Efficiently Enforcing Diversity in Multi-Output Structured Prediction

Abner Guzman-Rivera
University of Illinois

Pushmeet Kohli
Microsoft Research Cambridge

Dhruv Batra
Virginia Tech

Rob A. Rutenbar
University of Illinois

Abstract

This paper proposes a novel method for efficiently generating multiple diverse predictions for structured prediction problems. Existing methods like SDPPs or DivMBest work by making a series of predictions where each prediction is made after considering the predictions that came before it. Such approaches are inherently sequential and computationally expensive. In contrast, our method, Diverse Multiple Choice Learning, *learns* a set of models to make multiple independent, yet diverse, predictions at test-time. We achieve this by including a diversity encouraging term in the loss function used for training the models. This approach encourages diversity in the predictions while preserving computational efficiency at test-time. Experimental results on a number of challenging problems show that our method learns models that not only predict more diverse results than competing methods, but are also able to generalize better and produce results with high test accuracy.

1 Introduction

Classical discriminative approaches for machine learning are designed to produce a single prediction for the variables of interest. In the structured prediction setting, formulations such as Conditional Random Fields (CRFs) [17], Max-Margin Markov Networks (M^3N) [24], and Structured Support Vector Machines (SSVMs) [25] have provided principled models that reason about all output variables and make a joint global prediction, called maximum *a posteriori* (MAP) inference in the context of probabilistic models. However, in a number of settings it might be beneficial (even necessary) to make multiple predictions. This might be due to *Misspecified Models* or *Multi-Modality*.

Model misspecification is caused by (1) the model class being too restrictive to capture reality, *e.g.*, assuming that all interactions between variables are purely associative or attractive; or (2) lack of enough training data to learn the correct model in the model class; or (3) the inability to perform exact inference (estimation of the MAP solution may be NP-hard). The (approximate) MAP solution for such a misspecified model could be far from ground-truth.

Even if the model class is expressive enough, enough training data is available, and computing MAP is tractable, there may be global ambiguities or multiple possible interpretations that explain the input data. This is particularly evident in systems with a user in-the-loop, *e.g.*, interactive image segmentation where the user provides scribbles on an image and the system produces a cutout of the scribbled object [5]. There may be several possible outputs consistent with the user input. The interface could show not just a single prediction but a small set of diverse predictions and simply let the user pick the best one. Observations such as these have motivated the study of techniques for making multiple predictions.

A standard way of producing multiple hypotheses in probabilistic models is to find the top M most probable configurations, called the M-Best MAP problem [19, 27, 10, 2]. Recent works have also considered producing *Diverse* M-Best solutions [4] or *modes* of the Gibbs distribution [7]. However, these methods suffer from two problems. First, there exists a train-test disparity. Specifically, the model is trained to produce a single output, *i.e.*, either to match the data distribution (max-likelihood) or to score ground-truth the highest by a margin (max-margin); yet, it is used at test-time to produce multiple outputs. Second, inference of the diverse M-best solutions is a difficult optimization problem.

Two recent works have been attempted to overcome these problems [16, 11]. Kulesza and Taskar [16] proposed Structured Determinantal Point Process (SDPP), an elegant probabilistic model defined over sets of structured-outputs that naturally favors diversity. Unfortunately, inference in SDPPs is tractable only for low-treewidth models, which limits their applicability. Guzman-Rivera *et al.* [11] proposed a "mixture of experts" model called Multiple Choice Learning (MCL), which learns an M-tuple of predictors, each of which makes a single structured prediction. The

Appearing in Proceedings of the 17^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33.

mixture of predictors is trained to minimize the loss of the most accurate prediction in the output tuple. At test-time, each predictor makes a prediction independently, *i.e.*, without considering the predictions of other predictors. This makes their model quite efficient, and trivially parallelizable. Although MCL produces good results, the model does not have any *explicit* preference for diverse predictions – as long as the regularized risk is minimized, the predictors can make any predictions, including repeated identical ones.

Overview and Contributions. The central thesis of this paper is that diversity can serve as an effective regularizer – leading to possibly worse performance on training data but better generalization on unseen test data. We build on MCL [11] and improve upon it in several ways. We propose a new method for diverse multi-output structured prediction, Diverse Multiple Choice Learning (DivMCL). The method is efficient, explicitly models diversity, and provides a consistent train-test procedure. Unlike SDPPs, DivMCL is applicable in any structured-output space that allows efficient MAP inference (*e.g.*, binary pairwise supermodular MRFs). Unlike (Diverse) M-Best methods, DivMCL *learns* to produce multiple diverse solutions. Our method is similar to MCL in the sense that it learns a set of models to make multiple independent predictions at test-time. However, unlike MCL, the models are *explicitly* trained to produce results that are diverse. We achieve this by including a diversity encouraging term in the loss function used for training the models. Our approach then, enforces diversity in the predictions while preserving computational efficiency at test-time.

Our experimental results on a number of challenging problems confirm that DivMCL learns models that not only lead to diverse predictions, but more importantly, as a direct consequence of this achieved diversity, DivMCL produces results with high test accuracy and is able to generalize better than other multi-output prediction methods.

Learning to Enforce Explicit Diversity. Our thesis is similar in spirit to recent work on structured prediction, which shows how low-order models can be trained using high-order loss functions [23, 21].

2 Preliminaries

We begin by establishing notation and reviewing standard (single-output) structured prediction.

Notation. For any positive integer n, let $[n]$ be shorthand for the set $\{1, 2, \ldots, n\}$. Given a training set of input-output pairs $\left\{(\mathbf{x}_i, \mathbf{y}_i) \,|\, \mathbf{x}_i \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y}^k, i \in [n]\right\}$, we are interested in learning a mapping $f : \mathcal{X} \to \mathcal{Y}^k$ where $\mathcal{X}$ is the space of inputs (*e.g.*, images or Chinese sentences) and $\mathcal{Y}^k$ is the space of (k-variate and structured) outputs (*e.g.*, image segmentations or English translations of Chinese sentences).

Structured Support Vector Machines (SSVMs). In an SSVM setting, the mapping is linear and is defined as

$$f(\mathbf{x}; \mathbf{w}) = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}^k} \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y}) \,, \tag{1}$$

where $\boldsymbol{\psi}(\mathbf{x}, \mathbf{y})$ is a joint feature map, $\boldsymbol{\psi} : \mathcal{X} \times \mathcal{Y}^k \to \mathbb{R}^d$.

The quality of the prediction $\hat{\mathbf{y}}_i = f(\mathbf{x}_i)$ is measured by a task-specific loss function

$$\ell : \mathcal{Y}^k \times \mathcal{Y}^k \to \mathbb{R}^+ \,, \tag{2}$$

where $\ell(\mathbf{y}_i, \hat{\mathbf{y}}_i)$ denotes the cost of predicting $\hat{\mathbf{y}}_i$ when the correct label is $\mathbf{y}_i$. Some examples of loss functions are the intersection/union criterion used by the PASCAL Visual Object Category Segmentation Challenge [9], and the BLEU score used to evaluate machine translations [20].

For ease of notation, let $\ell_i(\cdot)$ be shorthand for $\ell(\mathbf{y}_i, \cdot)$; $\boldsymbol{\psi}_i(\cdot)$ be shorthand for $\boldsymbol{\psi}(\mathbf{x}_i, \cdot)$; and $\boldsymbol{\delta\psi}_i(\mathbf{y})$ be shorthand for $\boldsymbol{\psi}_i(\mathbf{y}) - \boldsymbol{\psi}_i(\mathbf{y}_i)$.

The task-loss ℓ is typically non-convex and non-continuous in $\mathbf{w}$, and thus Tsochantaridis *et al.* [25] proposed to optimize a regularized surrogate loss function

$$\min_{\mathbf{w}} \quad \frac{\lambda}{2} \|\mathbf{w}\|_2^2 + \sum_{i \in [n]} \hbar_i(\mathbf{w}) \,, \tag{3}$$

where λ is the regularization parameter and $\hbar_i(\cdot)$ is the structured hinge-loss

$$\hbar_i(\mathbf{w}) = \max_{\mathbf{y}} \left\{\ell_i(\mathbf{y}) + \mathbf{w}^T \boldsymbol{\delta\psi}_i(\mathbf{y})\right\}. \tag{4}$$

Objective (3) is a convex optimization problem which may be solved using (Stochastic) Subgradient [22], or Cutting-Plane [12] approaches, whenever an efficient separation-oracle for solving the maximization in (4) is available.

3 Multi-Output Structured Prediction

In this work, we study multiple-output structured prediction and the problem of learning a function

$$g : \mathcal{X} \to \mathcal{Y}^{kM} \,, \tag{5}$$

where the input space $\mathcal{X}$ is as before but where the output space $\mathcal{Y}^{kM}$ is an M-tuple[1] of structured-outputs (*e.g.*, multiple segmentations of an image or multiple English translations of a Chinese sentence).

In a manner similar to single-output SSVMs, we are interested in a linear mapping

$$g(\mathbf{x}; \mathbf{W}) = \operatorname*{argmax}_{\mathbf{Y} \in \mathcal{Y}^{kM}} \mathbf{W}^T \boldsymbol{\Psi}(\mathbf{x}, \mathbf{Y}) \,, \tag{6}$$

[1] Our formulation is described with a *nominal* ordering of the predictions. However, both the proposed objective function and optimization algorithm are invariant to permutations of this ordering.

where $\mathbf{Y} = \langle \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)} \rangle$ is an M-tuple of structured-outputs, *i.e.*, each $\mathbf{y}^{(i)} \in \mathcal{Y}^k$; and the joint feature map $\boldsymbol{\Psi} : \mathcal{X} \times \mathcal{Y}^{kM} \to \mathbb{R}^d$ is a function of the input and an M-tuple of structured-outputs.

We will assume the model factors such that the score for a tuple is the sum of scores of each item in the tuple:

$$g(\mathbf{x}; \mathbf{W}) = \operatorname*{argmax}_{\mathbf{Y} \in \mathcal{Y}^{kM}} \sum_{m=1}^{M} \mathbf{w}_m^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y}^{(m)}) \,. \tag{7}$$

That is, while each element of the tuple is structured, the model's predictions are independent of each other at test-time. MAP inference is thus very efficient compared to (6) and can be trivially parallelized.

3.1 Multi-Output Loss

Let $\hat{\mathbf{Y}}_i$ be the model prediction for input $\mathbf{x}_i$, *i.e.*,

$$\hat{\mathbf{Y}}_i = g(\mathbf{x}_i; \mathbf{W}) = \left\langle \hat{\mathbf{y}}_i^{(1)}, \ldots, \hat{\mathbf{y}}_i^{(M)} \right\rangle \,. \tag{8}$$

In the single-output SSVM we assumed the availability of a "ground-truth" output $\mathbf{y}_i$ for each datapoint and a pairwise task-loss (2) measuring the quality of the single-output prediction $\hat{\mathbf{y}}_i$ w.r.t. $\mathbf{y}_i$. Here, we do not assume the availability of ground-truth output tuples. Further, we address scenarios where there is no *natural* choice of ground-truth (*e.g.*, interactive segmentation), and would also like to be able to apply our method to standard datasets which usually have single ground-truth labels. For these reasons and following previous work [11] we make use of the "oracle" or "hindsight" set-loss

$$\mathcal{L}_i(\hat{\mathbf{Y}}_i) = \min_{m \in [|\hat{\mathbf{Y}}_i|]} \ell_i(\hat{\mathbf{y}}_i^{(m)}) \,, \tag{9}$$

i.e., the tuple of predictions $\hat{\mathbf{Y}}_i$ incurs loss only for the *most accurate* prediction it contains. This implicitly promotes diversity because individual predictors can specialize to different "domains" of input-output mappings without paying a penalty for being too diverse or inaccurate – as long as the tuple contains at least one accurate prediction. However, the loss does not *explicitly* reward or enforce diversity in the predictions. As long as the loss is minimized, the predictors can make any predictions, including repeated identical ones. This is potentially wasteful, and as we show in our experiments, can be outperformed by our approach which explicitly enforces diversity in the predictions.

3.2 Diverse Multi-Output Loss

To explicitly encourage diversity in the output-tuple, we augment the loss function to penalize tuples of results that lack diversity w.r.t. each other. Formally, our diversity-augmented loss has the form

$$\mathcal{L}_i^{+div}(\hat{\mathbf{Y}}_i) = \min_{m \in [|\hat{\mathbf{Y}}_i|]} \ell_i(\hat{\mathbf{y}}_i^{(m)}) + \alpha \, \ell^{div}(\hat{\mathbf{Y}}_i) \,, \tag{10}$$

where $\ell^{div}(\cdot)$ is the diversity encouraging augment and $\alpha \geq 0$ is a constant that controls the trade-off between task-loss and diversity.

The loss (10) allows for arbitrary diversity encouraging functions. However, we will be able to learn a predictor of the form (7) to minimize the diversity-augmented risk only for cases allowing efficient loss-augmented inference as we will see in Section 4.

As an example of a diversity encouraging function consider the following definition,

$$\ell^{div}(\mathbf{Y}) = \bigoplus_{\substack{m<m': \\ m,m' \in [|\mathbf{Y}|]}} \left(1 - \Delta^{div}(\mathbf{y}^{(m)}, \mathbf{y}^{(m')})\right) \,, \tag{11}$$

where $\bigoplus$ is an operator like mean or $\min$ and $\Delta^{div} \in [0, 1]$ is a function that measures dissimilarity between two predictions (*e.g.*, normalized Hamming distance). Thus, $\ell^{div}(\cdot)$ measures the (negative) dissimilarity of all pairs in the output-tuple and aggregates the measures using $\bigoplus$.

Note that while the above considers dissimilarity between all pairs of predictors, we could also consider a sparse graph on the predictors, *e.g.*, a tree on M vertices.

4 Minimizing the Diversified Risk

Here we show how we can learn the parameters for the mapping g in (7), given a set of training instances $\{(\mathbf{x}_i, \mathbf{y}_i) \,|\, \mathbf{x}_i \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y}^k, i \in [n]\}$. Following the Empirical Risk Minimization Principle [26], we would like the parameters to minimize the *diversified-risk* over the training set

$$\min_{\mathbf{W}} \quad \frac{1}{n} \sum_{i \in [n]} \mathcal{L}_i^{+div}(g(\mathbf{x}_i; \mathbf{W})) \,. \tag{12}$$

Note that regularization will only be included later as directly regularizing the risk for a linear classifier is meaningless [18].

4.1 Block-Coordinate Descent for Learning Joint Diverse Predictor

We propose to optimize (12) via a block-coordinate descent approach where each block-coordinate corresponds to the weights $\mathbf{w}_m$ associated with one element of the output-tuple. First, we will rewrite (12) for the case where $\hat{\mathbf{y}}^{(m')} : m' \neq m$ are fixed (and $\hat{\mathbf{y}}^{(m)}$ is allowed to change). Next, we will develop a surrogate objective for the rewritten risk and define the coordinate update.

Recall that $\hat{\mathbf{Y}}_i$ is the model prediction for example i, (8). Two additional definitions will be useful in the subsequent discussion. Let,

$$\hat{\mathbf{Y}}_{i\,-m} = \left\langle \hat{\mathbf{y}}_i^{(1)}, \ldots, \hat{\mathbf{y}}_i^{(m-1)}, \hat{\mathbf{y}}_i^{(m+1)}, \ldots, \hat{\mathbf{y}}_i^{(M)} \right\rangle \tag{13}$$

be the output-tuple with the m-th element removed; and let,

$$\hat{\mathbf{Y}}_i|^{\mathbf{y}}_m = \left\langle \hat{\mathbf{y}}_i^{(1)}, \ldots, \hat{\mathbf{y}}_i^{(m-1)}, \mathbf{y}, \hat{\mathbf{y}}_i^{(m+1)}, \ldots, \hat{\mathbf{y}}_i^{(M)} \right\rangle \quad (14)$$

be the output-tuple with the m-th element replaced by $\mathbf{y}$.

Then,

$$\mathcal{L}_i^{+div\,(m)}(\hat{\mathbf{y}}^{(m)}; \hat{\mathbf{Y}}_{i\,-m}) = \begin{cases} \ell_i(\hat{\mathbf{y}}_i^{(m)}) + \alpha\, \ell^{div}(\hat{\mathbf{Y}}_i|^{\hat{\mathbf{y}}^{(m)}}_m) & \text{if } \ell_i(\hat{\mathbf{y}}_i^{(m)}) \\ & \quad < \mathcal{L}_i(\hat{\mathbf{Y}}_{i\,-m}) \\ \mathcal{L}_i(\hat{\mathbf{Y}}_{i\,-m}) + \alpha\, \ell^{div}(\hat{\mathbf{Y}}_i|^{\hat{\mathbf{y}}^{(m)}}_m) & \text{otherwise} \end{cases} \quad (15)$$

is equivalent to the diversity-augmented loss (10) whenever $\hat{\mathbf{Y}}_{i\,-m}$ remains constant (and thus, $\mathcal{L}_i(\hat{\mathbf{Y}}_{i\,-m})$ also remains constant). This truncated form says that when $\hat{\mathbf{y}}^{(m)}$ is the minimizer of $\ell_i(\cdot)$ the loss behaves linearly w.r.t. ground-truth. Otherwise, the loss is constant w.r.t. to ground-truth.

In the case of fixed $\hat{\mathbf{y}}^{(m')} : m' \neq m$ we can rewrite (12) as

$$\min_{\mathbf{w}_m} \sum_{i \in [n]} \mathcal{L}_i^{+div\,(m)}(\hat{\mathbf{y}}^{(m)}; \hat{\mathbf{Y}}_{i\,-m}) \,, \quad (16)$$

where $\hat{\mathbf{y}}^{(m)} = \operatorname{argmax}_{\mathbf{y}} \mathbf{w}_m^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y})$; and where we have restricted minimization to $\mathbf{w}_m$ and dropped constant $\frac{1}{n}$. This is the coordinate-update we focus on for the remaining of this section.

We will first develop a surrogate for coordinate-update (16) and then show how it can be minimized. It is important to highlight that we do not have a single surrogate for (12) but rather a surrogate for the minimization w.r.t. each $\mathbf{w}_m$ coordinate.

There are two natural surrogates for the truncated loss (15):

$$\mathcal{L}_{i\ \text{linear}}^{+div\,(m)}(\hat{\mathbf{y}}^{(m)}; \hat{\mathbf{Y}}_{i\,-m}) = \ell_i(\hat{\mathbf{y}}^{(m)}) + \alpha\, \ell^{div}(\hat{\mathbf{Y}}_i|^{\hat{\mathbf{y}}^{(m)}}_m) \,, \quad (17)$$

$$\mathcal{L}_{i\ \text{const}}^{+div\,(m)}(\hat{\mathbf{y}}^{(m)}; \hat{\mathbf{Y}}_{i\,-m}) = \mathcal{L}_i(\hat{\mathbf{Y}}_{i\,-m}) + \alpha\, \ell^{div}(\hat{\mathbf{Y}}_i|^{\hat{\mathbf{y}}^{(m)}}_m) \,. \quad (18)$$

These correspond to the linear and constant regimes (the two cases) in (15).

Next, we define hinge-type upper-bounds for both surrogates. For the linear case an upper-bound is

$$\begin{aligned} &\tilde{\mathcal{L}}_{i\ \text{linear}}^{+div\,(m)}(\mathbf{w}_m; \hat{\mathbf{Y}}_{i\,-m}) \\ &= \max_{\bar{\mathbf{y}}} \left[\mathbf{w}_m^T \boldsymbol{\delta\psi}_i(\bar{\mathbf{y}}) + \mathcal{L}_{i\ \text{linear}}^{+div\,(m)}(\bar{\mathbf{y}}; \hat{\mathbf{Y}}_{i\,-m}) \right] \\ &= \max_{\bar{\mathbf{y}}} \left[\mathbf{w}_m^T \boldsymbol{\delta\psi}_i(\bar{\mathbf{y}}) + \ell_i(\bar{\mathbf{y}}) + \alpha\, \ell^{div}(\hat{\mathbf{Y}}_i|^{\bar{\mathbf{y}}}_m) \right]; \end{aligned} \quad (19)$$

Algorithm 1 Learn Joint Diverse Predictor.

Input: $S = \{(\mathbf{x}_i, \mathbf{y}_i) : i \in n\}$, M, λ, α
Initialize $I_m : m \in [M]$ at random.
for $m = 1, \ldots, M$ **do**
  $\mathbf{w}_m \leftarrow$ Update$(I_m : m \in [M], \lambda, \alpha'{=}0)$ see (21)
end for
repeat
  for $m = 1, \ldots, M$ **do**
    Recompute $I_m : m \in [M]$ using (22)
    $\mathbf{w}_m \leftarrow$ Update$(I_m : m \in [M], \lambda, \alpha)$ see (21)
  end for
until No I_m changed in last M updates.
return $\mathbf{W}$

and an upper-bound for the constant surrogate is

$$\begin{aligned} &\tilde{\mathcal{L}}_{i\ \text{const}}^{+div\,(m)}(\mathbf{w}_m; \hat{\mathbf{Y}}_{i\,-m}) \\ &= \max_{\bar{\mathbf{y}}} \left[\alpha\, \mathbf{w}_m^T \boldsymbol{\delta\psi}_i(\bar{\mathbf{y}}) + \mathcal{L}_{i\ \text{const}}^{+div\,(m)}(\bar{\mathbf{y}}; \hat{\mathbf{Y}}_{i\,-m}) \right] \\ &= \max_{\bar{\mathbf{y}}} \left[\alpha\, \mathbf{w}_m^T \boldsymbol{\delta\psi}_i(\bar{\mathbf{y}}) + \mathcal{L}_i(\hat{\mathbf{Y}}_{i\,-m}) + \alpha\, \ell^{div}(\hat{\mathbf{Y}}_i|^{\bar{\mathbf{y}}}_m) \right], \end{aligned} \quad (20)$$

where $\mathcal{L}_i(\hat{\mathbf{Y}}_{i\,-m})$ is constant. Note that we purposely scale the hinge for the constant case differently. This ensures that as $\alpha \to 0$, the optimization tends to the case of no diversity-augment (*i.e.*, to the MCL model).

Putting both surrogates together and adding regularization we define the **Update** for $\mathbf{w}_m$ as the minimizer of

$$\begin{aligned} \min_{\mathbf{w}_m} \ &\frac{\lambda}{2} \|\mathbf{w}_m\|_2^2 + \sum_{i \in I_m} \tilde{\mathcal{L}}_{i\ \text{linear}}^{+div\,(m)}(\mathbf{w}_m; \hat{\mathbf{Y}}_{i\,-m}) \\ &+ \sum_{m' \neq m} \sum_{i \in I_{m'}} \tilde{\mathcal{L}}_{i\ \text{const}}^{+div\,(m')}(\mathbf{w}_m; \hat{\mathbf{Y}}_{i\,-m}) \,, \end{aligned} \quad (21)$$

where $I_m, I_{m'}$ are example-index sets which correspond to a choice of surrogate (linear or constant) for each example. In our implementation, before every coordinate step we split the dataset into two disjoint groups:

$$\text{One set } I_m = \left\{ i : \ell_i(\hat{\mathbf{y}}^{(m)}) = \mathcal{L}_i(\hat{\mathbf{Y}}_i) \right\}, \text{ and} \quad (22a)$$

$$M{-}1 \text{ sets } I_{m'} = \left\{ i : \ell_i(\hat{\mathbf{y}}^{(m')}) = \mathcal{L}_i(\hat{\mathbf{Y}}_i) \right\} \quad (22b)$$

where $m' \neq m$. That is, we determine which predictor best explains each example and use the linear surrogate only for those examples better explained by the m-th predictor.

Update (21) is a convex optimization problem which can be solved using standard methods such as Stochastic Subgradient or Cutting-Plane algorithms.

Algorithm 1 summarizes our learning algorithm and provides an initialization strategy for weights $\mathbf{w}_m$ using a random split of the data and no diversity augment ($\alpha'{=}0$).

Loss-Augmented Inference. Equations (19) and (20) are

similar to standard SSVM hinge-loss formulations. The main difference is the diversity-augment in the loss. For instance, if in the definition of ℓ^{div} (11), we use mean for aggregation then (19) becomes

$$\max_{\bar{\mathbf{y}}} \left[\mathbf{w}_m^T \boldsymbol{\psi}(\mathbf{x}_i, \bar{\mathbf{y}}) + \ell_i(\bar{\mathbf{y}}) - \frac{\alpha}{\pi} \sum_{m'' \neq m} \Delta^{div}(\bar{\mathbf{y}}, \hat{\mathbf{y}}_i^{(m'')}) \right] + \begin{matrix} \text{terms not} \\ \text{dependent on } \bar{\mathbf{y}} \end{matrix}, \tag{23}$$

where π is the constant used for computing the mean. If Δ^{div} is a decomposable function (*e.g.*, Hamming distance), then loss-augmented inference maintains the same efficiency of MAP inference.

5 Experiments

Setup. We tested DivMCL on two problems: i) foreground-background segmentation in image collections and ii) protein side-chain prediction. In both problems making a single perfect prediction is difficult due to inherent ambiguity in the tasks. Moreover, inference-time computing limitations force us to learn restricted models (pairwise attractive MRFs, in the case of foreground-background segmentation); or resort to approximate inference (TRW-S [14], in the case of protein side-chain prediction). We will compare the ability of DivMCL to produce sets of hypotheses which contain more accurate predictions than other methods for producing multiple (diverse) hypotheses.

5.1 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.*, images from a baseball game, a safari *etc*.). The dataset provides pixel-level ground-truth foreground-background segmentations for each image. We used 9 difficult groups from iCoseg containing 166 images in total. These images were split at random into 5-folds of roughly equal size. See Fig. 1 for some example images and segmentations.

Model and Features. The segmentation task is modeled as a binary pairwise MRF where each node corresponds to a superpixel [1] in the image. We extracted 12-dim color features at each superpixel (mean RGB; mean HSV; 5 bin Hue histogram; Hue histogram entropy). The edge features, computed for each pair of adjacent superpixels, correspond to a standard Potts model and a contrast sensitive Potts model. The weights at each edge were constrained to be positive so that the resulting supermodular potentials could be maximized via graph-cuts [6, 15].

Choice of Loss. The task-loss in this experiment (ℓ) is the fraction of incorrectly labeled nodes; Δ in the diversity-augment is hamming-distance; and $\bigoplus$ is the mean operator. For evaluation we use the set-loss, $\mathcal{L}$ (9), *i.e.*, the error of the best segmentation in the output-tuple.

Baselines and Evaluation. We compare our algorithm against two alternatives for producing diverse predictions: i) Single SSVM + Diverse M-Best MAP [4], and ii) Multiple Choice Learning (MCL) [11]. Diverse M-Best MAP is a sequential algorithm for generating diverse high scoring solutions from MRFs. The first solution corresponds to the MAP solution and subsequent solutions are computed from a modified energy: The original MRF compounded with a linear penalty on the similarity of the next solution w.r.t. previous solutions. The algorithm takes a parameter δ which is a trade-off between the original energy and the similarity penalty. For the MCL baseline we use our implementation of DivMCL with α=0.

All experiments in this section were repeated 5 times and averages are reported. Each time models were trained on 1 fold and the remaining 4 folds were used for testing and validation. In general 2 folds are used for validation and 2 for testing; but in a few occasions we used the 4 remaining folds for validation.

MCL requires an initial "assignment" of examples to predictors. We provide a random assignment. For DivMCL we use the initialization strategy in Algorithm 1.

For Diverse M-Best MAP, we trained a single SSVM on 1 fold, used another 2 folds to select the best diversity parameter δ, and test on the remaining 2 folds. We implemented algorithm DivM-Best [4] using dynamic graph-cuts [13].

Optimization. On this task we performed coordinate update (21) using our implementation of Cutting-Plane.

5.1.1 Experiments with coarsened superpixels.

We ran a series of experiments with coarsened superpixels (*i.e.*, full-images but large superpixels, roughly $\sim$100 per image) so as to observe trends as we vary multiple parameters. We varied the number of predictions M; the strength of regularization λ; and the strength of the diversity-augment α.

Effect of α. In Fig. 2 we show trends as α increases. These experiments were repeated 5 times training on 1 of 5 random folds and testing on the remaining 4. When regularization is low (*i.e.*, small λ) our approach leads to increased performance for both train and test. This is partly explained due to coordinate-descent getting stuck in poor local-optima for α=0, and α>0 often leading the optimization to better optima.

With strong regularization trends are different and we mainly see α modulating fit to train. This is to be expected as we conjectured earlier. Further, for all λ and all M our approach (on average) leads to test-time improvements.

Behaviour of Coordinate Descent. Fig. 3a shows average trends as the optimization progresses during learning. As

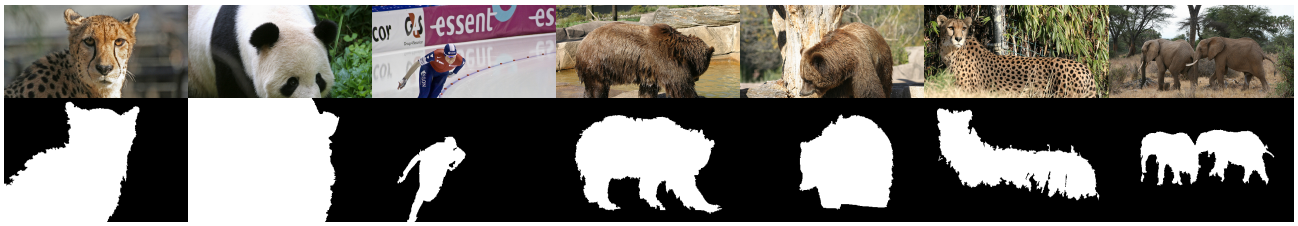
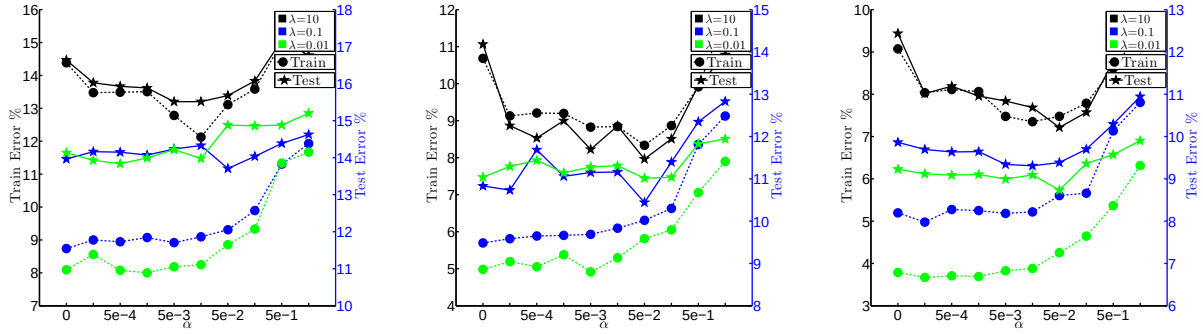


Figure 1: First row: input images. Second row: corresponding ground-truth segmentations.



(a) M=4. (b) M=8. (c) M=12.

Figure 2: For $M \in \{4, 8, 12\}$, experiments on coarsened segmentation showing trends as regularization parameter λ; and diversity-parameter α are varied. We show averages over 5 random folds (1 fold used for training and the remaining for testing). Circles correspond to train-error (left axis) and stars to test-error (right axis). We note that for all λ and all M the proposed approach leads to improved test-performance. At low regularization the approach leads to improved train and test performance. As expected, at high regularization α modulates fit to train and often leads to improved test performance.

above, the plots are averages over 5 random folds. Here we set M and λ to intermediate values for illustration. In squares (left axis) we show the (normalized) *diversified-risk* (12). Every iteration is a descent-step and learning is halted whenever a step does not reduce the risk. In this example we see that $\alpha>0$ always leads to reduced fit to train, yet may also lead to test-time improvements.

Behaviour as M increases. In Fig. 3b we investigate the benefits achievable by the approach as M increases. As an upper-bound on performance we pick the best performing α for each fold before averaging (*i.e.*, we tune α on test data). We had conjectured higher benefits would correspond to higher M but the experiments show more of a constant improvement. Next, in Fig. 3c we repeat the experiment validating on held-out data. That is, for each of the 5 experiments we train on 1 fold, validate on 2 folds and test on the remaining 2. In this set of experiments validation gives good results with our approach improving on the MCL baseline.

Note that at $M \geq 16$ we observed that $\alpha=0$ leads to few iterations of coordinate descent. This means the optimization quickly finds a local optimum and is unable to progress further. This also explains why, for large M, $\alpha>0$ leads again to better performance on train data.

5.1.2 Comparison against Baselines.

For these set of experiments we perform segmentation on regular superpixels (roughly $\sim 3,000$ of them per image), and compare against baselines MCL and Diverse M-Best MAP.

First, in Fig. 4a we observe trends as α increases for two settings of the regularization parameter, $\lambda \in \{0.01, 0.1\}$, and M=12. Note that the MCL baseline corresponds to the left-most points (*i.e.*, α=0). We observe the same trend as before where increasing α reduces fit to train, but at the same time leads to superior performance on test data for a range of values of α.

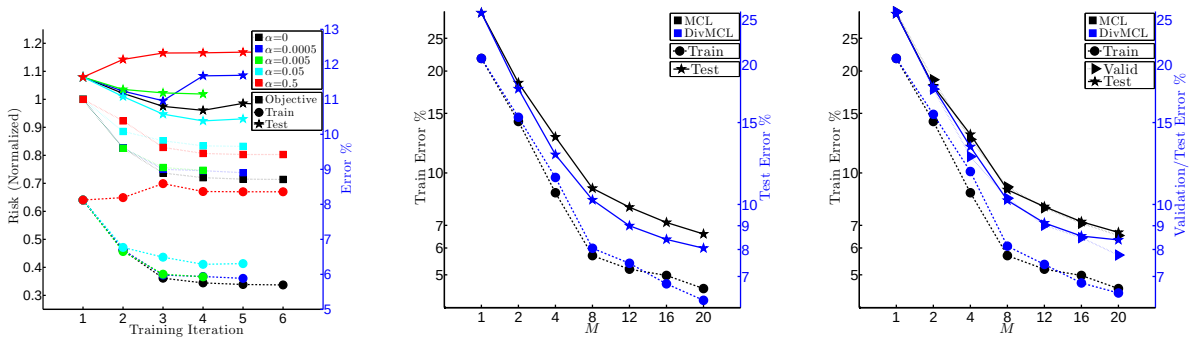
Fig. 4b and 4c show the comparison against both baselines for λ=0.1, 0.01, respectively. For Diverse M-Best MAP and DivMCL we tune their respective parameters on held-out data.

Both MCL and DivMCL always outperform Diverse M-Best MAP. This is to be expected as MCL and DivMCL are trained to predict multiple hypotheses. DivMCL is on average better than MCL.

Here, we see further confirmation that DivMCL is able to maintain performance for larger ranges of regularization strengths. That is, MCL overfits for lower values of λ and M than DivMCL does.

5.2 Protein Side-Chain Prediction

Model and Dataset. Given a protein backbone structure, the task here is to predict the amino acid side-chain configurations. This side-chain prediction problem has been traditionally formulated as a pairwise MRF with node labels corresponding to (discretized) side-chain configura-

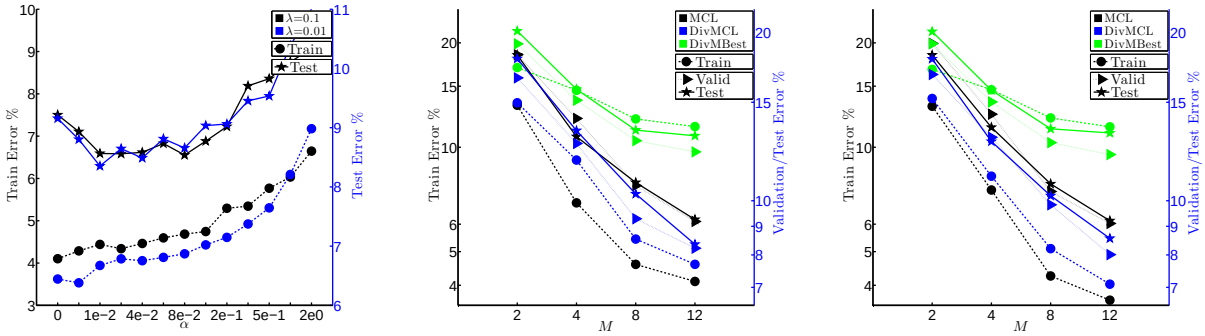


(a) Objective and performance vs. training iteration (M=8, λ=0.1).

(b) Comparison against MCL, best α (λ=0.1).

(c) Comparison against MCL, α from validation (λ=0.1).

Figure 3: Experiments on coarsened segmentation (averages over 5 random folds). (a) Behavior of coordinate descent (M=8, λ=0.1). The *diversified-risk* is plotted on the left-axis (squares); and train, test errors are plotted on the right-axis (circles, stars). Note how α modulates fit to train nicely and that certain values of α lead to test-time improvements. (b) Comparison against MCL baseline as M increases, α tuned on test. (c) Comparison against MCL baseline as M increases, α tuned on held-out data. We observe that the proposed approach leads to performance improvements over a range of M.



(a) Average performance vs. α (M=12). (b) Comparison against baselines, λ=0.1. (c) Comparison against baselines, λ=0.01.

Figure 4: Experiments on segmentation (averages over 5 random folds). Circles correspond to train-error (left axis); stars to test-error (right axis) and, where applicable, triangles to validation-error (right axis). (a) Behavior of coordinate descent. Note how α modulates fit to train nicely and that multiple values of α lead to test-time improvements. (b,c) Comparison against MCL baseline as M increases, α tuned on held-out data.

tions (*rotamers*). The MRFs include pairwise interactions between nearby side-chains, and between side-chains and backbone. We use the dataset of [8] and inherit their test/train split and evaluation metric.[2] The dataset consists of 276 proteins split into train and test sets of sizes 55 and 221 respectively. The energy function is defined as a weighted sum of eight known energy terms. The weights for the energy terms are the parameters to be learned. We used TRW-S [14] for inference.

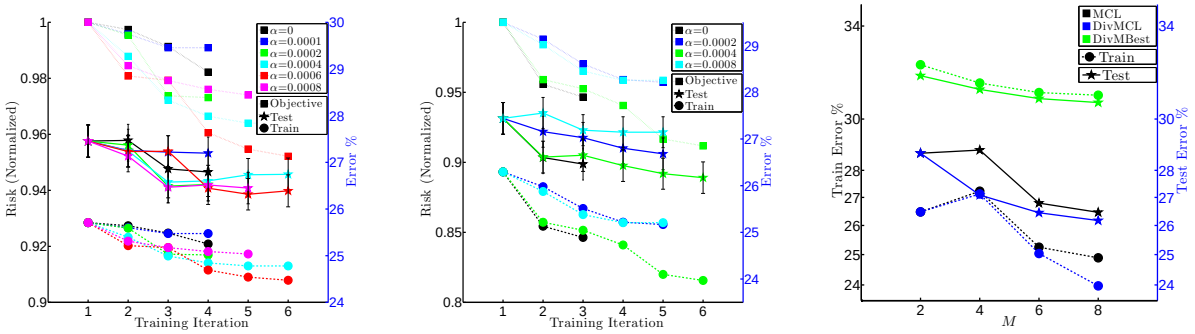
Baselines and Evaluation. We compare to the same baselines as in the previous application: i) Single SSVM + Diverse M-Best MAP [4], and ii) Multiple Choice Learning (MCL) [11]. We use the initialization strategies outlined in the previous section. For both DivMCL and Diverse M-Best MAP we report results with each algorithm's parameter tuned on test data. Following [8], we report average error rates for the first two angles (χ_1 and χ_2) on all test proteins.

Optimization. For this dataset we experimented with both Cutting-Plane and Stochastic Subgradient (SSG) methods when performing update (21). We obtained better results with SSG and conjecture this may be partly due to the use of approximate loss-augmented inference.

Results. In Fig. 5a and 5b we investigate the behavior of coordinate-descent together with train and test errors. The squares (left axis) plot the (normalized) *diversified-risk* (12). Every iteration is a descent-step and learning is halted whenever a step does not reduce the risk. For this dataset, similar to coarsened segmentation with high regularization, we often see that the diversity-augment leads to improved train (circles) and test (stars) performances.

In Fig. 5c we compare against the baselines. Again, MCL and DivMCL outperform Diverse M-Best MAP. DivMCL is able to outperform MCL.

[2]Dataset available from: http://cyanover.fhcrc.org/recomb-2007/



(a) Error vs. training iteration, M=6. (b) Error vs. training iteration, M=8. (c) Error vs. M. Comparison with baselines.

Figure 5: Experiments on protein side-chain prediction (λ=1). (a,b) Show the behavior of coordinate descent. The *diversified-risk* is plotted on the left-axis (squares); and train, test errors are plotted on the right-axis (circles, stars). For this dataset, α>0 often leads to train and test-time improvements. (c) The proposed approach (blue) outperforms the Diverse M-Best MAP (green) and MCL (black) baselines.

6 Discussion and Conclusions

We presented a new method for learning a predictor that outputs a tuple of *diverse* (structured) hypotheses. The proposed model is a set of predictors which are explicitly trained to prefer diverse output-tuples, yet make independent predictions at test-time.

We proposed and investigated the idea of augmenting the risk at train-time with a penalty for lack of diversity. The diversity-augment couples the parameters of the predictors but we show how an efficient iterative learning procedure is able to minimize the proposed *diversified-risk*.

There are a number of directions to extend this work. While we evaluated performance of all algorithms in terms of "oracle" set-loss, it would be interesting to measure the impact of DivMCL and other baselines on user experience or final-stage performance in cascaded algorithms.

Another direction for future work is the study of efficient models of structure in the output-tuple. In this work we designed the predictors to be independent, mainly for efficiency reasons. However, further gains in accuracy would be possible if we could afford to couple the predictions at test time. We leave to future work the investigation of efficient methods to couple the prediction tuples at test time.

Acknowledgments

This work was supported in part by Systems On Nanoscale Information fabriCs (SONIC), one of the six SRC STARnet Centers sponsored by MARCO and DARPA; and by the National Science Foundation Grant No. IIS-1353694, awarded to DB.

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] D. Batra. An Efficient Message-Passing Algorithm for the M-Best MAP Problem. In *UAI*, 2012. 1

[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. 1, 5, 7

[5] Y. Boykov and M.-P. Jolly. Interactive Graph Cuts for Optimal Boundary And Region Segmentation of Objects in N-D Images. In *ICCV*, 2001. 1

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

[7] C. Chen, V. Kolmogorov, Y. Zhu, D. Metaxas, and C. H. Lampert. Computing the M Most Probable Modes of a Graphical Model. In *AISTATS*, 2013. 1

[8] 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. 7

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html. 2

[10] M. Fromer and A. Globerson. An LP View of the M-best MAP problem. In *NIPS*, 2009. 1

[11] A. Guzman-Rivera, D. Batra, and P. Kohli. Multiple Choice Learning: Learning to Produce Multiple Structured Outputs. In *NIPS*, 2012. 1, 2, 3, 5, 7

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

[13] P. Kohli and P. H. S. Torr. Measuring uncertainty in graph cut solutions. *CVIU*, 112(1):30–38, 2008. 5

[14] V. Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *PAMI*, 28(10):1568–1583, 2006. 5, 7

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

[16] A. Kulesza and B. Taskar. Structured Determinantal Point Processes. In *NIPS*, 2010. 1

[17] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, 2001. 1

[18] D. Mcallester and J. Keshet. Generalization Bounds and

Consistency for Latent Structural Probit and Ramp Loss. In *NIPS*, 2011. 3

[19] D. Nilsson. An efficient algorithm for finding the M most probable configurations in probabilistic expert systems. *Statistics and Computing*, 8:159–173, 1998. 1

[20] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318, 2002. 2

[21] P. Pletscher and P. Kohli. Learning Low-order Models for Enforcing High-order Statistics. In *AISTATS*, 2012. 2

[22] N. Ratliff, J. A. D. Bagnell, and M. Zinkevich. (Online) Subgradient methods for Structured Prediction. In *AISTATS*, 2007. 2

[23] D. Tarlow, R. P. Adams, and R. S. Zemel. Randomized Optimum Models for Structured Prediction. In *AISTATS*, 2012. 2

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

[25] 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

[26] V. N. Vapnik. *Statistical learning theory*. Wiley, 1 edition, 1998. 3

[27] C. Yanover and Y. Weiss. Finding the M Most Probable Configurations Using Loopy Belief Propagation. In *NIPS*, 2003. 1