UDC 004.93'1:519.157

V. Franc, P. Laskov

Learning Maximal Margin Markov Networks via Tractable Convex Optimization

Показано, что обучение марковской сети общего вида может быть представлено в виде задачи выпуклой оптимизации. Основная идея метода заключается в использовании *LP*-релаксации (max,+)-задачи непосредственно при формулировании задачи обучения.

It is shown that the learning of a general Markov network can be represented as a convex optimization problem. The key idea of the method is to use a linear programming relaxation of the (max,+)-problem directly in the formulation of the learning problem.

Показано, що навчання марківської мережі загального вигляду може бути подано у вигляді задачі опуклої оптимізації. Основна ідея методу полягає у використанні *LP*-релаксації (max,+)-задачі безпосередньо при формулюванні задачі навчання.

Abstract

The learning of Markov networks constitutes a challenging optimization problem. Even the predictive step of a general Markov network involves solving an NP-complete max-sum problem. By using the discriminative approach, learning of the Markov networks from noisy examples can be transformed to a convex quadratic program with intractably large number of linear constraints. The intractable quadratic problem can be attacked by an approximate cutting plane (ACP) algorithm proposed in [5]. The ACP requires repeatedly solving a predictive step for finding the most violated constraint. The intractable search for the most violated constraint is approximated by using an (approximate) solver for the max-sum problem. The use of the approximative max-sum solver inside the ACP algorithm brings two important problems. First, the ACP algorithm is not guaranteed to converge to the optimal solution. Second, the max-sum solvers are time consuming algorithms which forbids using the ACP algorithm to large scale problems. In this paper, we show that learning of a general Markov network can be expressed as a convex optimization problem which is solvable efficiently without calling any external max-sum solver. The key idea, generalizing work of [12], is to use a linear programing relaxation of the max-sum problem directly in the formulation of the learning problem. We show that the proposed learning problem can be solved efficiently by the Generalized Proximal Point Algorithm [7]. The empirical evaluation shows that our algorithm speeds up learning of general Markov networks by a factor of up to 20 compared to the ACP algorithm.

1. Introduction

A Markov network is a powerful representation of dependencies in structured objects. A Markov network is defined by an undirected graph consisting of nodes $\mathcal{T}$ and edges $\mathcal{E} \subseteq \binom{\mathcal{T}}{2}$. The nodes correspond to elementary objects while the edges represent possible structural interactions between them. Each object $t \in \mathcal{T}$ is characterized by an observable variable x_t which takes value from a set $\mathcal{X}$ and a hidden variable y_t which takes values from a finite set $\mathcal{Y}$. Dependencies between observed an a hidden variables are modeled for each object t by functions $q_t(x_t, y_t; \theta)$. Dependencies between hidden variables of pairs of objects are modeled for each pair t, t' by functions $g_{tt'}(y_t, y_{t'}; \theta)$. In order to apply a Markov network in practice the dependency functions $q_t(x_t, y_t; \theta)$ and $g_{tt'}(y_t, y_{t'}; \theta)$ have to be specified. In principle, this can be done by hand, but typically one assumes a certain parametrization $\theta \in \mathbb{R}^n$ which controls the functional form of dependency functions q_t and $g_{tt'}$. The parameter θ is then learned from examples of observations and corresponding hidden variables.

The Markov networks are typically used for prediction of the hidden variables $y = (y_t \in \mathcal{Y} \mid t \in \mathcal{T})$ given a tuple of observable variables $x = (x_t \in \mathcal{X} \mid t \in \mathcal{T})$. The predictive step amounts to solving

$$h(x;\theta) = \operatorname*{argmax}_{y \in \mathcal{Y}^{\mathcal{T}}} H(x, y; \theta) = = \operatorname*{argmax}_{y \in \mathcal{Y}^{\mathcal{T}}} \left[\sum_{t \in \mathcal{T}} q_t(x_t, y_t; \theta) + \sum_{tt' \in \mathcal{E}} g_{tt'}(y_t, y_{t'}; \theta) \right], \quad (1)$$

where $h:\mathcal{X}^{\mathcal{T}} \to \mathcal{Y}^{\mathcal{T}}$ denotes the classifier (prediction rule) given by the scoring function $H:\mathcal{X}^{\mathcal{T}} \times \mathcal{Y}^{\mathcal{T}} \to \mathbb{R}$ which measures a match between the observations x and the hidden variables y. The integer programming problem on the right-hand side of (1) is called a *max-sum problem*. We will denote the classification rule (1) as the *max-sum classifier*. The max-sum problem is NP-complete in general but it can be made tractable by imposing additional constraints on the graph $(\mathcal{T},\mathcal{E})$ or the quality functions $g_{tt'}$. The important examples of polynomially solvable max-sum problems involve the problems with *acyclic* graphs, graphs with a low tree-width (also called simple nets), and the problems with *supermodular* functions $g_{tt'}$. For a survey of relevant results we refer to [9, 17].

A probabilistic interpretation of a Markov network can be derived by assuming the observations x and the hidden variables y to be realizations of random variables $X=(X_t \,|\, t\in\mathcal{T})$ and $Y=(Y_t \,|\, t\in\mathcal{T})$ distributed according to the Gibbs distribution $P(x,y;\theta)=\dfrac{1}{Z(\theta)}\exp H(x,y;\theta)$ where $Z(\theta)$ is the partition function. The Maximum A Posteriori Prediction (MAP) of the hidden variables y given observations x is carried out by evaluating $y^* = \mathrm{argmax}_{y\in\mathcal{Y}^{\mathcal{T}}} P(y|x)$ which coincides with the classifier (1).

The parametrization by θ offers two general approaches for learning Markov networks from a training set $\{(x^1,y^1),\ldots,(x^m,y^m)\}\in(\mathcal{X}^{\mathcal{T}}\times\mathcal{Y}^{\mathcal{T}})^m$. A *generative approach* is based on the estimation of parameters θ using a Maximum-Likelihood (ML) principle. Such estimation is well known for Hidden Markov Models with acyclic graphs (see e.g. [11]). In a general case, however, the ML estimation is not tractable for Markov networks because no polynomial time algorithm is known for computing the partition function Z(θ). A *discriminative approach* is an alternative method which is based on direct optimization of parameters θ in order to minimize the prediction error of classifier (1) computed on the training set. The discriminative learning inherently requires solving the intractable predictive step of the max-sum classifier. For this reasons most of the existing methods restrict the class of learned max-sum classifiers in order to make the problem tractable. The restriction is imposed either on the neighborhood graph $(\mathcal{T},\mathcal{E})$ or on the quality functions $g_{tt'}$. In the next paragraph we shortly review existing methods.

The methods for learning max-sum classifiers with acyclic graph $(\mathcal{T},\mathcal{E})$ are well understood [11, 3, 1, 2, 14]. Learning of the Associative Markov Networks (AMN) with an arbitrary graph structure was proposed by [13]. The AMN is the Markov network with the quality functions $g_{tt'}$ restricted in a way similar to the Potts model. Even though the predictive step of the AMN is not tractable, the learning algorithm in [13] alleviates the problem by using a kind of Linear Programming (LP) relaxation. By this manner, the learning problem is transformed into a Quadratic Programming (QP) task with a polynomial number of constraints. Learning of the max-sum classifiers with super-modular quality functions $g_{tt'}$ was proposed in [5]. In this case, the learning leads to a QP task solvable in polynomial time by the Cutting Plane Algorithm (CPA). Another polynomially learnable class is formed by the max-sum classifier with a strictly trivial equivalent (STE) [5]. The max-sum problems with STE are those whose response can be computed exactly via LP relaxation. Provided the training examples are separable, learning of the max-sum classifier with STE can be transformed to solving a system of strict linear inequalities manageable by the Perceptron algorithm. In the case of non-separable examples, the max-sum classifier with STE can be learned by minimizing a regularized upper bound on the empirical risk with zero-one loss function. The only existing approach for learning truly general max-sum classifiers was proposed in [5]. The intractable learning problem is attacked by an Approximate Cutting Plane (ACP) algorithm. The ACP requires repeatedly solving a predictive step for finding the most violated constraint. The intractable search for the most violated constraint is approximated by using an (approximate) solver for the max-sum problem.

The use of the approximative max-sum solver inside the ACP algorithm brings two important problems. First, the ACP algorithm is not guaranteed to converge to the optimal solution. Second, the max-sum solvers are time consuming algorithms which forbids the use of the ACP algorithm to large scale problems.

In this paper, we concentrate on learning of the max-sum classifier with a general neighborhood graph $(\mathcal{T},\mathcal{E})$ and general quality functions $g_{tt'}$. We show that the learning of a general max-sum classifier from non-separable examples can be expressed as a convex optimization problem solvable efficiently without calling any external max-sum solver. Our approach, named LP-M^{3}N formulation, generalizes the Maximum Margin Markov Network (M^{3}N) algorithm for learning of the Associative Markov Networks (AMN) [13]. We show that the LP-M^{3}N problem can be solved efficiently by the Generalized Proximal Point Algorithm. We show experimentally that our algorithm speeds up the learning by a factor of up to 20 compared to the ACP algorithm.

The paper is organized as follows. Section 2 describes the LP relaxation of the max-sum problems which is essential for our approach. The existing M^{3}N methods for learning max-sum classifiers are outlined in Section 3. Our LP-M^{3}N formulation is described in Section 4. The Generalized Proximal Point Algorithm for optimization of the LP-M^{3}N problem is given in Section 5. Section 6 presents an empirical evaluation and Section 7 concludes the paper.

2. Linear Programming Relaxation

Computing the output of a classifier (1) requires solving the max-sum problem which is NP-complete in general. An approximate solution can be found by the linear programming (LP) relaxation proposed by [8] and later reinvented by [6, 16]. We introduce only the basic concepts of the LP relaxation necessary for this article. For more information we refer to a survey in [17].

Let $(\mathcal{T}\times\mathcal{Y},\mathcal{E}_{\mathcal{Y}})$ denote an undirected graph with edges $\mathcal{E}_{\mathcal{Y}}=\{\{(t,y),(t',y')\}\mid tt'\in\mathcal{E},y,y'\in\mathcal{Y}\}$. Each *node* $(t,y)\in\mathcal{T}\times\mathcal{Y}$ and each *edge* $\{(t,y),(t',y')\}\in\mathcal{E}_{\mathcal{Y}}$ is assigned a number $\beta_t(y)$ and $\beta_{tt'}(y,y')$, respectively. Let $\beta\in\mathbb{R}^{|\mathcal{T}||\mathcal{Y}|+|\mathcal{E}||\mathcal{Y}|^2}$ be an ordered tuple which contains elements $\beta_t(y)$, $(t,y)\in\mathcal{T}\times\mathcal{Y}$ and $\beta_{tt'}(y,y')$, $\{(t,y),(t',y')\}\in\mathcal{E}_{\mathcal{Y}}$. The LP relaxation of (1) reads

$$\beta^*\in\operatorname*{argmax}_{\beta}[\sum_{t\in\mathcal{T}}\sum_{y\in\mathcal{Y}}\beta_t(y)q_t(y,x_t;\theta)+$$
$$+\sum_{tt'\in\mathcal{E}}\sum_{(y,y')\in\mathcal{Y}^2}\beta_{tt'}(y,y')g_{tt'}(y,y';\theta)]$$
$$\textit{subject to} \qquad (2)$$
$$\sum_{y'\in\mathcal{Y}}\beta_{tt'}(y,y')=\beta_t(y),\ tt'\in\mathcal{E},y\in\mathcal{Y},$$
$$\sum_{y\in\mathcal{Y}}\beta_t(y)=1,\ t\in\mathcal{T},\quad \beta_t(y)\geq 0,\ t\in\mathcal{T},y\in\mathcal{Y}.$$

We denote the objective of (2) by $P(x,\beta;\theta)$. It can be seen that solving (2) with an additional constraint $\beta\in\{0,1\}^{|\mathcal{T}||\mathcal{Y}|+|\mathcal{E}||\mathcal{Y}|^2}$ is an integer linear programming (ILP) problem equivalent to (1). It implies that $P(x,\beta^*;\theta)\geq\max_{y\in\mathcal{Y}^{\mathcal{T}}}H(x,y;\theta)$ holds in general. Further, we introduce the Lagrange dual of the LP task (2) which will be later used to construct the learning algorithms. Let $\rho_{tt'}:\mathcal{Y}\to\mathbb{R}$ and $\rho_{t't}:\mathcal{Y}\to\mathbb{R}$ be a pair of functions introduced for each pair of neighbouring objects $tt'\in\mathcal{E}$, that is, we have $2|\mathcal{E}|$ functions in total. We will use $\rho\in\mathbb{R}^{2|\mathcal{E}||\mathcal{Y}|}$ to denote an ordered tuple which contains elements $\rho_{tt'}(y)$, $\{t,t'\}\in\mathcal{E}$, $y\in\mathcal{Y}$ and $\rho_{t't}(y')$, $\{t,t'\}\in\mathcal{E}$, $y'\in\mathcal{Y}$. Let further $N(t)=\{t'\in\mathcal{T}\mid\{t,t'\}\in\mathcal{E}\}$ denote the set of objects neighbouring with the object $t\in\mathcal{T}$. Let us define a convex function

$$D(x,\rho;\theta)=\sum_{t\in\mathcal{T}}\max_{y\in\mathcal{Y}}[q_t(x_t,y;\theta)-\sum_{t'\in N(t)}\rho_t(y)]+$$
$$+\sum_{tt'\in\mathcal{E}}\max_{\{y,y'\}\in\mathcal{Y}^2}[g_{tt'}(y_t,y_{t'}\theta)+\rho_{tt'}(y_t)+\rho_{tt'}(y_{t'})]. \qquad (3)$$

It can be shown that the dual of the LP relaxation (2) is equivalent to

$$\rho^*\in\operatorname*{argmin}_{\rho}D(x,\rho;\theta). \qquad (4)$$

By the weak duality theorem, $D(x,\rho;\theta)\geq$ $\geq P(x,\beta^*;\theta)$, $\forall\rho$, and thus also $D(x,\rho;\theta)\geq$ $\geq\max_{y\in\mathcal{Y}^{\mathcal{T}}}H(x,y;\theta),\forall\rho$. It means that $D(x,\rho;\theta)$

provides an upper bound on the optimal value of the max-sum problem which will be later used to construct the learning algorithms. The problem (4) can be interpreted as searching for the best (that is, minimal) upper bound.

Provided the LP relaxation is tight, one can compute the maximizer $y^* = \arg\max_{y\in\mathcal{Y}^\mathcal{T}} H(x,y;\theta)$ from ρ^* by solving an instance of the Constraint Satisfaction Problem (CSP). The CSP is a special case of the max-sum problem when the quality functions q_t, $g_{tt'}$ attain values $\{-\infty,0\}$. The CSP itself is NP-complete in general. There is no guarantee that we can compute y^* even if $D(x,\rho^*;\theta) = H(x,y^*;\theta)$ holds and thus computing the optimal solution y^* is a more complex than computing the value $H(x,y^*;\theta)$ which can be always approximated by $D(x,\rho^*;\theta)$ solvable in a polynomial time.

The LP tasks (2) and (4) are not the only possible way to approximate the max-sum problem (1). If the graph $(\mathcal{T},\mathcal{E})$ can be easily decomposed to acyclic sub-graphs with non-overlapping edges, the task (4) can be replaced by an equivalent problem potentially more suitable for optimization [10]. Let us consider a grid graph $(\mathcal{T},\mathcal{E})$ useful in image analysis, that is, $\mathcal{T} = \{(i,j)\,|\, i=1,\dots,\text{Height}, j=1,\dots,\text{Width}\}$ and $\mathcal{E} = \{\{(i,j),(i',j')\}\,|\,(i,j)\in\mathcal{T},(i',j')\in\mathcal{T}, |i-i'|+|j-j'|=1\}$. We define the sets of horizontal $\mathcal{E}_H$ and vertical $\mathcal{E}_V$ edges as follows

$$\mathcal{E}_H = \{\{(i,j-1),(i,j)\}\,|\,1\le i\le \text{Height}, 1< j\le \text{Width}\},$$

$$\mathcal{E}_V = \{\{(i-1,j),(i,j)\}\,|\,1< i\le \text{Height}, 1\le j\le \text{Width}\}.$$

It can be seen that $\mathcal{E} = \mathcal{E}_H \cup \mathcal{E}_V$ and $\mathcal{E}_H \cap \mathcal{E}_V = \{\varnothing\}$. Let $\varphi_t : \mathcal{T} \to \mathcal{Y}$ be functions defined for each object $\mathcal{T}$ and let $\varphi \in \mathbb{R}^{|\mathcal{Y}||\mathcal{T}|}$ denote an ordered tuple which contains elements $\varphi_t(y)$, $t\in\mathcal{T}$, $y\in\mathcal{Y}$. It is easy to see that

$$H(x,y;\theta) = \left[\sum_{t\in\mathcal{T}}\left(\frac{1}{2}q_t(x_t,y_t;\theta)+\varphi_t(y_t)\right) + \sum_{tt'\in\mathcal{E}_H} g_{tt'}(y_t,y_{t'})\right] + \left[\sum_{t\in\mathcal{T}}\left(\frac{1}{2}q_t(x_t,y_t;\theta) - \varphi_t(y_t)\right) + \sum_{tt'\in\mathcal{E}_V} g_{tt'}(y_t,y_{t'})\right] \quad (5)$$

holds $\forall\varphi$. Taking the maximum over the terms in (5) independently we get

$$U(x,\varphi;\theta) = \max_{y\in\mathcal{Y}^\mathcal{T}}\left[\sum_{t\in\mathcal{T}}\left(\frac{1}{2}q_t(x_t,y_t;\theta) + \varphi_t(y_t)\right) + \sum_{tt'\in\mathcal{E}_H} g_{tt'}(y_t,y_{t'})\right] + \max_{y\in\mathcal{Y}^\mathcal{T}}\left[\sum_{t\in\mathcal{T}}\left(\frac{1}{2}q_t(x_t,y_t;\theta) - \varphi_t(y_t)\right) + \sum_{tt'\in\mathcal{E}_V} g_{tt'}(y_t,y_{t'})\right], \quad (6)$$

which is obviously an upper bound on the optimal value $\max_{y\in\mathcal{Y}^\mathcal{T}} H(x,y;\theta)$. The upper bound $U(x, \varphi, \theta)$ can be evaluated efficiently since it involves two max-sum problems with an acyclic neighboring structure solvable by the dynamic programming. It can be proved [10] that the upper bounds $D(x,\rho;\theta)$ and $U(x,\varphi;\theta)$ are equivalent in the sence that

$$\min_\rho D(x,\rho;\theta) = \min_\varphi U(x,\varphi;\theta).$$

Minimizing the bound $U(x,\varphi;\theta)$ involves smaller number of variables compared to the bound $D(x,\rho;\theta)$ and thus is preferable for large scale problems.

3. Maximal Margin Markov Networks

In this section, we describe a discriminative approach for learning the quality functions q_t, $g_{tt'}$. We consider learning from a single example $(\hat{x},\hat{y})\in\mathcal{X}^\mathcal{T}\times\mathcal{Y}^\mathcal{T}$. This assumption considerably simplifies notation but all the introduced algorithms can be easily extended for a finite number of examples.

Let $L:\mathcal{Y}^\mathcal{T}\times\mathcal{Y}^\mathcal{T}\to\mathbb{R}$ be a loss function penalizing a prediction $h(x;\theta)$ by a penalty $L(y, h(x;\theta))$ provided the true hidden variables are y.

We assume that L is additive over the objects $t\in\mathcal{T}$, that is, $L(y,y')=\sum_{t\in\mathcal{T}}L_t(y_t,y'_t)$, where $L_t:\mathcal{Y}\times\mathcal{Y}\to\mathbb{R}$, $t\in\mathcal{T}$. In addition, we assume that $L_t(y,y')=0$ for $y=y'$ and $L_t(y,y')>0$ otherwise. An example of a proper loss function is the Hamming distance $L(y,y')=\sum_{t\in\mathcal{T}}[[y_t\neq y_{t'}]]$ where $[[A]]=1$ if A is valid and $[[A]]=0$ otherwise. Further, we assume that the functions $q_t(x,y;\theta)$ and $g_{tt'}(y,y';\theta)$ are linear in the parameter $\theta\in\mathbb{R}^n$, that is,

$$\begin{aligned} q_t(x,y;\theta)&=\langle\Psi_t(x,y),\theta\rangle \quad \textit{and}\\ g_{tt'}(y,y')&=\langle\Psi_{tt'}(y,y'),\theta\rangle, \end{aligned} \tag{7}$$

where $\Psi_t:\mathcal{X}\times\mathcal{Y}\to\mathbb{R}^n$ and $\Psi_{tt'}:\mathcal{Y}\times\mathcal{Y}\to\mathbb{R}^n$ are arbitrary fixed mappings. Under the assumption (7), the max-sum classifier (1) becomes an instance of the linear multiclass classifier

$$\begin{aligned} h(x;\theta)&=\operatorname*{argmax}_{y\in\mathcal{Y}^\mathcal{T}}\langle\theta,\sum_{t\in\mathcal{T}}\Psi_t(x_t,y_t)+\sum_{tt'\in\mathcal{E}}\Psi_{tt'}(y_t,y_{t'})\rangle=\\ &=\operatorname*{argmax}_{y\in\mathcal{Y}^\mathcal{T}}\langle\theta,\Psi^q(x,y)+\Psi^g(y)\rangle=\\ &=\operatorname*{argmax}_{y\in\mathcal{Y}^\mathcal{T}}\langle\theta,\Psi(x,y)\rangle, \end{aligned}$$

where $\Psi^q:\mathcal{X}^\mathcal{T}\times\mathcal{Y}^\mathcal{T}\to\mathbb{R}^n$, $\Psi^g:\mathcal{Y}^\mathcal{T}\to\mathbb{R}^n$ and $\Psi:\mathcal{X}^\mathcal{T}\times\mathcal{Y}^\mathcal{T}\to\mathbb{R}^n$ are mappings constructed from Ψ_t and $\Psi_{tt'}$.

Given an example $(\hat{x},\hat{y})$, the parameter vector θ can be learned by minimizing the following convex problem

$$\theta^*=\operatorname*{argmin}_{\theta\in\mathbb{R}^n}F(\theta):=\left[\frac{1}{2}|\theta|^2+C\cdot R(\theta)\right], \tag{8}$$

where

$$\begin{aligned} R(\theta)=\max_{y\in Y^\mathcal{T}}&[L(\hat{y},y)+\langle\theta,\Psi(\hat{x},y)\rangle]-\\ &-\langle\theta,\Psi(\hat{x},\hat{y})\rangle, \end{aligned} \tag{9}$$

and $C>0$ is some prescribed constant. The max-sum classifier with parameters obtained by solving (8) is called the Maximum Margin Markov Network (M^3N) [14]. In the sequel, we will denote (8) as the M^3N problem.

The objective function $F(\theta)$ of the M^3N problem is composed of the quadratic regularization term $\frac{1}{2}\|\theta\|^2$ and the function $R(\theta)$ which is a convex approximation of the empirical risk, that is, $R(\theta)\geq L(y,h(x;\theta))$, $\forall\theta$. The quadratic term $\frac{1}{2}\|\theta\|^2$ and the regularization constant $C>0$ regularizes the solution in order to prevent the classifier from over-fitting. The constant C is typically tuned on a validation set. It can be shown that the quadratic term in the objective is equivalent to restricting the admissible parameters θ into a ball with a radius proportional to the constant C. In addition, for sufficiently large C the problem (8) becomes equivalent to solving $\theta^*=\operatorname{argmin}_\theta R(\theta)$.

Though the M^3N problem is convex it is not tractable due to the function $R(\theta)$ whose evaluation involves solving an instance of the max-sum problem which is NP-complete. The complexity of the M^3N problem becomes more apparent if it is transformed into an equivalent quadratic programming (QP) problem with $|\mathcal{Y}|^{|\mathcal{T}|}$ linear constrains and $n+1$ variables.

We now summarize the existing methods for solving (8). We consider only the methods that can deal with an arbitrary neighborhood graph $(\mathcal{T},\mathcal{E})$.

- Associative Markov Networks (AMNs) [13]. AMNs is the Markov Network with the quality functions satisfying $g_{tt'}(y,y')=0$, for $y\neq y'$. [13] proposed to replace the intractable function $R(\theta)$ in the definition of (8) by its LP relaxation which is particularly simple for the AMNs. In that manner the intractable M^3N problem (8) is approximated by a tractable QP task with $O(|\mathcal{E}||\mathcal{Y}|^2)$ linear constrains.

- Super-modular max-sum classifiers [5]. Under the assumption that the quality functions $g_{tt'}$ are super-modular, the max-sum problem can be solved in polynomial time. In turn, the value of the function $R(\theta)$ and its sub-gradient can be computed which makes it possible to solve the

M^{3}N problem efficiently by a variant of the cutting plane algorithm (CPA). The CPA of [5] finds ε-optimal solution of (8) in $O\left(\frac{1}{\varepsilon}\right)$ iterations.

• A max-sum classifier with strictly trivial equivalent (STE) [5]. The max-sum problems with STE are those whose response can be computed exactly by the LP relaxation. In the case of separable training examples, the parameters of the max-sum classifier can be found by solving a set of strict linear inequalities whose number is proportional to $O(|\mathcal{E}||\mathcal{Y}|^2)$. In turn, the parameters can be found by the Perceptron algorithm which is guaranteed to stop in a finite number of iterations. In the case of non-separable training examples, the parameters can be learned by minimizing a quadratically regularized upper bound on the training error defined for a zero-one loss function (the loss is 0 if all labels are correctly classified otherwise the loss is 1). The learning problem amounts to solving a QP task with $O(|\mathcal{E}||\mathcal{Y}|^2)$ linear constraints.

• General Markov Networks [5]. The M^{3}N problem can attacked by an Approximate Cutting Plane (ACP) algorithm. The APC algorithm requires computation of the value of $R(\theta)$ and its sub-gradient which involves solving an instance of an intractable max-sum problem. can be solved approximately by the LP relaxation. The resulting algorithm is guaranteed to stop in a finite number of iterations, however, there are no guarantees on the precision of the found solution. Moreover, the ACP algorithm is computationally demanding as it calls in each iteration the max-sum solver for each training example. Note, that the max-sum solvers are time consuming as they solve a large-scale linear program as well as the CSP problem to obtain the labels.

To sum up, there is currently no efficient method for learning parameters of a general max-sum classifier from non-separable examples. To close this gap, we show in Section 4 that the objective function of the M^{3}N problem can be replaced by another convex function whose value and sub-gradient can be computed efficiently without a need to call an external max-sum solver. In turn, the learning problem can be solved by a plethora of algorithms for non-smooth optimization. We describe one of such algorithm in Section 5.

4. Proposed LP-M^{3}N formulation

The key idea is to replace the intractable max-sum problem in the definition of the M^{3}N problem (8) by its upper bound derived from the LP relaxation (c.f. Section 2). Henceforth, we concentrate on the max-sum problem with a grid graph $(\mathcal{T},\mathcal{E})$ whose optimal value can be upper bounded by (6). Note, however, that the derivation for a general neighborhood graph using the bound (3) is analogical.

Using (6) we can upper bound the maximization term in (9) as follows

$$\max_{y\in\mathcal{Y}^{\mathcal{T}}}[L(\hat{y},y)+\langle\theta,\Psi(\hat{x},y)\rangle]\le$$

$$\le\max_{y\in\mathcal{Y}^{\mathcal{T}}}\left[\sum_{t\in\mathcal{T}}\left(\frac{1}{2}\big(q_t(\hat{x}_t,y_t;\theta)+L_t(\hat{y}_t,y_t)\big)+\varphi_t(y_t)\right)+\right.$$

$$\left.+\sum_{tt'\in\mathcal{E}_H}g_{tt'}(y_t,y_{t'})\right]+\max_{y'\in\mathcal{Y}^{\mathcal{T}}}\left[\sum_{t\in\mathcal{T}}\left(\frac{1}{2}\big(q_t(\hat{x}_t,y'_t;\theta)+\right.\right.$$

$$\left.\left.+L_t(\hat{y}_t,y'_t)\big)-\varphi_t(y'_t)\right)+\sum_{tt'\in\mathcal{E}_V}g_{tt'}(y_t,y'_{t'})\right]= \qquad (10)$$

$$=\max_{y\in\mathcal{Y}^{\mathcal{T}}}\left[\langle\theta,\frac{1}{2}\Psi^q(\hat{x},y)+\Psi^g(y)\rangle+L(\hat{y},y)+\right.$$

$$\left.+\langle\varphi,\Phi(y)\rangle\right]+\max_{\#y'\in\mathcal{Y}^{\mathcal{T}}}\left[\langle\theta,\frac{1}{2}\Psi^q(\hat{x},y')+\Psi^g(y')\rangle+\right.$$

$$\left.+L(\hat{y},y')-\langle\varphi,\Phi(y')\rangle\right],$$

where $\Phi:\mathcal{Y}^{\mathcal{T}}\to\mathbb{R}^{|\mathcal{Y}||\mathcal{T}|}$ is a mapping constructed appropriately. We define a function

$$R(\theta,\varphi)=\max_{y\in\mathcal{Y}^{\mathcal{T}}}\left[\langle\theta,\frac{1}{2}\Psi^q(\hat{x},y)+\Psi^g(y)\rangle+\right.$$

$$\left.+L(\hat{y},y)+\langle\varphi,\Phi(y)\rangle\right]+$$

$$+\max_{y'\in\mathcal{Y}^{\mathcal{T}}}\left[\langle\theta,\frac{1}{2}\Psi^q(\hat{x},y')+\Psi^g(y')\rangle+L(\hat{y},y')-\right. \qquad (11)$$

$$\left.-\langle\varphi,\Phi(y')\rangle\right]-\langle\theta,\Psi(\hat{x},\hat{y})\rangle.$$

Note, that $R(\theta,\varphi)$ and $R(\theta)$ are two different functions distinguished by their arguments. By (10) we have that $R(\theta)\le R(\theta,\varphi)$, $\forall\varphi\in\mathbb{R}^{|\mathcal{T}||\mathcal{Y}|}$, that is, $R(\theta,\varphi)$ is an LP-relaxation based upper bound on $R(\theta)$. By replacing $R(\theta)$ with $R(\theta,\varphi)$ in the definition of M^{3}N problem (8) we obtain

$$(\hat\theta,\hat\varphi)=\operatorname*{argmin}_{\theta\in\mathbb{R}^n,\varphi\in\mathbb{R}^{|\mathcal{T}||\mathcal{Y}|}} F(\theta,\varphi):=$$
$$:=\left[\frac{1}{2}\|\theta\|^2+C\cdot R(\theta,\varphi)\right]. \qquad (12)$$

We will denote (12) as the LP-M^{3}N problem. It is seen, that the objective $F(\theta,\varphi)$ is convex and it upper bounds the objective of the original M^{3}N problem (8) . An advantage of the LP-M^{3}N problem is that its objective value $F(\theta,\varphi)$ and sub-gradient $F'(\theta,\varphi)=[F'_\theta(\theta,\varphi);F'_\varphi(\theta,\varphi)]\in\mathbb{R}^{n+|\mathcal{T}||\mathcal{Y}|}$ can be computed efficiently as follows:

$$\tilde y=\max_{y\in\mathcal{Y}^{\mathcal{T}}}\left[\langle\theta,\frac{1}{2}\Psi^q(\hat x,y)+\Psi^g(y)\rangle+\right.$$
$$\left.+L(\hat y,y)+\langle\varphi,\Phi(y)\rangle\right],$$

$$\tilde y'=\max_{y'\in\mathcal{Y}^{\mathcal{T}}}\left[\langle\theta,\frac{1}{2}\Psi^q(\hat x,y')+\Psi^g(y')\rangle+\right.$$
$$\left.+L(\hat y,y')-\langle\varphi,\Phi(y')\rangle\right],$$

$$F'_\theta(\theta,\varphi)=\theta+C\cdot\left[\frac{1}{2}\Psi^q(\hat x,\tilde y)+\Psi^g(\tilde y)+\right.$$
$$\left.+\frac{1}{2}\Psi^q(\hat x,\tilde y')+\Psi^g(\tilde y')\right]-\Psi(\hat x,\hat y),$$

$$F'_\varphi(\theta,\varphi)=C\cdot[\Phi(\tilde y)-\Phi(\tilde y')],$$

$$F(\theta,\varphi)=\langle\theta,F'_\theta(\theta,\varphi)\rangle+\langle\varphi,F'_\varphi(\theta,\varphi)\rangle-\frac{1}{2}\|\theta\|^2.$$

Recall, that the maximization tasks are solvable by the dynamic programming.

5. Generalized Proximal Point Algorithm

The objective function $F(\theta,\varphi)$ of the LP-M^{3}N problem is convex and non-differentiable. This suggest usage of the methods from non-smooth optimization. The simples options is the plain sub-gradient algorithm which starts from an arbitrary $(\theta_0,\ \varphi_0)$ and then iteratively computes

$$\theta_{t+1}=\theta_t-\lambda_t\frac{F'_\theta(\theta_t,\varphi_t)}{\|F'_\theta(\theta_t,\varphi_t)\|}\quad \textit{and}$$

$$\varphi_{t+1}=\varphi_t-\lambda_t\frac{F'_\varphi(\theta_t,\varphi_t)}{\|F'_\varphi(\theta_t,\varphi_t)\|},$$

where $\lambda_0,\ldots,\lambda_\infty$ is a prescribed sequence of positive numbers satisfying $\sum_{t=0}^{\infty}\lambda_t=\infty$ and $\lim_{t\to 0}\lambda_t=$ $=0$. The sub-gradient algorithm is guaranteed to converge to the optimal solution of the LP-M^{3}N problem.

In this section, we describe another algorithm for non-smooth optimization which often convergences faster compared to the plain sub-gradient algorithm. In particular, we use the Generalized Proximal Point Algorithm (GPPA) [7]. Let us define an auxiliary objective function

$$F_t(\theta,\varphi):=F(\theta,\varphi)+\frac{1}{\lambda_t}\|\varphi-\varphi_t\|^2, \qquad (13)$$

where $\varphi_t\in\mathbb{R}^{|\mathcal{T}||\mathcal{Y}|}$ is a vector and $\lambda_t>0$ is a scalar. The added quadratic term makes the auxiliary objective $F_t(\theta,\varphi)$ strictly convex and more amenable to minimization compared to the original objective $F(\theta,\varphi)$.

There are several methods which can solve the auxiliary problem (13) efficiently. We use the Bundle Method (BM) algorithm described in [15]. The BM algorithm only requires to access the optimized objective via its value and sub-gradient. The BM algorithm is guaranteed to find ε-optimal solution in $O\left(\frac{1}{\varepsilon}\right)$ time. In its inner loop, the BM algorithm solves instances of convex QP of a small size with particularly simple linear constraints. We used the QP solver described in [4].

It is seen that $F_t(\theta,\varphi)$ upper bounds $F(\theta,\varphi)$. The approximation gets tighter for larger λ_t and for φ_t closer to the optimal vector $\hat\varphi$. The Generalized Proximal Point Algorithm 1 iteratively tightens the approximation by applying block-coordinate minimization on the auxiliary objective $F_t(\theta,\varphi)$.

Algorithm 1: Generalized Proximal Point Algorithm for LP-M^{3}N Problem

```
1. input & initialization: Set $\varphi_0 \leftarrow 0$, and select a sequence of positive numbers $\lambda_0,\dots,\lambda_\infty$;
2. repeat;
3. Solve the auxiliary problem
   $(\theta_{t+1},\varphi_{t+1}) = \operatorname*{argmin}_{\theta\in\mathbb{R}^n,\varphi\in\mathbb{R}^{|\mathcal{T}||\mathcal{Y}|}} F_t(\theta,\varphi)$ ;    (14)
4. until convergence.
```

It is easy to see that the sequence $(\theta_1,\varphi_1),\dots,(\theta_t,\varphi_t)$ generated by Algorithm 1 monotonically decreases the objective function $F(\theta,\varphi)$. Indeed, by a simple algebra we get that

$$F(\theta_t,\varphi_t) - F(\theta_{t+1},\varphi_{t+1}) \geq \frac{1}{\lambda_t}\|\varphi_{t+1}-\varphi_t\|^2.$$

There exists many convergence results regarding the standard PPA. Unfortunately, this is not the case for the GPPA which has not been studied so intensively. The result closest to our problem is the following convergence theorem proved in [7]:

Theorem 1 *Suppose* $F:\mathbb{R}^n\times\mathbb{R}^m \to (-\infty,+\infty]$ *is a function which is convex and inf-compact, that is, for each* γ *the set* $\{(\theta,\varphi)\,|\,F(\theta,\varphi)\leq\gamma\}$ *is compact. Let* $(\theta_0,\varphi_0),\dots,(\theta_\infty,\varphi_\infty)$ *be any sequence generated by Algorithm* 1. *Then*

$$\lim_{t\to\infty} F(\theta_t,\varphi_t) = F^* \text{ where } F^* = \min_{\theta,\varphi} F(\theta,\varphi).$$

Unfortunately, the theorem cannot be applied to our problem as the objective function $F(\theta,\varphi)$ is not inf-compact. Nevertheless, we provide a convincing empirical study of the convergence speed of Algorithm 1 in Section 6.

Algorithm 1 is specified up to the choice of the sequence of λ_t and a practical stopping condition. A proper selection of the sequence is delicate as it involves trade-off between the number of iterations and the complexity of the auxiliary problem (14). That is, higher values of λ_t leads to a large decrease in the objective value, however, the optimization of the auxiliary problem (14) gets harder and vice versa. Based on experiments, we found useful to use a geometrical sequence $\lambda_{t+1}=\lambda_t a$, where $a>1$ and $\lambda_0>0$ are selected constants.

In the comparison presented in Section 6 we stop the algorithm when the value of $F(\theta,\varphi)$ drops below a given threshold which is known a priory. In practice, when no such threshold is provided, one can stop the algorithm based on observing the objective function. Alternatively, the algorithm can be stopped when the added quadratic term in the auxiliary function $F_t(\theta,\varphi)$ gets sufficiently small meaning that $F_t(\theta,\varphi)$ is a tight approximation of $F(\theta,\varphi)$. That is the stopping condition may read

$$\frac{1}{\lambda^t}\|\varphi_{t+1}-\varphi_t\|^2 \leq \varepsilon. \qquad (15)$$

It is clear, that the stopping condition (15) is satisfied in a finite number of iterations for any $\varepsilon>0$. However, we do not have any result which would relate a solution (θ,φ) satisfying the stopping condition (15) to the optimal solution of the LP-M^{3}N problem (12). Derivation of theoretically grounded stopping condition will be a subject of our future research.

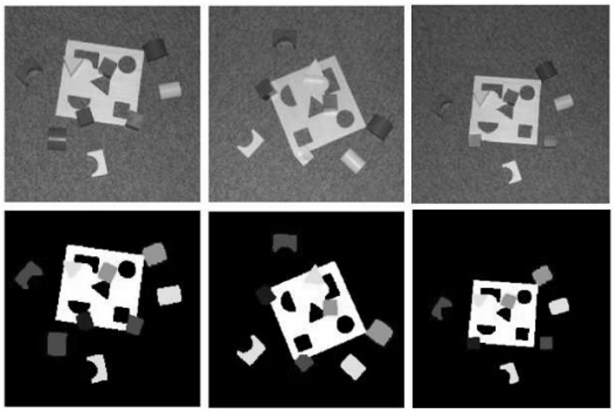
6. Experiments

We consider a problem of learning the max-sum classifier (1) for their color image segmentation. The training examples are color images along with ground-truth segmentation produced by a human annotator. The task is to learn the classifier (1) which produces the segmentation similar to the human. We use the Hamming loss $L(y,\hat{y})=\sum_{t\in\mathcal{T}}[[y_t\neq\hat{y}_t]]$.

The images are snapshots of a shape-sorter puzzle placed on a carpet. Figure shows example images. The task is to segment the input images into color blobs corresponding to the carpet and 5 colored puzzle pieces. The images are split into training set (14 images [100×100] pixels), the validation set (4 images [100×100] pixels) and testing set (4 images [200×200] pixels).

The objects $\mathcal{T}$ correspond to pixels of the input image. The edges $\mathcal{E}$ captures the 4-neighborhood structure of the pixels, that is, the graph $(\mathcal{T},\mathcal{E})$

is a discrete grid (c.f. Section 2 for definition). The observable state $x_t \in \mathcal{X} = \{1,\ldots,N_X\}$ is a natural number which encodes the RGB color of the t-th pixel. We used 3 bits per each color channel which means $N_X = 2^{3\cdot 3} = 512$. The assignment of the t-th pixel to a segments is determined by the label $y_t \in \mathcal{Y} = \{1,\ldots,N_Y\}$. The number of segments is $N_Y = 6$. We further assume that functions $q_t : \mathcal{X}\times\mathcal{Y} \to \mathbb{R}$ and $g_{tt'} : \mathcal{Y}\times\mathcal{Y} \to \mathbb{R}$ are discrete functions which do not depend on the pixel $t \in \mathcal{T}$ (called homogeneous model). Thus the dimension of the parameter vector θ is $n = N_X \cdot N_Y + N_Y \cdot N_Y$.


Example images and their segmentation predicted by a max-sum classifier learned from examples

We compared the Approximate Cutting Plane (ACP) algorithm [5] against the proposed Generalized Proximal Point Algorithm (GPPA) defined by Algorithm 1. We are not aware of any other discriminative approach able to solve this task. To solve the predictive step required by the ACP algorithm, we used a highly optimized implementation of the Augmenting DAG algorithm described in [17]. Both ACP and GPPA require to compute the objective value, its sub-gradient and also to solve an instance of the QP task. We used exactly the same implementation of these sub-tasks in both methods. Both ACP and GPPA return an upper bound of the objective value of the M^{3}N problem (8). This enables a fair comparison in terms of the objective function and the computational time (the wall clock time). The ACP algorithm also produces a lower bound $F_{LB}(\theta)$ on the optimal value $F(\theta^*)$. First, we run the ACP algorithm with stopping condition $(F_{UB}(\theta) - F_{LB}(\theta))/F(\theta) \le 0{,}01$ (precision 0.001 was not already attained by the ACP). Second, we run the proposed GPPA until it reached the precision better or equal to the ACP. The obtained objective function and the required computational time (on standard PC, Intel CPU 1,83 GHz, 2GB RAM) as a function of the varying regularization constant are presented in Table 1. It is seen, that the GPPA algorithm consistently attains more precise solution in a shorter time. The average speedup was about 20, that is, GPPA was more than an order of magnitude faster compared to the ACP.

For completeness, we also report the training, validation and test errors. For both algorithms these errors differed on the forth decimal place due to the tight stopping condition we used. Thus the values in Table are valid for both methods. The testing error for was 1,04 ± 0,02%.

The comparison of the approximated cutting plane (ACP) algorithm and the proposed generalized proximal point algorithm (GPPA) on color image segmentation problem. The better (smaller) values of the upper bound on the optimal value $F(\theta^*)$ and the computation time are boldfaced. The errors are percentage of misclassified pixels.

	ACP of [5]		Proposed GPPA			
C	$F_{UB}(\theta)$	Time [h:m:s]	$F(\theta,\varphi)$	Time [h:m:s]	TrnErr	ValErr
0.01	20.6042	1:3:15	**20.6018**	**0:7:31**	3.26	2.69
0.05	50.7373	2:2:43	**50.6869**	**0:11:5**	1.93	1.06
0.10	72.9288	2:44:45	**72.8832**	**0:12:34**	1.71	0.98
0.50	197.7133	4:53:29	**197.6659**	**0:14:48**	1.11	0.09
1.00	327.7952	6:47:25	**327.6003**	**0:19:26**	1.09	0.87
5.00	1279.7376	10:15:56	**1279.2701**	**0:21:4**	0.98	0.99
		27:47:33		**1:26:28**		

7. Conclusions

We have shown that the learning of a general Markov network can be expressed as a convex optimization problem which is solvable efficiently without calling any external max-sum solver. The key idea of our LP-M^{3}N formulation is to use the Linear Programing relaxation of the max-sum problem directly in the formulation of the Maximum Margin Markov Network learning algorithm. The resulting LP-M^{3}N problem can be solved by a plethora of algorithms for non-smooth optimization. We proposed a variant of the Generalized Proximal Point Algorithm which is able to solve large instances of the LP-M^{3}N problem. The empirical comparison demonstrates that our algorithm speeds up the learning of general Markov networks by a factor of up to 20 compared to the Approximate Cutting Plane Algorithm, so far the only existing learning algorithm for general Markov Networks.

Acknowledgments

Vojtěch Franc was supported by the Czech Ministry of Education project 1M0567 and by EC projects FP7-ICT-247525 HUMAVIPS, PERG04-GA-2008-239455 SEMISOL. Any opinions expressed in this paper do not necessarily reflect the views of the European Community. The Community is not liable for any use that may be made of the information contained herein. Pavel Laskov was supported by the Heisenberg Fellowship of the German Science Foundation.

1. *Altun Y.*, *Hofmann T.* Large Margin Methods for Label Sequence Learning // Proc. of 8th Europ. Conf. on Speech Communication and Technology, 2003.
2. *Altun Y.*, *Tsochantaridis I.*, *Hofmann T.* Hidden Markov Support Vector Machines // Proc. of 20th Intern. Conf. on Machine Learning, 2003.
3. *Collins M.* Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms // Proc. of the Conf. on Empirical Methods in Natural Lang. Proces., 2002.
4. *Franc V.* Optimization Algorithms for Kernel Methods // PhD thesis, Czech Technical University, Karlovo nám. 13, 12135, Prague, Czech Rep., July 2005.
5. *Franc V.*, *Savchynskyy B.* Discriminative Learning of Max-Sum Classifiers // J. of Machine Learning Research. – Jan. 2008. – **9**(1). – P. 67–104.
6. *Koster A. Hoesel C.P.M. Kolen A.W.J.* The Partial Constraint Satisfaction Problem: Facets and Lifting Theorems // Operations Research Letters. – 1998. – **23**(3–5). – P. 89–97.
7. *Medhi D. Ha C.D.* Generalized Proximal Point Algorithm for Convex Optimization // J. of Optimization Theory and Applications. – 1996. – **88**(2). – P. 475–488.
8. *Schlesinger M.I.* Syntactic analysis of two-dimensional visual signals in noisy conditions // Kibernetika. – 1976. – N 4. – P. 113–130. In Russian.
9. *Schlesinger M.I.*, *Flach B.* Some solvable sublcasses of structural recognition problems // Proc. of the Czech Pattern Recognition Workshop, 2000. Czech Pattern Recognition Society.
10. *Schlesinger M.I.*, *Giginyak V.V.* Solving (max,+) problems in structural recognition using equivalent transformations // Systems and machines for control. – 2007. – N 2. – P. 5–17. In Russian.
11. *Schlesinger M.I.*, *Hlaváč V.* Ten lectures on statistical and structural pattern recognition // Kluw. Acad. Publ., 2002.
12. *Taskar B.* Learning Structured Prediction Models: A Large Margin Approach. PhD thesis, Stanford Univ., 2004.
13. *Taskar B.*, *Chatalbashev V.*, *Koller D.* Learning Associative Markov Networks // Proc. of 21th Intern. Conf. Machine Learning, 2004.
14. *Taskar B.*, *Guestrin C.*, *Koller D.* Maximum-margin Markov Networks // Proc. of Neural Inform. Proces. Syst., 2003.
15. *Teo C.H.*, *Vishwanathan S.V.N.*, *Smola A.*, *Quoc, V.Le.* Bundle Methods for Regularized Risk Minimization // J. of Machine Learning Research. – 2010. – N 11. – P. 311–365.
16. *Wainwright M.*, *Jaakkola T.*, *Willsky A.* MAP estimation via agreement on hypertrees: message passing and linear programming approaches // Conf. on Communication, Control and Computing, 2002.
17. *Werner T.* A Linear Programming Approach to Max-sum Problem: A Review // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2007. – **29**(7).

© V. Franc, Center for Machine Perception, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University, Technická 2, 166 27 Prague 6, Czech Republic, P. Laskov, Wilhelm-Schickard-Institut für Informatik, Sand 13, 72076 Tübingen, Germany, 2011

●