\|\boldsymbol{w}\|^2 + \max_{\boldsymbol{y}' \in \bar{\mathcal{Y}}} \big(\boldsymbol{w}^T \Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}') + \Delta(\boldsymbol{y}', \boldsymbol{y})\big) - \boldsymbol{w}^T \Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}) \quad (6)$$

Solving the inner *max*, which is called loss augmented inference or the most violated constraint is computationally challenging given an arbitrary loss function $\Delta(\boldsymbol{y}, \boldsymbol{y}')$ and scoring function $\Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}')$. However, solving loss augmented inference has been shown to be efficient in some special cases. The first special case is when the inference itself is tractable and the loss function is decomposable [24]. The second case happens when the Ψ function is decomposable over examples and the loss function is a function of false positive and false negative counts [10]. For this type of loss function Ranjbar et al. [16] approximate loss with a piecewise linear function when the model is a Markov network over binary variables. In this approach, a linear program is solved for each loss piece, which is computationally expensive when the number of regions is large. In this paper we propose to solve the loss augmented inference for Markov networks using dual decomposition. The idea of dual decomposition is to divide the optimization problem into simpler subproblems that are solved independently and combined into a global solution.

3. Loss Optimization via Dual Decomposition

Our goal in this paper is to solve the loss augmented inference problem for Markov networks and a subset of non-decomposable loss functions, those that can be maximized when augmented with a linear term. Multivariate non-linear performance measures (e.g. F_1 measure, intersection over union, and area under ROC) and mean average precision are some examples of such losses.

We assume that we are given a Markov network represented by a graph $G = (V, E)$ where V is the set of nodes, and E is the set of edges. The score of an assignment $\boldsymbol{y}'$ is defined as a linear function of unary and pairwise scoring functions. Although several methods have been proposed to find the best assignment by maximizing the scoring function of a Markov network, $\boldsymbol{w}^T\Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}')$, the optimization becomes challenging when the score is augmented with a non-decomposable loss function. In this section we propose to use the dual decomposition technique to divide the loss augmented inference into simpler components that can be solved independently.

We can modify the loss augmented inference problem (maximization in Eq. 6) by duplicating the y'_i variables, and then forcing the duplicates to be equal. So, the equivalent optimization is:

$$\begin{aligned} \max_{\boldsymbol{y}', \boldsymbol{y}''} \quad & \boldsymbol{w}^T\Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}') + \Delta(\boldsymbol{y}'', \boldsymbol{y}) \\ \text{s.t.} \quad & y'_i = y''_i, \forall i \end{aligned} \tag{7}$$

Without the constraints, the maximization in Eq. 7 could be decomposed into independent maximization for each term which are easier to solve. In order to remove the constraints, we use the Lagrangian Relaxation technique. We define the Lagrangian as:

$$\begin{aligned} L(\bar{\boldsymbol{\lambda}}, \boldsymbol{y}', \boldsymbol{y}'') =& \boldsymbol{w}^T\Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}') + \Delta(\boldsymbol{y}'', \boldsymbol{y}) + \\ & \sum_{i=1}^{N} \sum_{\hat{y}_i \in \mathcal{Y}} \lambda_i(\hat{y}_i) \left(\mathbb{1}_{[y'_i = \hat{y}_i]} - \mathbb{1}_{[y''_i = \hat{y}_i]} \right) \end{aligned} \tag{8}$$

where $\bar{\boldsymbol{\lambda}} = [\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \ldots, \boldsymbol{\lambda}_N]$ and $\boldsymbol{\lambda}_i$ is a vector including all Lagrange multipliers for the i^{th} node of Markov network for all $y_i \in \mathcal{Y}$. Then the following optimization is equivalent to the loss augmented inference optimization in Eq. 7:

$$\begin{aligned} \max_{\boldsymbol{y}', \boldsymbol{y}''} \quad & L(\bar{\boldsymbol{\lambda}}, \boldsymbol{y}', \boldsymbol{y}'') \\ \text{s.t.} \quad & y'_i = y''_i, \forall i \end{aligned} \tag{9}$$

The optimization in Eq. 9 is as hard as the original optimization. But we can omit the constraints and define $L(\bar{\boldsymbol{\lambda}})$ as:

$$L(\bar{\boldsymbol{\lambda}}) = \max_{\boldsymbol{y}', \boldsymbol{y}''} L(\bar{\boldsymbol{\lambda}}, \boldsymbol{y}', \boldsymbol{y}'') = \tag{10}$$

$$\max_{\boldsymbol{y}'} \left(\boldsymbol{w}^T\Psi(\bar{\boldsymbol{x}}, \boldsymbol{y}') + \sum_{i=1}^{N} \lambda_i(y'_i) \right) + \max_{\boldsymbol{y}''} \left(\Delta(\boldsymbol{y}'', \boldsymbol{y}) - \sum_{i=1}^{N} \lambda_i(y''_i) \right)$$

The optimization subproblems in Eq. 10 are independent and can be solved separately. It can be easily shown that $L(\bar{\boldsymbol{\lambda}})$ is an upper bound for the original optimization problem (Sontag et al. [18]). So, we can find an approximate solution to the loss augmented inference by finding the tightest upper bound. We solve the dual problem, $min_{\bar{\boldsymbol{\lambda}}} L(\bar{\boldsymbol{\lambda}})$ to find such an upper bound, and use its solution to recover an assignment for the primal optimization.

Note that $L(\bar{\boldsymbol{\lambda}})$ is non-differentiable at all points $\bar{\boldsymbol{\lambda}}$ that either of the two terms in Eq. 10 has multiple local optima. So, similar to [18] the subgradient method is used to solve the dual problem. In each step of this method, the maximization problems in Eq. 10 are solved for the current $\bar{\boldsymbol{\lambda}}^t$, then the $\bar{\boldsymbol{\lambda}}^{t+1}$ is updated toward the direction of subgradient using $\bar{\boldsymbol{\lambda}}^{t+1} = \bar{\boldsymbol{\lambda}}^t - \alpha^t \bar{\boldsymbol{g}}^t$, where $\bar{\boldsymbol{g}}^t$ is the subgradient of $L(\bar{\boldsymbol{\lambda}})$ at $\bar{\boldsymbol{\lambda}}^t$ and α^t is a step-size that is set to $\alpha^t = \frac{1}{\sqrt{t}}$ in our experiments. In the remainder of this section we discuss how we solve the subproblems in Eq. 10 and calculate the subgradient.

We need to solve two maximization problems in each iteration of the subgradient method. The first maximization problem in Eq.10 is maximizing a Markov network with additional unary terms from the Lagrange variables. So, for solving the first subproblem in Eq. 10 we modify the Markov network such that the pairwise scores are the same, but the unary scores are added with $\boldsymbol{\lambda}_i$ at each node. Hence, the first maximization can be solved (exactly or approximately) using any MAP inference technique (e.g. see [28]).

The second optimization in Eq. 10 involves maximizing the loss function plus a linear term, which is challenging, because an exhaustive search over all $\boldsymbol{y}'' \in \bar{\mathcal{Y}}$ is not tractable. However, several methods have been proposed to solve this maximization. For example, Joachmis method [10] solves the maximization for multivariate non-linear performance measures and Yue et al. method [29] solves the maximization for mean average precision. Here, we show how to solve this maximization for multivariate non-linear performance measures using the general form of Joachims approach. However, any other loss function, for which this maximization is tractable could be considered.

Note that for multivariate performance measures there are only $O(N^2)$ different contingency cases when the network has N nodes. This is due to the fact that the number of false positives can change from zero to the size of the negative set, and number of false negatives can change from zero to the size of the positive set. So, overall, loss function can take at most $O(N^2)$ different values. Joachims's approach [10] takes advantage of this fact and proposes an algorithm to solve the loss augmented inference efficiently in $O(N^2)$ time. However, the limiting assumption in this work is that the model only involves unary terms. So, instead of using this algorithm to solve the entire loss augmented inference, which involves unary and pairwise potentials, we employ this approach to solve the second term in Eq. 10.

Note that the second maximization in Eq. 10 is a summation of the loss and unary terms that score the assignments individually. We further generalize the Joachims method for multi-label problems in Algorithm 1.

In Algorithm 1, we assume that the loss function is defined on the number of false positives and false negatives even for multi-class labeling. In other words, the loss is defined for a particular class versus other classes. That is the reason our algorithm accepts the positive label p as an input. For more complex loss functions defined as a function of false positives for each class, we still could generalize the Algorithm 1 by repeating the for loops for each class and finding the labeling that maximizes the summation of loss and Lagrange multipliers. However, in this case the complexity of this algorithm would be $O(N^{|\mathcal{L}|})$. It can be easily shown that the Algorithm 1 is the generalized form of Joachims's algorithm [10] for multi-label problems. Note that for binary label problems these two algorithms are exactly the same.

Algorithm 1 Finding the argmax value of $\bar{\boldsymbol{\lambda}}(\boldsymbol{y}')+\Delta(\boldsymbol{y},\boldsymbol{y}')$ over $\boldsymbol{y}'$

Input : $\bar{\boldsymbol{\lambda}} = \{\boldsymbol{\lambda}_1,\dots,\boldsymbol{\lambda}_N\}$, $\boldsymbol{y} = \{y_1,\dots,y_N\}$, $\mathcal{Y}$ and positive label p
$\#pos \leftarrow \sum_j \mathbb{1}_{[y_j=p]}$
$\#neg \leftarrow \sum_j \mathbb{1}_{[y_j\neq p]}$
$(i^p_1,\dots,i^p_{\#pos}) \leftarrow$ Sort $\downarrow$ $\{i : y_i = p\}$ by $\max_{j\neq i}(\lambda_i(j)) - \lambda_i(p)$
$(i^n_1,\dots,i^n_{\#neg}) \leftarrow$ Sort $\downarrow$ $\{i : y_i \neq p\}$ by $\lambda_i(p) - \max_{j\neq i}(\lambda_i(j))$
$(r_1,\dots,r_{\#neg}) \leftarrow \{\arg\max_r \lambda_i(r) : i \in \{j : y_j \neq p\}, r \in \mathcal{Y}\backslash\{p\}\}$
for $fn \leftarrow 0$ **to** $\#pos$ **do**
  set $\{y'_{i^p_1},\dots,y'_{i^p_{fn}}\}$ to $\{r_{i^p_1},\dots,r_{i^p_{fn}}\}$ **and** set $\{y'_{i^p_{fn+1}},\dots,y'_{i^p_{\#pos}}\}$ to p
  for $fp \leftarrow 0$ **to** $\#neg$ **do**
    set $\{y'_{i^n_1},\dots,y'_{i^n_{fp}}\}$ to p **and** set $\{y'_{i^n_{fp+1}},\dots,y'_{i^n_{\#neg}}\}$ to $\{r_{i^n_{fp+1}},\dots,r_{i^n_{\#neg}}\}$
    $v \leftarrow \sum_j \lambda_j(y'_j) + \Delta(fp,fn)$
    if v is the largest so far **then**
      $\boldsymbol{y}^{**} \leftarrow (y'_1,\dots,y'_N)$
    end if
  end for
end for

Having maximized the subproblems in Eq. 10, we calculate the subgradient of $L(\bar{\boldsymbol{\lambda}})$ to update the $\bar{\boldsymbol{\lambda}}^t$. Let $\boldsymbol{y}^*$ and $\boldsymbol{y}^{**}$ be the optimal assignments for the first and the second subproblems of Eq. 10 respectively. We calculate the subgradient of $L(\bar{\boldsymbol{\lambda}})$ at $\bar{\boldsymbol{\lambda}}^t$ using Algorithm 2. In this algorithm, if a node is assigned to the same label in both subproblems, the subgradient will be zero for all labels; Otherwise, the subgradient is set to the $+1$ and -1 for the labels assigned by the first and the second optimization in Eq. 10 respectively. In each iteration, if a node is labeled differently by the subproblems, the subgradient method will decrease the value of $\lambda_i(y^*_i)$, and increase the value of $\lambda_i(y^{**}_i)$. So, intuitively, in the next iteration, both y^*_i and y^{**}_i will have less chance to be assigned to their previous labels and the subproblems are more likely to produce agreeing assignments.

To find the tightest upper bound, we iterate the subgradient descent until $L(\bar{\boldsymbol{\lambda}})$ stops changing significantly or a desired maximum number of iterations (100 in our experiments) is reached. If at the final iteration subgradient becomes zero, we can conclude that both assignments, $\boldsymbol{y}^*$ and $\boldsymbol{y}^{**}$ are in agreement for all nodes, and the assignment is the global solution to the loss augmented inference problem. However, there is no guarantee to reach such an assignment, and in practice, we often have a few nodes that have different assignments in the final $\boldsymbol{y}^*$ and $\boldsymbol{y}^{**}$[2]. In this case we need to recover a primal solution from the assignments proposed by each subproblem. A simple heuristic to recover the solution to the primal equation, which works very well in practice, is to count the number of times that a node is assigned to a label in the iterations of the subgradient method, and assign to the nodes with disagreeing assignments the label that has occurred the most for each node [12].

Algorithm 2 Calculating $\bar{\boldsymbol{g}}^t$, the subgradient of $L(\bar{\boldsymbol{\lambda}})$ at $\bar{\boldsymbol{\lambda}}^t$

Input : $\boldsymbol{y}^*$ and $\boldsymbol{y}^{**}$
$g^t_i(y_i) = 0, \forall i, \forall y \in \mathcal{Y}$
for $i \leftarrow 1$ **to** N **do**
  if $y^*_i \neq y^{**}_i$ **then**
    $g^t_i(y^*_i) = +1$
    $g^t_i(y^{**}_i) = -1$
  end if
end for

4. Experimental Results

We employ object category segmentation as an example of a structured output problem. The task is to label the pixels of an image as being part of a known object (foreground) or not (background). We set the label of foreground to one and the label of background to zero. One example of a non-decomposable performance measure in image segmentation task is intersection over union defined as

$$Acc_{\frac{\cap}{\cup}}(FP,FN) = \frac{N_p - FN}{N_p + FP} \iff \Delta_{\frac{\cap}{\cup}}(FP,FN) = \frac{FP+FN}{N_p+FP}, \tag{11}$$

which has been used to compare the segmentation accuracies on the Pascal VOC challenge [6]. In our experiments we optimize against this loss function while comparing to two strong baselines and a previous approach [16].

[2] In our experiments, in more than 95% of the cases less than 1% of the nodes had different assignments

The first baseline called *Hamming* approximates the intersection over union loss with normalized Hamming loss – a decomposable loss defined as

$$\Delta_{Hamming} = \alpha FP + \beta FN \tag{12}$$

where the value of α and β are set to $1/(2N_n)$ and $1/(2N_p)$, respectively. Here, N_p and N_n represent the number of positive and negative samples in the dataset. Note that Hamming loss decomposes as an extra unary potential in the overall Markov network. Solving for the loss augmented inference in this case is computed by performing MAP inference on the network. For some special cases, e.g. when the graph is submodular, MAP inference has been shown to be exact. However, in a general graph MAP inference has been shown to be NP hard and therefore approximate inference techniques could be employed to find an approximate solution [2, 28]. Note that solving the loss augmented inference for Hamming loss is similar to solve the first maximization in Eq. 10. This baseline shares the same model as we have in the proposed approach, but optimizes against a decomposable loss as opposed to the original non-decomposable loss.

The second baseline is based on the idea of Joachims [10], which can optimize against a large set of loss functions, those which can be computed from the contingency table, a subset of loss functions that can be optimized using the proposed approach. However, this approach assumes a decomposable model and cannot benefit from the pairwise interactions on the outputs. It can only capture the dependency between the outputs through the loss function and not through the model. Note that the second maximization in Eq. 10 has exactly the same form as explained here – a non-decomposable loss computed from the contingency table plus a linear term.

The third method that we compare the results with is [16], which takes a very different approach. Briefly, the loss function that is assumed to be a function of false positive and false negative counts is approximated with a piecewise linear surface in false positive and false negative space. Then a linear program is solved for each piece independently and the piece with maximum loss augmented value is picked as the answer. The drawback of this method is the need to solve many linear programs in each iteration of the parameter learning. We use 15 pieces to approximate the loss function, which generates almost identical results to the original choice of 40 pieces in [16] for computational efficiency. We employ Mosek [15] the same off-the-shelf LP solver to solve the LPs. Note that solving each LP takes more than finding the loss augmented inference in the proposed method with 100 subgradient steps, which makes the proposed approach more than 15 times faster than the approach in [16], yet with comparable results.

4.1. Pixels vs. Superpixels

If we decide to perform segmentation on the pixel level, meaning that the input be the set of all features extracted from all pixels in the dataset and the output be the binary label of each pixel, then for a dataset with 750 images each of size 500×300 pixels, we will have $112,500,000$ variables (nodes in the Markov network). Solving the second term of Eq. 10 (Algorithm 1) as well as computing the loss augmented inference in Joachims' method [10] have time complexity $O(N^2)$, where N is the number of nodes. Clearly, such computation is intractable.

To solve this problem, we group neighbouring pixels into superpixels (of possibly different sizes[3]) based on their color distance and assign one label to all pixels of a superpixel. However, the assumption in Algorithm 1 is that all positive examples (superpixels of the foreground) contribute equally in the loss function, which is not true if the area of the superpixels are different. To solve this problem, one could look at the equivalent pixel level problem. Assuming a similar feature vector for all pixels of a superpixel, each pixel receives $1/m$ of the unary score of its superpixel, where m is the number of pixels in the superpixel. Therefore, sorting the scores (lines 4 and 5 of Algorithm 1) would rank all pixels of a superpixel sequentially. So the equivalent pixel level computation would never create inconsistent labels for pixels of a superpixel except for possibly one foreground superpixel and one background superpixel[4]. So instead of iterating over the pixels, we sort the unary score of the superpixels divided by their area and correct the false positive and false negative counts based on the area of the chosen superpixels. With superpixels of size at least 2000 pixels, which still can potentially provide very accurate segmentations, we achieved a speedup of more than $4,000,000$ times. In our experiments we employ the superpixel extractor of Felzenszwalb et al. [8] and set its parameters to $MinArea = 2000, k = 200, \sigma = 0.01$.

4.2. Learning Method

We utilize NRBM [5] – an instance of a bundle method – as the core of our learning and solve the loss augmented inference based on the proposed approach. Note that other structured prediction formulations such as Pegasos [17] or the formulation proposed by Meshi et al. [14] could easily replace the bundle method. We chose NRBM due to implementation simplicity knowing that it has the same bound of

[3]The other alternative is to force the superpixels to have the same size, but then large flat regions such as sky would be broken into many small superpixels and regions of small objects could be grouped with background.

[4]Let's first consider the simple case of binary labels($|\mathcal{Y}| = 2$). When the optimal labeling y^{**} is achieved, the first fn pixels of foreground pixels in the sorted order get value 0 and the rest get value 1. Knowing that the pixels of a superpixel are sorted sequentially, the only superpixel that may have inconsistent labels is the one that its pixel is located at position fn. The same argument holds for background superpixels. One could easily verify that the same property holds for the multi-label case as well.

$O(1/\epsilon)$ like aforementioned alternatives to obtain a solution of accuracy ϵ. We do cross validation to set the ρ parameter in Eq. 6. Although we conduct the experiments on binary label problems, the first term in Eq. 10 could still be NP hard to solve given an arbitrary vector w. To make the graph supermodular, we add constraints to force the weights corresponding to pairwise features (which are always positive) to be smaller than zero.

4.3. Features

In our experiments we use a set of features that represent each superpixel and the pairwise interaction between neighbouring superpixels. Our first feature is a bag-of-words representation over color SIFT [26] with 1000 visual words. Our second set of features are the part filter responses obtained from the object detector of Felzenszwalb et al. [7]. We aggregate the filter responses in different scales and sum the responses inside each superpixel. The pairwise features between neighbouring features have three components each representing the average absolute distance between pixels' colors of the two superpixels on one color channel. The cost of assigning the same labels to neighbouring superpixels is set to zero.

4.4. Pascal Visual Object Classes (VOC) 2009 and 2010 Segmentation Datasets

The Pascal VOC 2009 and 2010 datasets include 749 and 964 pixel-level labeled training images, respectively. They also include 750 and 964 validation images, respectively. We decide to train our method on the training set and test on the validation set, because the ground-truth for the test set is not publicly available and our focus is on comparison to baseline methods using a different model or learning criterion. We present the results on 6 object categories, Aeroplane, Bus, Car, Horse, Person, and TV/Monitor. We select these categories because the top-down unary features obtained from the Felzenszwalb et al. object detector provide reasonable detection on them. Without the top-down features, the overall accuracy would be so low as to make the comparison between different learning methods uninformative. One of the most challenging aspects of these datasets is the ratio of foreground to background pixels for all categories (Table 1). Moreover, the images in these datasets are not taken in a controlled environment and include severe illumination and occlusion. We compare the proposed approach to the baselines on the 6 object categories in Fig. 1. As illustrated, the proposed approach significantly outperforms the baselines in both datasets. We also compare the results with [16] in Fig. 1. The results of the proposed approach are slightly better in all classes except in class "Aeroplane" in VOC 2009 and "Car" in VOC 2010 dataset, but is comparable overall. However, the proposed approach is more than 15 times faster.

Experiments on class "Aeroplane" in VOC 2010 dataset show that our Matlab implementation takes 104.4 seconds to optimize the loss augmented inference for a given w on an Intel Core i7-860 2.80GHz with 8GB RAM. This running time is averaged over all inferences computed for NRBM algorithm given a ρ parameter. Consequently, training a model using 500 iterations of NRBM takes about 14.5 hours.

Table 1. Background to foreground pixel ratio in Pascal VOC 2009 and 2010 datasets

	Aeroplane	Bus	Car	Horse	Person	TV/Monitor
VOC 09	163	69	86	130	24	96
VOC 10	168	64	70	119	26	105

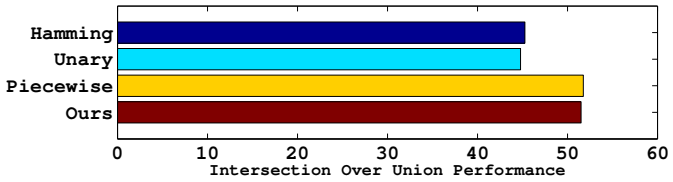

Figure 2. Intersection over union performance (%) comparison on H3D dataset

4.5. H3D Dataset

We also compare the results on the H3D dataset [1]. This dataset includes 273 training and 107 testing images along with three types of annotations – keypoint annotations, 3d pose annotation and region annotation. The keypoint annotation includes the location of joints and other keypoints such as eyes, nose, elbows, etc. The 3d pose annotation has been inferred from the keypoints. The region annotation, which we use in this paper, provides detailed annotation of people, such as face, neck, lower and upper cloth, etc. For our experiments we compute the union of all region annotations that are part of a person (bags, occluder and hat are not considered as parts of a person) as foreground and the rest as background. The ratio of background to foreground pixels in this dataset is 3.9, which is significantly lower than the ratio in Pascal VOC datasets. The reason is that all images in H3D dataset include at least some foreground pixels, which is not the case in Pascal VOC datasets. The comparison result in Fig. 2 shows that the proposed approach outperforms the baselines significantly on this dataset and is comparable to a previous approach.

4.6. Convergence Analysis

The proposed approach provides an efficient way of decomposing the loss augmented inference into two subproblems – one involving a supermodular function and the other one that can be exhaustively searched over to find the maximum value. At the same time, the proposed method inherits all the convergence properties of Lagrangian relaxation solved by the subgradient method. Precisely, the subgradi-

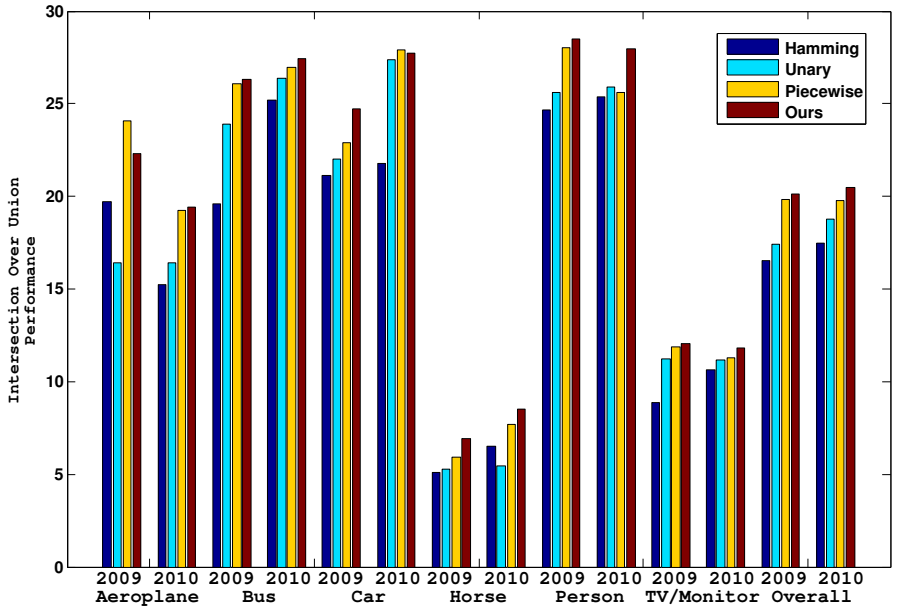

Figure 1. Intersection over union performance (%) comparison on VOC 2009 and VOC 2010 datasets

ent method is guaranteed to converge to the optimal of the dual function if the sequence of the multipliers α^t satisfy

$$\alpha^t \geq 0, \ \lim_{t\to\infty} \alpha^t = \infty, \ \sum_{t=1}^{\infty} \alpha^t = \infty \qquad (13)$$

which is the case for our choice of $\alpha^t = 1/\sqrt{t}$ [12]. Although the solution to the Lagrangian relaxation is as good as the solution of the LP relaxation, there is no guarantee that the LP relaxation is tight. Fig. 3 shows the improvement of the primal and the dual values in loss augmented inference for a given w. The gap between the primal and the dual functions decreases as subgradient descent algorithm proceeds. Fig. 4 shows the average percentage of disagreeing assignments versus the number of subgradient iterations. The average is taken over all inferences computed for the NRBM algorithm in training a model for the class 'Aeroplane" in VOC 2010 dataset. Number of disagreeing assignments decreases as the loss augmented model is being optimized and in average only 0.63% of assignments are disagreeing at the last iteration. Note, of course, that amounts of disagreement do not necessarily translate into solution quality. However, the small amount of disagreement is suggestive of reasonable dual decomposition performance.

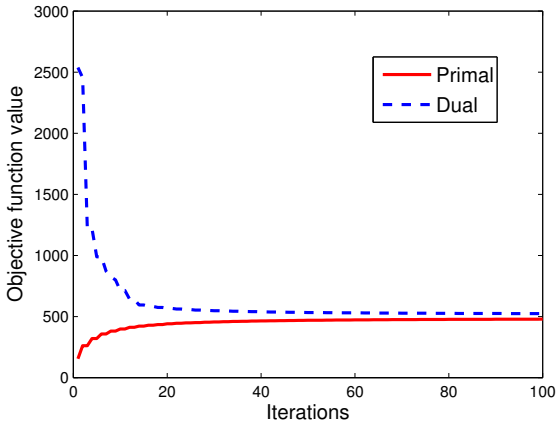

Figure 3. The improvement of the primal and the dual for solving the loss augmented inference versus the subgradient iterations. A typical case is shown.

5. Conclusion

This paper presents a novel approach for optimizing against a large class of non-decomposable performance measures with a Markov network model. The approach can be used for performance measures which can maximized when augmented with a linear term, when the model is a Markov network with unary and pairwise potentials. Observing that the computationally challenging part of most structured prediction approaches is to find the subgradient, which requires solving the loss augmented inference, we proposed an approach to solve the loss augmented inference efficiently. We adopt the idea of dual decomposition and factorize the loss augmented inference into two factors and show how to solve each factor efficiently. The proposed

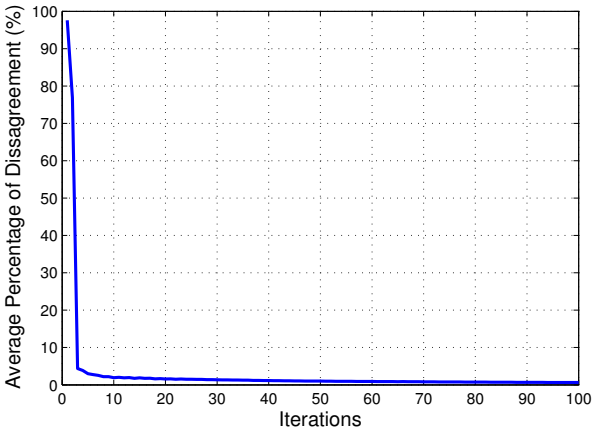

Figure 4. The average percentage of disagreeing assignments versus the number of subgradient iterations. In average only 0.63% of assignments are disagreeing at the last iteration.

approach is applicable to a large range of structured prediction methods. We compared the proposed approach to two strong baselines – one with the same model as ours, but optimizing against decomposable Hamming loss, and the other one optimizing against the same non-decomposable loss function, but using a decomposable model. We showed significant improvements over the baselines on three object segmentation datasets – Pascal VOC 2009, Pascal VOC 2010 and H3D. We also compare the results with a competing approach for solving the same problem and show comparable results with a general purpose algorithm that is 15 times faster.

References

[1] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009. 6

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11):1222–1239, 2001. 5

[3] S. Chakrabarti, R. Khanna, U. Sawant, and C. Bhattacharyya. Structured learning for non-smooth ranking losses. In *KDD*, 2008. 1

[4] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, 2009. 1

[5] T. Do and T. Artieres. Large margin training for hidden markov models with partially observed states. In *ICML*, 2009. 2, 6

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *IJCV*, 88(2), June 2010. 5

[7] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9):1627–1645, 2010. 6

[8] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2), 2004. 5

[9] M. Hoai, Z. Lan, and F. De la Torre. Joint segmentation and classification of human actions in video. In *CVPR*, 2011. 1

[10] T. Joachims. A support vector method for multivariate performance measures. In *ICML*, 2005. 1, 2, 3, 4, 5

[11] N. Komodakis. Efficient training for pairwise or higher order crfs via dual decomposition. In *CVPR*, pages 1841–1848, 2011. 2

[12] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *PAMI*, 33:531–552, March 2011. 4, 7

[13] D. McAllester, T. Hazan, and J. Keshet. Direct loss minimization for structured prediction. In *NIPS*, 2010. 1

[14] O. Meshi, D. Sontag, T. Jaakkola, and A. Globerson. Learning efficiently with approximate inference via dual losses. In *ICML*, 2010. 2, 6

[15] Mosek. The mosek optimization software. http://www.mosek.com, March 2010. 5

[16] M. Ranjbar, G. Mori, and Y. Wang. Optimizing complex loss functions in structured prediction. In *ECCV*, 2010. 2, 5, 6

[17] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *International Conference on Machine Learning*, 2007. 6

[18] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011. 3

[19] M. Szummer, P. Kohli, and D. Hoiem. Learning crfs using graph cuts. In *ECCV*, 2008. 1

[20] D. Tarlow and R. Zemel. Big and tall: Large margin learning with high order losses. In *CVPR 2011 Workshop on Inference in Graphical Models with Structured Potentials*, 2011. 2

[21] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: a large margin approach. In *ICML*, 2005. 1

[22] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *NIPS*, 2003. 1

[23] B. Taskar, S. Lacoste-julien, and M. I. Jordan. Structured prediction via the extragradient method. In *NIPS*, 2005. 1

[24] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004. 2

[25] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, September 2005. 1

[26] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *PAMI*, 32(9):1582–1596, 2010. 6

[27] Y. Wang and G. Mori. Hidden part models for human action recognition: Probabilistic vs. max-margin. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011. 1

[28] T. Werner. A linear programming approach to max-sum problem: A review. *PAMI*, 29(7):1165–1179, 2007. 3, 5

[29] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *30th ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007. 1, 3