A Transductive Max-Margin Framework for Completion of Structured Variables with Application to Semi-Supervised Graph Inference

Stuart J. Andrews and Tony Jebara, Columbia University, New York, NY 10027

The task of learning to predict structured output variables has received a great deal of attention recently (see [3] for a broad overview). In this setting, $\mathbf{x} \in \mathbb{X}$ is an input feature vector representation of an object, and $\mathbf{y} \in \mathbb{Y}$ is a multi-variate output label vector, typically with binary entries, that specifies the underlying structure of the object. Note that typically $\mathbb{Y} \subsetneq \{0,1\}^N$ restricting $\mathbf{y}$ to be a member of a combinatorial family of valid structures. Structures of interest include chains (label-sequences), trees (parse trees), matchings (word and sequence alignments), and partitions of graphs (image segmentations). Application areas include natural language processing, computational molecular biology, and image processing.

Semi-supervised techniques have recently been developed [1], [5], wherein the *training data set* is partially observed in the following sense: some structured examples are labeled $(\mathbf{x}, \mathbf{y})$ and some are not $(\mathbf{x}, \cdot)$ and the goal is to predict the missing labels. We formalize an alternate mode of semi-supervised learning that addresses the problem of learning from examples $(\mathbf{x}, \mathbf{y})$ where some of the *structures* $\mathbf{y}$ are incomplete (i.e. only a subset of the entries in the multi-variate $\mathbf{y}$ are observed), and the goal is their *completion*. The motivation for this formulation comes from the observation that structured examples are difficult and/or expensive to collect, while on the other hand, partial labelings are typically less expensive. Partial labelings are often easier to annotate by hand, or can even be extracted from other sources. This form of learning has received attention for univariate predictions, for example in multiple-instance learning [2], but not apparently for structured outputs.

In learning to predict structured outputs, a risk minimization framework is typically employed, leading to a max-margin problem formulation. First, we introduce a multi-variate joint feature map $\Phi(\mathbf{x}, \mathbf{y}) = \mathbf{F}_{\mathbf{x}}\mathbf{y}$ that can be expressed as a matrix-vector product for some matrix $\mathbf{F}_{\mathbf{x}}$ (true for applications listed above). Next, we define the linear discriminant function $f(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \Phi(\mathbf{x}, \mathbf{y})$ parameterized by a weight vector $\mathbf{w}$. The discriminant function is used to rank the structures $\mathbf{y} \in \mathbb{Y}$ for a given $\mathbf{x}$. By the factorization, we have a bilinear form for $f(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \mathbf{F}_{\mathbf{x}} \mathbf{y}$. For a given weight vector $\mathbf{w}$, the vector-matrix product $\mathbf{w}^T \mathbf{F}_{\mathbf{x}}$ comprises a multivariate score vector $\mathbf{s}^T$, with which predictions are made via $\operatorname{argmax}_{\mathbf{y} \in \mathbb{Y}} \mathbf{s}^T \mathbf{y}$. Structured output predictions are made by identifying the structure from the combinatorial family $\mathbb{Y}$ that is closest to the score vector. To learn $\mathbf{w}$ from *fully labeled* structured data, we solve the following optimization problem, where for simplicity we assume there is only *one* training example $(\mathbf{x}, \mathbf{y})$:

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + C\xi \quad \text{s.t. } \xi \geq \max_{\mathbf{z} \in \mathbb{Y}} \Delta(\mathbf{y}, \mathbf{z}) - \mathbf{w}^T \mathbf{F}_{\mathbf{x}} (\mathbf{y} - \mathbf{z}) \ . \tag{1}$$

The margin is defined in terms of a "gap" between the true output and the highest-ranking alternative output. We use the term $\Delta(\mathbf{y}, \mathbf{z})$, typically the Hamming loss, to encourage greater separation of vastly different structures. Discriminative learning algorithms include the averaged perceptron, the exponentiated gradient algorithm, sequential minimal optimization, cutting-plane methods, extra and dual-extragradient methods, and search-based optimization.

In completion problems, structured variables are partitioned into observed and unobserved components $\mathbf{y} = \{\mathbf{y}^{\mathcal{O}}\} \cup \{\mathbf{y}^{\mathcal{U}}\}$. Our framework adopts the technique used in transductive inference for binary classification problems, wherein the missing labels $y = \pm 1$ are chosen during learning to yield the largest margin and smallest slack $y = \operatorname{argmin}_{y=\pm 1} \left(1 - y\mathbf{w}^T \mathbf{x}\right)$. To this end, we replace the margin constraints in Eq. (1) with

$$\min_{\mathbf{v} \in \mathbb{Y}, \mathbf{v}^{\mathcal{O}} = \mathbf{y}^{\mathcal{O}}} \max_{\mathbf{z} \in \mathbb{Y}} \Delta(\mathbf{v}, \mathbf{z}) - \mathbf{w}^T \mathbf{F}_{\mathbf{x}} (\mathbf{v} - \mathbf{z}) \ . \tag{2}$$

For a fixed hyperplane, computing the "gap" thus requires the prediction of two structures. We generalize the following algorithms to approximate this new optimization problem: (1) the averaged perceptron; (2) the cutting-plane approach for solving convex optimization problems (CUT SVM); and (3) the dual-extragradient algorithm for solving saddle-point problems (DXG). Details omitted.

We demonstrate the proposed formulation on a graph inference problem, where the goal is to *complete* a partially observed directed graph, as in [4]. We represent a graph using n^2 structure variables $\mathbf{y}_{j,k}$ corresponding to the graph's

	i.i.d. SVM (c)		DXG (d)		Perceptron (e)		DXG (f)		CUT SVM (g)		DXG (h)	
	mean	std.	mean	std.	mean	std	mean	std	mean	std	mean	std.
acc	97.4	0.6	96.9	0.4	97.0	0.4	**97.8**	0.6	**97.8**	0.6	*99.4*	0.2
auc	0.97	0.02	0.84	0.13	0.92	0.04	**0.96**	0.02	0.72	0.07	*1.00*	0.01
recall	33.8	14.3	20.1	9.3	23.2	7.4	42.3	13.5	**45.9**	14.5	*82.2*	6.2

Table 1: Results averaged over 10 repeated trials, where each trials randomly selects a 50:50 split of the nodes to derive training and test splits as in Fig. 1(a).

0-1 adjacency matrix. Moreover, we define structure *variables* $\mathbf{y}_k^{in}$, $\mathbf{y}_k^{out}$ representing the degree of each node. The only constraints on $\mathbf{y} = \left(\mathbf{y}_{j,k}, \mathbf{y}_k^{in}, \mathbf{y}_k^{out} : 1 \leq j, k \leq n\right)$, imposed by the combinatorial family $\mathbb{Y}$, is that the degree variables must agree with the adjacency matrix variables; i.e. a valid structure satisfies $\sum_j \mathbf{y}_{j,k} = \mathbf{y}_k^{in}, \forall k$, and $\sum_k \mathbf{y}_{j,k} = \mathbf{y}_j^{out}, \forall j$. The addition of the degree variables allows us to learn to predict *the degree of each node* from the training data. As input, we receive a set of n nodes, a set of d attributes for each node that together comprise $\mathbf{x}$, and an incomplete adjacency matrix $\mathbf{y}^{obs}$, as indicated in Fig. 1(a). We designed a joint feature map to assign score values to each directed edge based on the node features at the head and tail of the edge. The scoring function also assigns a value to each degree that a node may assume. The projection step returns a 0-1 graph closest to these scores, via a compact reduction to a degree-constrained subgraph problem whose solution can be found in $\mathcal{O}\left(n^3\right)$ time.

Fig. 1 gives some insight into the predictions made by two baseline algorithms and several algorithmic instantiations of our transductive framework. The numerical results in Table 1 provide more conclusive evidence of the relative performance of these methods. The *i.i.d.* SVM baseline (c) ignores structure yet still appears to perform reasonably well. The same is true for the dual extra-gradient baseline (d) that does not attempt to use transduction. However, on closer inspection of Table 1, one observes that the area under the curve (AUC), and recall performance metrics for these algorithms are inferior to most techniques that do use transduction. The one exception is the averaged perceptron algorithm (e) which uses transduction, yet still has poor performance. This can be attributed to the lack of regularization of the weight vector $\mathbf{w}$, or potential inseparability of the data. The dual extragradient algorithm with transduction (f), and the cutting-plane algorithm (g), both use transduction to their advantage, obtaining high AUC and recall. For small problems, we favour the cutting-plane method, because it tends to give better results. However, our implementation of the dual extragradient method is notably faster on larger problems, primarily due to the fact that it does not guarantee integral solutions. The final result, in (h) shows the performance of the dual-extragradient with transduction when the true degrees are provided as inputs to the algorithm. Notice the large increase in recall. Comparing this result to (f) highlights the difficulty and importance of the estimation of node degrees.

This work introduces a framework for semi-supervised learning to complete structured output variables. We outline the key optimization problem, and use generalized versions of structured learning algorithms to find approximate solutions. Results are presented on a graph inference problem. In future, we hope to reduce the runtime complexity of these algorithms and improve the robustness in the case of sparse graphs.

References

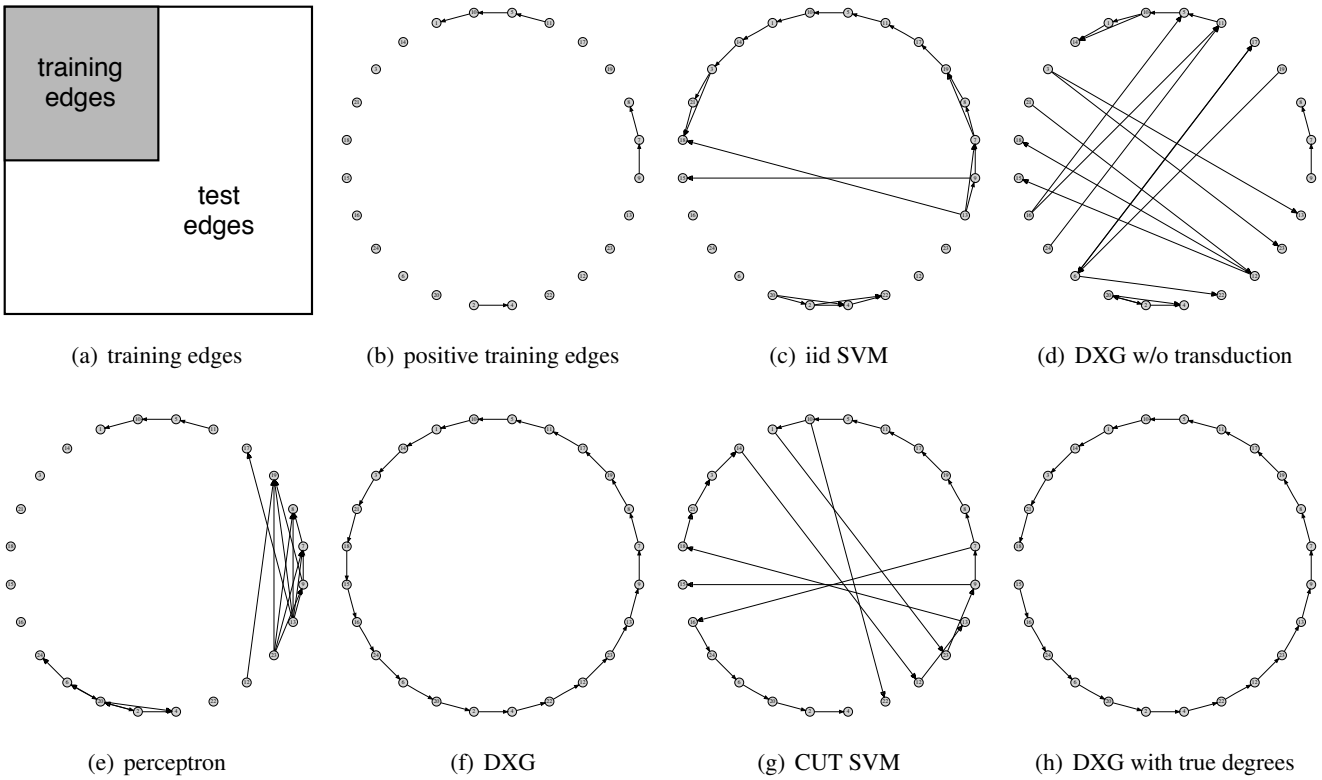
[1] Y. Altun, D. McAllester, and M. Belkin. Maximum margin semi-supervised learning for structured variables. *Advances in Neural Information Processing Systems*, 18, 2005.

[2] S. Andrews. *Learning from Ambiguous Examples*. PhD thesis, Brown University, 2007.

[3] G. Bakir, T. Hofmann, B. Schölkopf, and S. V. N. Vishwanathan. *Predicting Structured Data*. MIT Press, Cambridge, Massachusetts, 2007.

[4] Y. Yamanishi, J. P. Vert, and M. Kanehisa. Protein network inference from multiple genomic data: a supervised approach. *Bioinformatics*, 2004.

[5] A. Zien, U. Brefeld, and T. Scheffer. Transductive support vector machines for structured variables. *Proceedings of the 24th international conference on Machine learning*, pages 1183–1190, 2007.



(a) training edges (b) positive training edges (c) iid SVM (d) DXG w/o transduction

(e) perceptron (f) DXG (g) CUT SVM (h) DXG with true degrees

Figure 1: The partition of the adjacency matrix used for training is depicted in (a). Note that most of the entries in the training subset will be zero, which we call negative edges. The positive edges used for training are shown in (b). The remaining figures show the completed graphs after learning. These depict positive edges predicted on the test set, in addition to those from the training set. False-positives are demarked by open arrows, which can be seen by magnifying the individual figures (sorry!). Results include (c) *i.i.d.* SVM, (d) dual extragradient without using transduction, (e) averaged perceptron, (f) dual extragradient (g) cutting-plane SVM, and (h) dual extragradient augmented with an oracle that provides the true degree of each node. Algorithms (e)-(f) are derived from our proposed transductive framework.