Entropy-based Latent Structured Output Prediction

Diane Bouchacourt, Sebastian Nowozin, M. Pawan Kumar

Entropy-based Latent Structured Output Prediction

Diane Bouchacourt
CentraleSupélec and INRIA Saclay
diane.bouchacourt@ecp.fr

Sebastian Nowozin
Microsoft Research
Sebastian.Nowozin@microsoft.com

M. Pawan Kumar
CentraleSupélec and INRIA Saclay
pawan.kumar@ecp.fr

Abstract

Recently several generalizations of the popular latent structural SVM framework have been proposed in the literature. Broadly speaking, the generalizations can be divided into two categories: (i) those that predict the output variables while either marginalizing the latent variables or estimating their most likely values; and (ii) those that predict the output variables by minimizing an entropy-based uncertainty measure over the latent space. In order to aid their application in computer vision, we study these generalizations with the aim of identifying their strengths and weaknesses. To this end, we propose a novel prediction criterion that includes as special cases all previous prediction criteria that have been used in the literature. Specifically, our framework's prediction criterion minimizes the Aczél and Daròczy entropy of the output. This in turn allows us to design a learning objective that provides a unified framework (UF) for latent structured prediction. We develop a single optimization algorithm and empirically show that it is as effective as the more complex approaches that have been previously employed for latent structured prediction. Using this algorithm, we provide empirical evidence that lends support to prediction via the minimization of the latent space uncertainty.

1. Introduction

Structured output prediction methods [2, 19] have gained popularity in computer vision due to their ability to provide an elegant formulation for systems that perform various important visual tasks such as object detection [4] or semantic segmentation [13]. In the supervised setting these methods assume that training data is fully labeled. However in many computer vision tasks it can be very expensive, or even impossible, to gather such fully supervised datasets. For example when performing action recognition, we may know that a person is performing an action in the image. However, the exact location of the person may not be known as it is more expensive to obtain bounding box annotations compared to image-level annotations.

In order to learn from weakly supervised datasets (that is, datasets whose samples contain missing information in the annotation), a popular approach is to use the latent structural support vector machine (LSSVM) framework [10, 28]. The LSSVM framework models the missing information of weakly supervised datasets with latent/hidden variables. Its prediction criterion is the maximization of the joint posterior probability over the output and hidden variables. Its learning objective is an upper bound on a user-specified loss function that provides a measure of the prediction risk. Recently, several generalizations of the LSSVM framework have been proposed. While the generalized frameworks share the common characteristic that their parameters are estimated by minimizing the prediction risk, they differ from each other in the prediction criterion. Specifically, the methods can be separated into two categories. The first category predicts the output while marginalizing over the latent variable or setting it to its most likely value, and the second performs prediction while minimizing the uncertainty over the latent space with the use of an entropy-based uncertainty measure.

In this paper, our goal is to study the LSSVM framework and its generalizations in order to aid their application in computer vision. We propose a natural unified framework that contains all previous loss-based latent structured prediction frameworks as special cases. The UF performs inference of the output variable by minimizing the uncertainty over the hidden variable as measured by the Aczél and Daròczy (AD) entropy. We derive an optimization algorithm based on the concave convex procedure (CCCP) presented in [29] for learning the parameters of the UF. This allows us to compare the prediction criterion of all frameworks presented in Section 2 by discounting any variability that may arise due to differences in the optimization algorithm. We tested the UF, LSSVM, and

LSSVM's generalizations on the task of multiple gesture recognition from video sequences and action recognition in images. Our experiments convincingly demonstrate that, for large and ambiguous latent spaces, the entropy-based prediction criterion provides more accurate results.

2. Related Work

Two types of methods have been proposed to perform structured output prediction in the presence of latent variables. With probabilistic methods the parameters are estimated by maximizing the (incomplete) data log-likelihood. One famous example is the Expectation-Maximization (EM) algorithm [6, 24]. EM and its variants [12, 18, 22] have been widely used to learn the parameters of a model with latent variables.

In this paper, we are primarily interested in the second type of methods where the parameters are learned by minimizing the regularized empirical risk. The risk is measured by a user-specified loss function. The most commonly used example of a loss-based framework is the latent structural support vector machine (LSSVM) [10, 28]. LSSVM extends structural support vector machines (SSVM) for structured output prediction [25, 26] to latent variable models. The LSSVM model performs prediction by maximizing the joint posterior probability over the output and hidden variables. The parameters of the model are learned by minimizing an upper bound on the user-defined loss. Due to its simplicity and computational efficiency, the LSSVM formulation has gained popularity among the computer vision community with several applications such as object detection [10, 32], image segmentation [13], indoor scene interpretation [27] and action classification [3]. Indeed, its formulation enables LSSVM to be optimized with energy minimization techniques such as graph cuts [5]. However, this advantage is not reflected in our experiments as we choose application problems where all output configurations are computable in a computationally feasible manner, in order to remove from our comparison the possible effects of the use of approximation methods.

Recent years have witnessed the development of several generalizations of LSSVM. While all these frameworks estimate the parameters by minimizing an upper bound on the empirical risk, they differ greatly in their prediction criterion and can be divided into two categories. The first category uses marginalization of the latent variables. Schwing *et al*. [23] introduce a temperature parameter ϵ and propose a family of models, which we will refer to as the ϵ-framework. The prediction criterion of the ϵ-framework can range from the maximization over the output and hidden variables (by setting ϵ to the limit value of 0, noted $\epsilon \searrow 0$, recovering the LSSVM model) to the marginalization over these variables (by setting $\epsilon = 1$). Ping *et al*. [21] introduce marginal structured support vector machines (MSSVM). The MSSVM prediction criterion involves marginalizing the latent variables to estimate the probability of output variables. They also describe a more general framework with temperature parameters that includes LSSVM [10, 28], MSSVM [21] and the ϵ-framework [23] as special cases. The second category uses entropy as an uncertainty measure on the value of the hidden variable. Specifically, Miller *et al*. [17] proposed the max-margin min-entropy (M3E) family of models. M3E models account for uncertainty over the hidden variable by predicting the output with minimal Rényi entropy.

Unlike LSSVM, its generalizations mentioned above have not been widely used by the computer vision community. We believe that the reason for this is three-fold. First, there have been limited experimental results reported in the literature that compare the merits of each of these methods. Second, even in the limited experiments, it is difficult to assess whether the reported improvements of one method over the other are due to a better learning objective or a better optimization algorithm. Third, each method has in the best case a completely different software to the other, and in the worst case no publicly available implementation. This prevents a user, who is primarily interested in exploiting the advances made in machine learning to improve the state of the art in computer vision, to use the more sophisticated learning formulations.

Our goal is to study these generalizations and to define a simple prediction criterion that allows us to construct a Unified Framework (UF) for loss-based latent structured prediction. Our UF recovers all the aforementioned frameworks, that is, LSSVM [10, 28], MSSVM [21], the ϵ-framework [23] and M3E [17] as special cases. Furthermore, it also exposes an extra-degree of freedom that has not yet been explored. We develop an optimization procedure and propose a single simple and efficient algorithm for learning the parameters of the UF. With this setting we are able to compare the prediction criteria of the cited models by discounting the differences that could come from using different optimization algorithms. We built a software implementing the UF and the loss-based models we cited. Our software, as well as all the data required to replicate our experiments, will be made publicly available at the authors homepage.

Our UF generalizes formulations that obtain a point estimate of the model parameters. However a notable framework that is not covered in our generalization is the partially observed Maximum Entropy Discrimination

Markov Network (PoMEN) [31]. The PoMEN model builds on the maximum entropy discrimination Markov Networks (MaxEnDNet) [30], which subsumes SSVM. It combines Bayesian techniques and max-margin learning and its prediction criterion is based on model averaging which results in an advantageous smoothing effect. The conclusions drawn from the current paper lend support to the investigation of entropy-based generalizations of PoMEN.

3. Preliminaries

In this section we provide notations and definitions used in the rest of the paper.

Notations. For simplicity, we assume a discrete setting. We denote the input by $\boldsymbol{x} \in \mathcal{X}$, the output by $\boldsymbol{y} \in \mathcal{Y}$ and the hidden variables by $\boldsymbol{h} \in \mathcal{H}$. The value of $\boldsymbol{x}$ is known during both training and testing, the value of $\boldsymbol{y}$ is known only during training and the value of $\boldsymbol{h}$ is unknown during both training and testing. For example when performing multi-class gesture recognition in video sequences, the input $\boldsymbol{x}$ consists of joint coordinates of a person performing a gesture in the sequence. The output $\boldsymbol{y}$ is the gesture class label of the video. In a video sequence, only a small proportion of frames effectively represent the human performing the gesture. It is easy to collect a large number of such video sequences. However it would be time consuming to annotate each frame as containing the gesture or not. Thus we will only consider the label of the gesture performed in a video sequence but not the annotation at the frames level. In other words, we are in the setting of weakly supervised learning. The latent variable $\boldsymbol{h}$ is introduced to take into account this lack of information.

We assume $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$ follows a conditional model:

$$P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w}) \propto \exp\big(\frac{1}{\epsilon_h}\boldsymbol{w}^T\phi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})\big), \tag{1}$$

where $\phi : (\mathcal{X}, \mathcal{Y}, \mathcal{H}) \to \mathbb{R}^D$ refers to the joint feature vector of the input, the output and the hidden variables, and $\boldsymbol{w} \in \mathbb{R}^D$ are the model parameters.

We introduce the temperature parameter $\epsilon_h \in [0, 1]$ in the expression of the joint probability of the output and the hidden variables. The use of ϵ_h allows our prediction criterion to range from assigning the latent variable to its most likely value when $\epsilon_h \searrow 0$ to the marginalization over the hidden variables when $\epsilon_h = 1$. It follows that

$$P(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{w}) \propto \sum_{\boldsymbol{h}} \exp\big(\frac{1}{\epsilon_h}\boldsymbol{w}^T\phi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})\big). \tag{2}$$

To describe the joint conditional probability distribution of the output and hidden variables $(\boldsymbol{y}, \boldsymbol{h}) \in \mathcal{Y} \times \mathcal{H}$ given an input $\boldsymbol{x} \in \mathcal{X}$ and model parameters $\boldsymbol{w} \in \mathbb{R}^D$, $P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})$, we use the shorthand notation $P_{\boldsymbol{x}}$. For a fixed output $\boldsymbol{y} \in \mathcal{Y}$, $P_{\boldsymbol{x}}$ is a probability distribution of the hidden variable $\boldsymbol{h} \in \mathcal{H}$ which we denote as $Q_{\boldsymbol{x}}^{\boldsymbol{y}}$. Note that $Q_{\boldsymbol{x}}^{\boldsymbol{y}}$ is a generalized distribution that need not sum to one. Finally, we denote the conditional probability distribution of the hidden variables given the input, the output and the model parameters, $P(\boldsymbol{h}|\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w})$, as $P_{\boldsymbol{x}}^{\boldsymbol{y}}$.

Aczél and Daròczy entropy. To measure uncertainty we will use the Aczél and Daròczy (AD) entropy (AD) [1], parametrized by the scalars α and β. Formally, the AD entropy of the generalized distribution is

$$H_{\alpha,\beta}(Q_{\boldsymbol{x}}^{\boldsymbol{y}}; \boldsymbol{w}) = \frac{1}{1-\alpha}\log\Big(\frac{\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})^{\alpha+\beta-1}}{\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})^{\beta}}\Big), \tag{3}$$

with $\alpha \geq 0, \alpha \neq 1, \beta \geq 0, \alpha + \beta - 1 \geq 0$.

The AD entropy has been studied by Aczél and Daròczy [1] as a natural generalization of the Rényi entropy with an extra scalar parameter β. The AD entropy has since been shown to create a natural family of uncertainty measures, also recovering other existing entropy functions as special cases [7, 16].

Let us take a closer look at some interesting special cases of the AD entropy. Specifically when $\beta = 1$, the AD entropy is equivalent to the Rényi entropy used in the M3E models [17],

$$H_{\alpha,1}(Q_{\boldsymbol{x}}^{\boldsymbol{y}}; \boldsymbol{w}) = \frac{1}{1-\alpha}\log\Big(\frac{\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})^{\alpha}}{\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})}\Big). \tag{4}$$

When $\beta = 1$ and $\alpha \to \infty$, the AD entropy is equivalent to the minimum entropy

$$H_{\infty,1}(Q_{\boldsymbol{x}}^{\boldsymbol{y}}; \boldsymbol{w}) = -\log\max_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w}). \tag{5}$$

In other words, the AD entropy for a generalized distribution corresponding to $\boldsymbol{y}$ is obtained by maximizing the joint probability of $\boldsymbol{y}$ and $\boldsymbol{h}$ over the latent variables $\boldsymbol{h}$. When $\beta = 1$ and $\alpha \to 1$, the AD entropy is equivalent to the commonly used Shannon entropy

$$H_{1,1}(Q_{\boldsymbol{x}}^{\boldsymbol{y}}; \boldsymbol{w}) = -\frac{\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})\log P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})}{\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w})}. \tag{6}$$

When $\beta = 0$ and $\alpha = 2$, the AD entropy is equivalent to the marginalization over the latent variable:

$$H_{2,0}(Q_{\boldsymbol{x}}^{\boldsymbol{y}}; \boldsymbol{w}) = -\log\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}; \boldsymbol{w}) + K, \tag{7}$$

where K is a constant. In other words, the AD entropy for a generalized distribution corresponding to $\boldsymbol{y}$ is obtained by marginalizing the joint probability of $\boldsymbol{y}$ and $\boldsymbol{h}$ over the latent variables $\boldsymbol{h}$.

4. The Unified Framework

In order to facilitate our exploration of the performance of the various generalizations of LSSVM, we would like to obtain a unified criterion for prediction that captures the previously used criteria. In other words, our criterion should include as special cases, (i) the maximization over the output $\boldsymbol{y}$ and the latent variable $\boldsymbol{h}$; (ii) the maximization over the output $\boldsymbol{y}$ after marginalizing the joint probability over $\boldsymbol{h}$; and (iii) the maximization over the output $\boldsymbol{y}$ while minimizing the uncertainty over the distribution of $\boldsymbol{h}$. Furthermore, we would also like to design a learning objective that minimizes an upper bound on the empirical risk based on the prediction criterion, where the risk is measured by a user-defined loss function. We begin by showing that the AD entropy provides a suitable prediction criterion that meets the aforementioned requirements. The identification of the AD entropy as our prediction criterion will allow us to develop a suitable learning objective.

4.1. Prediction

We propose to perform prediction for an input $\boldsymbol{x}_i$ by minimizing the AD entropy of the generalized distribution over all possible outputs, that is

$$\boldsymbol{y}_i(\boldsymbol{w}) = \operatorname*{argmin}_{\boldsymbol{y}} H_{\alpha,\beta}(Q^{\boldsymbol{y}}_{\boldsymbol{x}_i}; \boldsymbol{w}). \tag{8}$$

Again, let's look at special cases of this prediction procedure. When $\beta = 1$, the prediction recovers the prediction task of M3E models [17],

$$\boldsymbol{y}_i(\boldsymbol{w}) = \operatorname*{argmin}_{\boldsymbol{y}} H_{\alpha,1}(Q^{\boldsymbol{y}}_{\boldsymbol{x}_i}; \boldsymbol{w}). \tag{9}$$

Since M3E itself generalizes LSSVM, it follows that our prediction criterion also includes it as a special case. Specifically, $\beta = 1$ and $\alpha \to \infty$, the prediction recovers the prediction task of LSSVM [10, 28] by performing maximum a posteriori prediction,

$$\boldsymbol{y}_i(\boldsymbol{w}) = \operatorname*{argmax}_{\boldsymbol{y}} \max_{\boldsymbol{h}} \log P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}_i; \boldsymbol{w}). \tag{10}$$

Similarly when $\beta = 0$ and $\alpha = 2$, the prediction task is equivalent to the maximization of the marginalized joint probability over the output and hidden variable. This is the prediction setting of MSSVM [21] and ϵ-framework [23],

$$\boldsymbol{y}_i(\boldsymbol{w}) = \operatorname*{argmax}_{\boldsymbol{y}} \log \sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}_i; \boldsymbol{w}). \tag{11}$$

The following proposition sheds further light on our prediction criterion.

Proposition 1. *The AD entropy of the generalized distribution of $\boldsymbol{y}$ can be written as the sum of the negative log-likelihood of $\boldsymbol{y}$ and the AD entropy of the conditional distribution of the hidden variable given the output,*

$$H_{\alpha,\beta}(Q^{\boldsymbol{y}}_{\boldsymbol{x}}; \boldsymbol{w}) = -\log P(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}) + H_{\alpha,\beta}(P^{\boldsymbol{y}}_{\boldsymbol{x}}; \boldsymbol{w}). \tag{12}$$

Proof. In the supplementary material. □

Proposition 1 shows that performing prediction by minimizing the AD entropy is equivalent to predicting the output $\boldsymbol{y}$ which (i) has a high probability, and (ii) minimizes the uncertainty over the hidden variable $\boldsymbol{h}$. Specifically when $\beta = 0$ and $\alpha = 2$, the term $H_{\alpha,\beta}(P^{\boldsymbol{y}}_{\boldsymbol{x}}; \boldsymbol{w})$ in (12) disappears. That means that we do not minimize the uncertainty over the distribution of $\boldsymbol{h}$ and (8) and (11) are equivalent. This is another way to see how we recover the prediction procedure of the MSSVM and ϵ-framework models.

4.2. Learning

Given a training dataset of input-output pairs $D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i), i = 1...N\}$, we wish to learn the parameters of the model, described by the weight vector $\boldsymbol{w}$, to be able to predict the output for any input $\boldsymbol{x}$. We introduce the loss function $\Delta(\boldsymbol{y}, \boldsymbol{y}_i)$ with $\Delta(\boldsymbol{y}, \boldsymbol{y}) = 0$ that compares the risk of making the prediction $\boldsymbol{y}$ for the input $\boldsymbol{x}_i$ with ground truth output $\boldsymbol{y}_i$.

The parameters of the model are learned by minimizing the following objective function (13).

$$\min_{\boldsymbol{w}} \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{n}\sum_i \Big[\epsilon_y \log \sum_{\boldsymbol{y}} \exp \frac{1}{\epsilon_y}\Big(\Delta(\boldsymbol{y}_i, \boldsymbol{y}) - \epsilon_h H_{\alpha,\beta}(Q^{\boldsymbol{y}}_{\boldsymbol{x}_i}; \boldsymbol{w})\Big) + \epsilon_h H_{\alpha,\beta}(Q^{\boldsymbol{y}_i}_{\boldsymbol{x}_i}; \boldsymbol{w})\Big]. \tag{13}$$

We introduce regularization over the parameters of the model $\boldsymbol{w}$ to avoid overfitting the parameters to the training data. Furthermore, we introduce the temperature parameter $\epsilon_y \in [0, 1]$. When $\epsilon_y \searrow 0$ minimizing objective (13) results in maximizing the margin between the AD entropy of the ground truth value of the output and all other values of the output. For other values of ϵ_y, the objective function replaces the maximization by a soft max (log-sum-exp) function.

Proposition 2 shows that the optimization procedure of (13) minimizes an upper bound on the user-defined loss.

Proposition 2. *Objective* (13) *minimizes an upper bound on the loss $\Delta(\boldsymbol{y}_i, \boldsymbol{y}_i(\boldsymbol{w}))$ where $\boldsymbol{y}_i$ is the ground truth output of training example i and $\boldsymbol{y}_i(\boldsymbol{w})$ is the predicted output. This upper-bound is tightest when $\epsilon_y \searrow 0$.*

Proof. In the supplementary material. □

Our objective function (13) naturally derives from our prediction procedure and allow us to upper bound the user-defined loss. Furthermore we show by looking at special cases that it recovers the objective functions of the models we want to unify. Specifically when $\beta = 1, \epsilon_y \searrow 0, \epsilon_h = 1$ we retrieve the objective function used in training of the

M3E models [17], which maximizes the margin between the entropy of the ground truth output value and all other outputs:

$$\min_{\boldsymbol{w}} \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{n}\sum_i \max_{\boldsymbol{y}} \Big(\Delta(\boldsymbol{y}_i, \boldsymbol{y}) - H_{\alpha,1}(Q^{\boldsymbol{y}}_{\boldsymbol{x}_i}; \boldsymbol{w}) + H_{\alpha,1}(Q^{\boldsymbol{y}_i}_{\boldsymbol{x}_i}; \boldsymbol{w})\Big). \quad (14)$$

From the M3E models, taking $\alpha \to \infty$ recovers LSSVM [10, 28]. Thus when $\beta = 1, \epsilon_y, \searrow 0, \epsilon_h = 1, \alpha \to \infty$ the learning objective is equivalent to the learning problem of the LSSVM:

$$\min_{\boldsymbol{w}} \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{n}\sum_i \max_{\boldsymbol{y}\times\boldsymbol{h}} \Big(\Delta(\boldsymbol{y}_i, \boldsymbol{y}) + \boldsymbol{w}^T\phi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h})\Big) - \frac{C}{n}\sum_i \max_{\boldsymbol{h}} \boldsymbol{w}^T\phi(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{h}). \quad (15)$$

When $\beta = 0, \alpha = 2, \epsilon_y, \searrow 0, \epsilon_h = 1$, the log-sum-exp function over $\boldsymbol{y}$ approximates the max function and we end up maximizing the output $\boldsymbol{y}$ while marginalizing the hidden variable $\boldsymbol{h}$. Thus (13) becomes equivalent to the optimization problem of the MSSVM [21]:

$$\min_{\boldsymbol{w}} \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{n}\sum_i \max_{\boldsymbol{y}} \Big(\Delta(\boldsymbol{y}_i, \boldsymbol{y}) + \log\sum_{\boldsymbol{h}} \exp(\boldsymbol{w}^T\phi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}))\Big) - \frac{C}{n}\sum_i \log\sum_{\boldsymbol{h}} \exp\big(\boldsymbol{w}^T\phi(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{h})\Big). \quad (16)$$

Similarly when $\beta = 0, \alpha = 2, \epsilon_y = \epsilon_h$ we recover the same objective to minimize as in the ϵ-framework [23], that is,

$$\min_{\boldsymbol{w}} \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{n}\sum_i \epsilon \log\sum_{\boldsymbol{y}} \exp\Big(\frac{\Delta(\boldsymbol{y}_i, \boldsymbol{y}) + \boldsymbol{w}^T\phi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h})}{\epsilon}\Big) - \frac{C}{n}\sum_i \epsilon \log\sum_{\boldsymbol{h}} \exp\Big(\frac{\boldsymbol{w}^T\phi(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{h})}{\epsilon}\Big). \quad (17)$$

Figure 1 shows that our framework gathers all models presented in Section 2 with specific parameters values. We refer the reader to the supplementary material for more details.

5. Optimization method

We propose to use a common algorithm for all methods covered by the UF, which is computationally efficient. This allows us to compare the prediction criterion of all these models independently of their specific methods for solving their optimization problem. We derive an optimization procedure for learning the parameters of the UF. This procedure works for every setting of parameters of the UF. In other words it works for every model presented in Section 2.

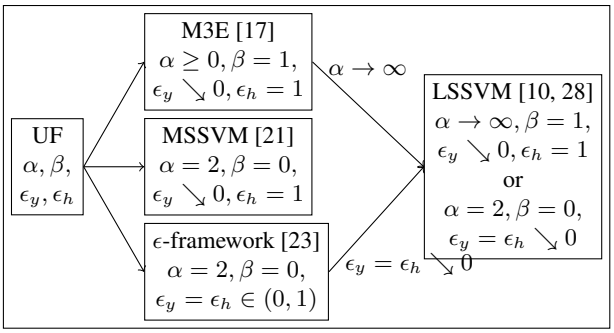


Figure 1: *Equivalence of UF with existing models.*

The optimization function of problem (13) is not convex. In order to obtain an approximate solution, we write the objective function of (13) as a difference of convex (DC) functions [29]. To this end, it would be helpful to introduce the following shorthand notations in the case $\alpha > 1$:

- $F^+_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w}) = -\epsilon_h \frac{1}{1-\alpha} \log\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}_i; \boldsymbol{w})^{\alpha+\beta-1}$,
- $F^-_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w}) = -\epsilon_h \frac{1}{1-\alpha} \log\sum_{\boldsymbol{h}} P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x}_i; \boldsymbol{w})^{\beta}$,
- $G^+_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w}) = -\epsilon_h \frac{1}{1-\alpha} \log\sum_{\boldsymbol{h}} P(\boldsymbol{y}_i, \boldsymbol{h}|\boldsymbol{x}_i, \boldsymbol{w})^{\beta}$,
- $G^-_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w}) = -\epsilon_h \frac{1}{1-\alpha} \log\sum_{\boldsymbol{h}} P(\boldsymbol{y}_i, \boldsymbol{h}|\boldsymbol{x}_i, \boldsymbol{w})^{\alpha+\beta-1}$.

The functions $F^+_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w}), F^-_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w}), G^+_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w})$, $G^-_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w})$ are convex with respect to $\boldsymbol{w}$. Their definitions are trivially inferred in the case $\alpha < 1$:

Proposition 3. *The optimization problem* (13) *can be equivalently written as a difference of convex (DC) functions for any values of* $\alpha \geq 0, \beta \geq 0$ *using the following formulation:*

$$\min_{\boldsymbol{w}} \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{n}\epsilon_y \sum_i \Big[\log\sum_{\boldsymbol{y}} \exp\frac{1}{\epsilon_y}\Big(\Delta(\boldsymbol{y}_i, \boldsymbol{y}) + F^+_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w}) - F^-_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w})\Big) + G^+_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w}) - G^-_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w})\Big], \quad (18)$$

where $F^+_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w})$, $F^-_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w})$, $G^+_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w})$, *and* $G^-_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w})$ *are convex.*

Proof. In the supplementary material. □

This results in Algorithm 1 for training the UF. Algorithm 1 is similar to standard concave convex procedure (CCCP) [29]. During step 1 of Algorithm 1, we solve the convex optimization problem (19):

$$\boldsymbol{w}_{t+1} = \operatorname*{argmin}_{\boldsymbol{w}} \ \frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{n}\epsilon_y \sum_i \Big[\log\sum_{\boldsymbol{y}} \exp\frac{1}{\epsilon_y}\Big(\Delta(\boldsymbol{y}_i, \boldsymbol{y}) + F^+_{\alpha,\beta}(\boldsymbol{y}, \boldsymbol{w}) - T^{F^-_{\alpha,\beta}}_{\boldsymbol{y},\boldsymbol{w}_t}(\boldsymbol{w})\Big) + G^+_{\alpha,\beta}(\boldsymbol{y}_i, \boldsymbol{w}) - T^{G^-_{\alpha,\beta}}_{\boldsymbol{y}_i,\boldsymbol{w}_t}(\boldsymbol{w})\Big]. \quad (19)$$

During step 1, $F^-_{\alpha,\beta}(\boldsymbol{y},\boldsymbol{w})$ and $G^-_{\alpha,\beta}(\boldsymbol{y}_i,\boldsymbol{w})$ are replaced by their first order Taylor expansion:

$$T^{F^-_{\alpha,\beta}}_{\boldsymbol{y},\boldsymbol{w}_t}(\boldsymbol{w}) = F_{\alpha,\beta}(\boldsymbol{y},\boldsymbol{w}_t)^- + (\boldsymbol{w}-\boldsymbol{w}_t)^T\nabla_{\boldsymbol{w}}F^-_{\alpha,\beta}(\boldsymbol{y},\boldsymbol{w})|_{\boldsymbol{w}_t},$$
$$T^{G^-_{\alpha,\beta}}_{\boldsymbol{y}_i,\boldsymbol{w}_t}(\boldsymbol{w}) = G_{\alpha,\beta}(\boldsymbol{y}_i,\boldsymbol{w}_t)^- + (\boldsymbol{w}-\boldsymbol{w}_t)^T\nabla_{\boldsymbol{w}}G^-_{\alpha,\beta}(\boldsymbol{y}_i,\boldsymbol{w})|_{\boldsymbol{w}_t}. \tag{20}$$

We denote by $\nabla_{\boldsymbol{w}}F^-_{\alpha,\beta}(\boldsymbol{y},\boldsymbol{w})|_{\boldsymbol{w}_t}$ the gradient of $F^-_{\alpha,\beta}(\boldsymbol{y},\boldsymbol{w})$ with respect to $\boldsymbol{w}$ estimated at $\boldsymbol{w}_t$ and similarly $\nabla_{\boldsymbol{w}}G^-_{\alpha,\beta}(\boldsymbol{y}_i,\boldsymbol{w})|_{\boldsymbol{w}_t}$.

Algorithm 1: Algorithm for trainin UF

Data: $D = \{(\boldsymbol{x}_i,\boldsymbol{y}_i), i = 1...N\}$
Result: Model parameter $\boldsymbol{w}$
initialize $\boldsymbol{w} = \boldsymbol{w_0}, t = 0$;

$$\begin{aligned}\text{obj}(\boldsymbol{w},\boldsymbol{w}_t) = \frac{1}{2}||\boldsymbol{w}||^2 \\ + \frac{C}{n}\epsilon_y\sum_i\Big[\log\sum_{\boldsymbol{y}}\exp\frac{1}{\epsilon_y}\Big(\Delta(\boldsymbol{y}_i,\boldsymbol{y}) \\ + F^+_{\alpha,\beta}(\boldsymbol{y},\boldsymbol{w}) - T^{F^-_{\alpha,\beta}}_{\boldsymbol{y},\boldsymbol{w}_t}(\boldsymbol{w})\Big) \\ + G^+_{\alpha,\beta}(\boldsymbol{y}_i,\boldsymbol{w}) - T^{G^-_{\alpha,\beta}}_{\boldsymbol{y}_i,\boldsymbol{w}_t}(\boldsymbol{w})\Big]\end{aligned} \tag{21}$$

while $t \le T$ *and* $\delta_{obj} \ge C\lambda$ **do**
1 $\quad\boldsymbol{w}_{t+1} \leftarrow \operatorname{argmin}_{\boldsymbol{w}} \text{obj}(\boldsymbol{w},\boldsymbol{w}_t)$ by gradient descent.
2 $\quad\delta_{obj} \leftarrow \text{obj}(\boldsymbol{w}_t,\boldsymbol{w}_{t-1}) - \text{obj}(\boldsymbol{w}_{t+1},\boldsymbol{w}_t)$
3 $\quad t \leftarrow t+1$
4 **end**
5 return $\boldsymbol{w}$

We solve the optimization problem (19) of step 1 in Algorithm 1 by performing gradient descent with a step size found by line search. We refer the reader to the supplementary material for details on our gradient descent procedure.

6. Experiments

We performed experiments to compare the UF, LSSVM [10, 28], MSSVM [21], the ϵ-framework [23], M3E models [17] and replications of these models by the UF. Our goal with these experiments is to assess which is the most accurate of prediction criteria between the two types of models generalizing LSSVM (described in Section 2), regardless of their specific optimization algorithm. To this end, we compare them and their replications using the UF on two different tasks. In all figures the sign $\sim$ means that the UF's parameters were set to replicate an existing model.

6.1. Binary action classification

We start our empirical comparison on an example where the latent space size is small and the uncertainty over the hidden variable is low. Specifically, we perform the following experiment of binary action classification over the 10 classes of the PASCAL VOC 2011 dataset.

Pascal VOC dataset. We use the "trainval" dataset of the PASCAL VOC 2011 [8] action classification dataset, consisting of 2424 images depicting 10 action classes. We used the 10 classes of the dataset. For each image we are provided the bounding boxes of the persons in the image and its action class. However, in our experiments, we discard the bounding box information and instead model it using a latent variable.

Modelling and features. The score of a bounding box in the image $\boldsymbol{x}$ is $\boldsymbol{w}^T\phi(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h}) = \boldsymbol{w}^T_{\boldsymbol{y}}\phi(\boldsymbol{x},\boldsymbol{h})$ where $\boldsymbol{w}_{\boldsymbol{y}}$ are the parameters that correspond to the label $\boldsymbol{y}$ and $\phi(\boldsymbol{x},\boldsymbol{h})$ is the feature vector extracted from the bounding box $\boldsymbol{h}$. Similar to [3], we consider the bounding boxes of the image with the top 20 scores found by a standard person detector [9]. Thus we reduce the uncertainty on the latent space since we take only the top scoring bounding boxes and we work with a small latent space size. The feature vector $\phi(\boldsymbol{x},\boldsymbol{h})$ consists of the standard poselet-based feature vector [15], that is a 2404 dimensional vector consisting of 2400 activation scores of action specific poselets and 4 object activation scores. We added the score given by the person detector [9] making $\phi(\boldsymbol{x},\boldsymbol{h})$ a 2405 dimensional feature vector. The loss Δ is the standard 0-1 loss.

Experimental setting. We split the dataset into 1940 training images and 484 validation images. Hyper parameters are chosen via 5 fold cross-validation. We use four random seed values in order to mitigate possible effects of the initialization; this is to ensure that the non-convexity of the optimization objective does not lead to poor results by local minima. For each fold, we report the test error corresponding to the seed with the lowest training objective value. All models are initialized with LSSVM [10, 28] except for M3E with $\alpha \to \infty$ that includes random initialization of the hidden variable value. The convergence tolerance is set to $\lambda = 0.01$. For each model, we tested the following range of parameters: $C = [0.1, 1, 10, 100, 1000]$, $\epsilon = [0.001, 0.01, 0.1, 1]$ (referring to the parameter of the ϵ-framework), $\epsilon_h = [0.001, 0.01, 0.1, 1.0]$, $\alpha = [0.01, 0.1, 2, 100, 1000, 10000]$ and $\beta = [0.0, 0.5, 1.0]$. In terms of computing CPU times, all models require comparable times. We refer the reader to the supplementary material for detailed timing.

Results. Figure 2 shows the mean loss on the test set over the 10 classes of each model with best cross-validated parameters (averaged over the 5 folds). We refer the reader to the supplementary material for additional details regarding the experimental setup and the results. From figure 2, we see that the performances of UF as a replication and the corresponding existing model are similar. We see that

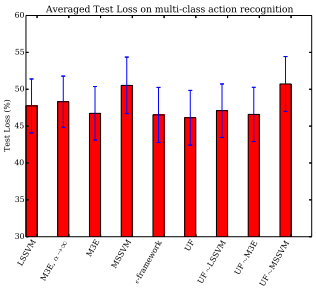


Figure 2: *Test loss mean on the 10 classes (in %) ± standard error of the mean (in %) with cross-validated parameters on the PASCAL VOC 11 dataset, after averaging on the 5 folds.*

predicting the output by marginalizing the output and hidden variables as done by MSSVM is the less accurate criterion. Our UF and M3E models provide slightly better performances than LSSVM, MSSVM and the ϵ-framework. For clarity purposes we do not report the best parameters chosen by cross-validation but they are available in the supplementary material. In most cases the set of UF parameters chosen by cross-validation boils down to a prediction criterion that maximizes over the output and hidden variables. Similarly, the best α chosen for the M3E models is of high value. In other words, M3E recovers LSSVM. The ϵ-framework chooses a small value of ϵ, which also approximates LSSVM. Thus for this specific task there is a small gain to be made from taking into account the uncertainty over the hidden variable. This comes at the cost of using a more complicated model with more parameters to cross-validate. This result is aligned with the fact that the size of the latent space is small (we use 20 bounding boxes) and that uncertainty over the hidden variable is reduced since we consider top scoring bounding boxes.

6.2. Multi-class gesture recognition

We now explore an experiment where the uncertainty over the value of the hidden variable is high and the latent space is large. To do so, we tested the UF and the loss-based models presented in Section 2 on the task of gesture recognition in video sequences. Given a set of video sequences, our goal is to learn to classify, among c possible classes of gestures, the gesture performed in a video sequence.

MSRC-12. The MSRC-12 data set [11] contains 594 sequences of motion capture data, recording the 3D world position of 20 joints in the human body using a Kinect sensor from a population of 30 individuals. In each sequence the actor performs one type of gesture repeatedly, typically 10 times, and each instance of the gesture is marked at a typical frame. The original purpose of the data set was to enable research into low latency gesture detection [20]. However, it has since been used to classify the sequence as a whole [11, 14]. We also perform sequence classification but do not use the individual frame-level annotations. Instead, as training data we use only the sequence class label. This is a realistic assumption for example in an interactive setting when the user is instructed by a system to perform a gesture for an initial training phase. In this case we only know that a gesture of a given class will be performed but do not know when the user will perform it.

Noisy MSRC-12 dataset. In order to evaluate the behavior of the different models in presence of noisy data, we corrupt the MSRC-12 dataset by adding random Gaussian noise with zero mean and various standard deviations. In particular we add noise to each elements of all frames' feature vectors. We use three values of standard deviations: $\sigma = [1\text{cm}, 5\text{cm}, 8\text{cm}]$.

Modelling and features. The input data $\boldsymbol{x}$ consists of 3D world position of a person performing a gesture. The output $\boldsymbol{y}$ is the class of the gesture performed in the video sequence. In a video sequence only a few percent of the video frames effectively contain the person performing the gesture. We consider the training data as weakly labeled, i.e. the video sequences are only labeled at the sequence level and each frame is not individually annotated as containing the gesture or not. We use the latent variable $\boldsymbol{h}$ to model this lack of information. The size of the latent space is the number of frames per video sequence, that is on average 1200 frames per sequence. The score of a specific frame in a video sequence $\boldsymbol{x}$ is $\boldsymbol{w}^T\phi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) = \boldsymbol{w}_{\boldsymbol{y}}^T\phi(\boldsymbol{x}, \boldsymbol{h})$ where $\boldsymbol{w}_{\boldsymbol{y}}$ are the parameters that correspond to the label $\boldsymbol{y}$ and $\phi(\boldsymbol{x}, \boldsymbol{h})$ is the feature vector extracted from the video sequence frame $\boldsymbol{h}$. We take the same features derived from 3D joint locations as in [11, 14, 20], obtaining a feature vector $\phi(\boldsymbol{x}, \boldsymbol{h})$ of dimension 130. The loss Δ is the standard 0-1 loss.

Experimental setting. We use the 594 sequences from the MSRC-12 dataset. We divide this dataset into five folds, each fold containing 20% of the trainval data set for testing and 80% for training, using stratified sampling over class labels in order to ensure a uniform distribution over classes. As in the action classification experiment of Section 6.1, hyper parameters are cross-validated over 5 folds and we use 4 random seeds for initialization and for each fold, we report the test error corresponding to the seed with the lowest training objective value. All models are initialized with LSSVM [10, 28] except for M3E with $\alpha \to \infty$. The convergence tolerance is set to $\lambda = 0.01$. We tested the same range of parameters as in the binary action classification experiment except for the parameter C we used the values $[1, 10, 100, 1000, 10000]$. When looking at CPU times, all models require comparable times. We refer the reader to the supplementary material for detailed timing.

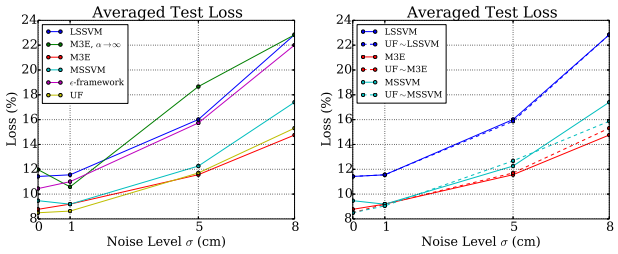
Results. Table 1 reports the best parameters chosen by cross-validation for the UF. Table 1 shows that the best parameters combination $(\epsilon_h, \alpha, \beta)$ always take in account the AD entropy of the hidden variable and the term $H_{\alpha,\beta}(P_{\boldsymbol{x}}^{\boldsymbol{y}}; \boldsymbol{w})$ in (12) is never set to 0. This would be the case if we had $(\alpha = 2, \beta = 0)$ or $(\alpha \to \infty, \beta = 1)$. Thus the UF is never boiling down to either LSSVM,

MSSVM or the ϵ-framework. Recall that UF recovers all models with the use of the same algorithm. This means that regardless of the algorithm used to solve the minimization problem during learning, predicting the output variables by minimizing the uncertainty over the latent space is a more relevant method than marginalizing the latent variables or estimating their most likely values.

σ = 0cm	σ = 1cm	σ = 5cm	σ = 8cm
$\epsilon_h = 1, \alpha = 0.01, \beta = 1$	$\epsilon_h = 1, \alpha = 2, \beta = 0.5$	$\epsilon_h = 1, \alpha = 0.1, \beta = 1$	$\epsilon_h = 1, \alpha = 0.1, \beta = 1$

Table 1: *Cross-validated parameters for the UF.*

Figure 3a shows the average loss on the test set for each model with respect to the noise level corrupting the dataset.



(a) *Existing models and the UF.* (b) *Existing models (line curves) and their replication by the UF (dashed curves).*

Figure 3: *Test loss (%) on the MSRC-12 dataset averaged on 5 folds, per noise level. Each model is shown with best cross-validated parameters.*

Figure 3b shows the average loss on the test set for LSSVM, M3E and MSSVM, and their replication with the UF. This figure shows two things. *First*, by looking at the performances of their replication by the UF, we can compare LSSVM, MSSVM and M3E models without taking into account their specific training algorithm. *Second*, this figure also shows how the UF replicates existing models. In the cases when the UF replicates LSSVM and MSSVM results are similar, this was expected since the algorithm we derived for training the UF is similar to the specific algorithm of these models. When the UF replicates M3E, results are also similar even if the optimization procedures are different (the M3E models use a trust-region based algorithm to solve the optimization problem). From these two figures 3a and 3b we conclude that entropy-based models (either M3E or the UF) give significantly better performances at all noise levels. We refer the reader to the supplementary material for p-values and comparison at statistical significance level of 0.05.

Figure 4 shows the scoring of the UF on an example "bowing" gesture of the non-noisy MSRC-12 dataset. A 3D representation of the frame feature vector is added in the upper part of the plot. For the ground-truth label, the

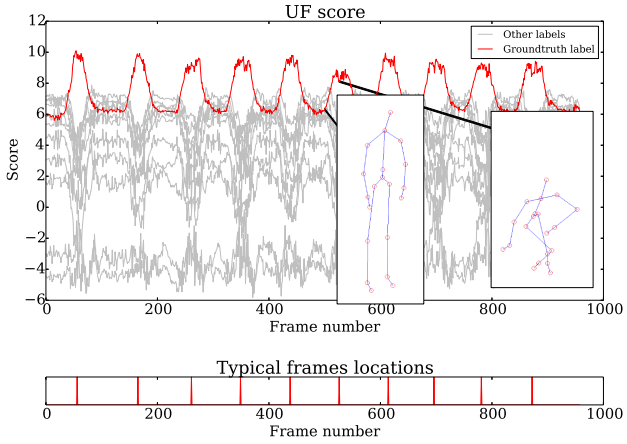


Figure 4: *Scoring of the UF on an example "bowing" gesture of the non noisy MSRC-12 dataset. Upper part of the plot shows the scoring for the ground truth label (in red), and all other labels in grey. Lower part of the plot shows the location of the action frames as per MSRC-12 ground truth (we do not use this frame-level annotation).*

UF gives high score at typical frames of the video sequence when the person is effectively bowing, and a low score in between. The scores of all others labels (shown in grey) are smaller than the smallest score of the ground-truth label frames.

7. Discussion

We developed a Unified Framework (UF) by defining a simple prediction criterion for generalizations of LSSVM. By developing an optimization algorithm for learning the parameters of the UF, we evaluate each prediction criterion without taking into account the differences in performances that could arise from using their specific optimization algorithm. Our experimental results show that the use of the minimization of the latent space uncertainty is an accurate prediction criterion when the size of the latent space is large and when there is uncertainty on the hidden variable. The UF also offers an additional advantage, namely, the use of the extra parameter β. This has yet to be explored thoroughly, and could lead to improved performance. As a direction for future work, we suggest to use a fully Bayesian setting with prior distributions over the UF parameters. We also propose to incorporate the UF in the framework of deep learning to further improve the performance of this popular framework.

8. Acknowlegements

This work is co-funded by the Microsoft Research PhD Scholarship Programme in EMEA (Europe, Middle East, Africa) and Ecole Centrale Paris.

References

[1] J. Aczél and Z. Daróczy. Charakterisierung der Entropien positiver Ordnung und der Shannonschen Entropie. *Acta Mathematica Hungarica, Volume 14, Issue 1*, 1963.

[2] G. H. BakIr, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data*. MIT Press, 2007.

[3] A. Behl, C. V. Jawahar, and M. P. Kumar. Optimizing average precision using weakly supervised data. In *CVPR*, 2014.

[4] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008.

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 23*, 2001.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 1977.

[7] M. D. Esteban and D. Morales. A summary on entropy statistics. *Kybernetika, Volume 31, Nº 4*, 1995.

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision, Volume 88*, 2010.

[9] P. F. Felzenszwalb, R. Girshick, and D. A. McAllester. Discriminatively trained deformable part models, release 4, http://people.cs.uchicago.edu/ pff/latent-release4/.

[10] P. F. Felzenszwalb, D. A. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[11] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin. Instructing people for training gestural interactive systems. In *CHI*, 2012.

[12] A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman and Hall, 1995.

[13] M. P. Kumar, H. Turki, D. Preston, and D. Koller. Learning specific-class segmentation from diverse data. In *ICCV*, 2011.

[14] A. M. Lehrmann, P. V. Gehler, and S. Nowozin. Efficient nonlinear Markov models for human motion. In *CVPR*. IEEE, 2014.

[15] S. Maji, L. D. Bourdev, and J. Malik. Action recognition from a distributed representation of pose and appearance. In *CVPR*, 2011.

[16] A. Mathai and P. Rathie. *Basic Concepts in Information Theory and Statistics: Axiomatic Foundations and Applications*. Wiley, 1975.

[17] K. Miller, M. P. Kumar, B. Packer, D. Goodman, and D. Koller. Max-margin min-entropy models. In *AISTATS*, 2012.

[18] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*. MIT Press, 1999.

[19] S. Nowozin and C. H. Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 2011.

[20] S. Nowozin and J. Shotton. Action points: A representation for low-latency online human action recognition. Technical Report MSR-TR-2012-68, Microsoft Research Cambridge, July 2012.

[21] W. Ping, Q. Liu, and A. Ihler. Marginal structured SVM with hidden variables. In *ICML*, 2014.

[22] J. Salojärvi, K. Puolamäki, and S. Kaski. Expectation maximization algorithms for conditional likelihoods. In *ICML*, 2005.

[23] A. G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Efficient structured prediction with latent variables for general graphical models. In *ICML*, 2012.

[24] R. Sundberg. Maximum likelihood theory for incomplete data from an exponential family. *Scandinavian Journal of Statistics*, 1974.

[25] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *NIPS*, 2003.

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

[27] H. Wang, S. Gould, and D. Koller. Discriminative learning with latent variables for cluttered indoor scene understanding. *Communications of the ACM*, 2013.

[28] C.-N. Yu and T. Joachims. Learning structural SVMs with latent variables. In *ICML*, 2009.

[29] A. L. Yuille and A. Rangarajan. The Concave-Convex Procedure (CCCP). In *NIPS*, 2002.

[30] J. Zhu and E. P. Xing. Maximum entropy discrimination Markov networks. In *JMLR*, 2009.

[31] J. Zhu, E. P. Xing, and B. Zhang. Partially observed maximum entropy discrimination Markov networks. In *NIPS*, 2008.

[32] L. Zhu, Y. Chen, A. L. Yuille, and W. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010.