Joint Kernel Support Estimation for Structured Prediction

Christoph H. Lampert, Matthew B. Blaschko {CHL, BLASCHKO}@TUEBINGEN.MPG.DE
Max Planck Institute for Biological Cybernetics, Tübingen, Germany

Abstract

We present a new technique for structured prediction that works in a hybrid generative/discriminative way, using a one-class support vector machine to model the joint probability of *(input, output)*-pairs in a joint reproducing kernel Hilbert space.

Compared to discriminative techniques, like *conditional random fields* or *structured output SVMs*, the proposed method has the advantage that its training time depends only on the number of training examples, not on the size of the label space. Due to its generative aspect, it is also very tolerant against ambiguous, incomplete or incorrect labels.

Experiments on realistic data show that our method works efficiently and robustly in situations for which discriminative techniques have computational or statistical problems.

1. Introduction

We study the problem of structured prediction, i.e. the task of learning a function $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{Y}$ is not just a simple boolean or discrete variable, but can have a rich substructure. Typical examples are graph-labeling problems, *e.g.* in *natural language parsing* or *image segmentation*. While some generative prediction techniques have been known and used for several decades, new momentum has been given to the field by recent discriminative techniques such as *conditional random fields* (CRFs (Lafferty et al., 2001), KCRFs (Lafferty et al., 2004)), *structured perceptrons* (Collins, 2002; Kashima & Tsuboi, 2004), and *structured maximum margin techniques* (M^3N (Taskar et al., 2003), S-SVM (Tsochantaridis et al., 2005)).

These discriminative methods often achieve superior performance compared to earlier generative ones, but they also come with certain disadvantages. In particular their training is often computationally costly, and they do not always deal well with training data that contains very noisy or ambiguous labels.

In this paper we introduce *joint-kernel support estimation* (JKSE), a new method for structured prediction that avoids these problems by using a hybrid generative/discriminative setup. The central idea is to work with the joint probability density of samples and labels, like generative models do. However, instead of fully modelling the density itself, we only estimate its support using a discriminative one-class support vector machine (Schölkopf et al., 2001). JKSE can handle both structured input and output spaces through the use of a joint kernel function over the input and output domains. It is trained by solving a convex optimization problems for which efficient standard software packages are available, making JKSE applicable to datasets with tens of thousands of examples or more, orders of magnitude larger than what many other techniques for structured output prediction can handle.

2. Joint Kernel Support Estimation

Following the common language of the field we formulate structured prediction in probabilistic terms as a MAP-prediction problem. By $\mathcal{X}$ we denote the space of observations and by $\mathcal{Y}$ the space of possible outputs. Note that for our setup it is not required that $\mathcal{X}$ or $\mathcal{Y}$ decompose into smaller entities like nodes or edges in a graph. We assume that *(sample, label)* pairs (x, y) follow a joint-probability density $p(x, y)$, and that a set of i.i.d. samples $(x_i, y_i)_{i=1,\dots,n}$ is available for training. The task then consists of learning a mapping $f : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected loss in a classification sense, i.e. for a new sample $x \in \mathcal{X}$, we want to determine the labeling $y \in \mathcal{Y}$ that maximizes the posterior probability $p(y|x)$ or, equivalently, the joint probability $p(x, y)$.

In joint kernel support estimation we follow a generative path and form a model of the joint likelihood $p(x, y)$ from the given training data. However, since density estimation in high dimensional spaces is notoriously difficult, we simplify the problem by assuming that the posterior probability in the feature space is *distinctive*, i.e. $p(y|x) \gg 0$ for correct predictions y and $p(y|x) \approx 0$ for incorrect y. Consequently, $p(x, y) \gg 0$ only if y is a correct label for x, and it suffices to estimate the *support* of $p(x, y)$ instead of the full density. Because $p(y|x) \approx 0$ implies $p(x, y) \approx 0$, we can afterwards still use $f(x) := \operatorname{argmax} p(x, y)$ for

prediction. Note that we do not require $p(y|x)$ to be unimodal: different $y \in \mathcal{Y}$ can be "correct" predictions for $x \in \mathcal{X}$, a situation that occurs quite frequently in realistic structured prediction tasks.

The generative setup limits JKSE's ability to *extrapolate*, because $p(x, y) = p(y|x)p(x)$ and for a test sample $x \in \mathcal{X}$ this expression can vanish not only for wrong predictions y but also if x lies an area of $\mathcal{X}$ not covered by the training set. However, the generative setups also have certain advantages, *e.g.*, the possibility for adaptive and online learning by building a smaller model of $p(x, y)$ first and later extending it without or with only very little retraining.

2.1. Representation

We model the central quantity of interest $p(x, y)$ by a log-linear model

$$p(x, y) = \frac{1}{Z} \exp(\, w^t \phi(x, y)\,) \tag{1}$$

with the *partition function* $Z = \sum_{x,y} \exp(\, w^t \phi(x, y)\,)$. Because our model is generative, *Z depends neither on x nor on y* and we will see later that it can be ignored during both training and inference. Consequently, the prediction step reduces to

$$f(x) = \mathrm{argmax}_{y \in \mathcal{Y}}\, w^t \phi(x, y) \tag{2}$$

which is equivalent to a MAP-estimate by Bayes' rule.

The feature map $\phi(x, y)$ is completely generic. While in many situations, such as sequence labeling, it is natural to form $\phi(x, y)$ by a concatenation or summation of per-site features and neighborhood relations, we do not formally require such a decomposition. Furthermore, $\phi(x, y)$ does not have to be explicit and in the following we will assume that it is given only implicitly by a suitable positive definite joint kernel function $k : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R}$.

2.2. Parameter Learning

To train JKSE, we only need to find a suitable weight vector w such that the right hand side of (1) reflects $p(x, y)$ over the training set. Since we are mainly interested in the support of $p(x, y)$, we use a *one-class support vector machine* (1-SVM) concept for this purpose. It allows the robust estimation of quantiles of probability densities in high dimensional spaces (Schölkopf et al., 2001). 1-SVM training consists of finding the hyperplane in a latent Hilbert space $\mathcal{H}$ that best separates the training samples from the origin with the exception of possible *outliers* that are identified automatically during the training procedure. A free parameter $\nu \in (0, 1]$ acts as an upper bound to the percentage of outliers, *i.e.* the larger ν is, the more freedom the method has to disregard any of the training samples.

In JKSE, the role of the data samples is taken by *(sample, label)* pairs, and $\mathcal{H}$ is induced by the joint kernel function $k(\,(x, y), (x', y')\,)$. By dualization, we can write JKSE training as solving the convex optimization problem

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i,j} \alpha_i \alpha_j K_{ij} - \sum_i \alpha_i K_{ii} \tag{3}$$

subject to $0 \le \alpha_i \le \frac{1}{\nu m}$ and $\sum_i \alpha_i = 1$, where $K_{ij} = k(\,(x_i, y_i), (x_j, y_j)\,)$ is the joint-kernel matrix. Note that the solution to (3) is generally sparse, *i.e.* most coefficients α_i vanish and need not be considered when evaluating the *JKSE decision function*, which is

$$f(x) = \mathrm{argmax}_{y \in \mathcal{Y}} \sum_{i=1}^{n} \alpha_i\, k(\,(x_i, y_i), (x, y)\,). \tag{4}$$

In the following, we will assume that some inference algorithm for calculating the argmax in (4), or a suitable approximation to it, is available.

Note that neither at training nor at prediction time do we need a method to calculate or approximate the partition function Z. In particular, because JKSE training relies only on the matrix of joint-kernel values, the training time depends only on the number of samples in the training set, not on the structure of the output space. This is in contrast other techniques for structured prediction, like *CRFs* or *S-SVMs* that require marginalization or repeated inference over the label space during training. As an additional advantage, the automatic identification of outliers in the training set by a suitable choice of ν allows training of JKSE even on data with a significant number of label errors.

2.3. Large Scale Training

Solving the quadratic optimization problem (3) has in principle the same computational complexity as training an ordinary two-class SVM. However, because 1-SVMs are less popular for pattern recognition tasks, significantly less effort has been spend on developing fast training routines and on optimizing the implementations. Existing packages, e.g. `libSVM`, can handle thousands of examples, but not tens of thousands.

We can overcome this limitation by reformulating JKSE training as a *binary classification problem*, for which faster training techniques are available. The original analysis in (Schölkopf et al., 2001) shows that –for datasets that can be linearly separated from the origin and in the *hard-margin case ($\nu \to 0$)*– the weight vector found by optimizing the 1-SVM problem (3) is equivalent to solving the optimization problem of a regular support vector machine for binary classification with only positive training examples and a classification hyperplane that passes through the origin. Equivalently, one can allow arbitrary hyperplanes, but

augment the training set with a mirrored copy of the data and inverted labels. In the soft-margin case, there is a one-to-one correspondence between the parameter ν of the one-class problems and the parameter C of the two-class problem such that both methods results in the same separation hyperplane. Consequently, a 1-SVM with model selection over ν can equivalently be implemented by a bias term free SVM with model selection over C. This allows us to make use of existing software packages for large scale SVM training, e.g. (Bordes et al., 2005), to train JKSE with datasets of tens of thousand of examples or more. For feature maps that can be computed explicitly, specialized linear SVM solvers, e.g. (Joachims, 2006; Shalev-Shwartz et al., 2007; Lin et al., 2008; Hsieh et al., 2008), allow JKSE training even with millions of examples.

3. Experimental Evaluation

We apply JKSE to a *computer vision* task that allows us to demonstrate the two major claims that we made: robustness against large amounts of label noise, and computational efficiencies of training without iterated inference. We compare JKSE to an S-SVM based structured prediction method that has recently been shown to achieve state-of-the-art performance on similar object detection tasks (Blaschko & Lampert, 2008).

3.1. Object Localization in Images

Object localization by structured prediction can be modeled by taking the observations to be natural images and the labels to be bounding boxes of the target objects. As dataset we use the `UIUC Cars` set (Agarwal et al., 2004) using the *multiscale* part for training and the *singlescale* part for testing. This leaves us with 108 train images and 170 test images, which is close to the upper limit of what the S-SVM based reference implementation is able to handle in reasonable time. The images are transformed into a standard feature representation, and the *localization kernel* from (Blaschko & Lampert, 2008) is used as joint kernel function. It represents image regions by their *bag-of-visual-words* histogram over a four level spatial pyramid and combines the resulting histograms into kernel values by either a linear scalar product or a χ^2-kernel. The former choice allows fast MAP-inference using an integral-image trick, whereas the latter is generally accepted as a better kernel for computer vision tasks, but the results in a computationally expensive MAP-inference step that requires an exhaustive scan over all image locations. Because we also want to measure the performance of JKSE and S-SVM for training sets with noisy labels, we simulate label noise by artificially introducing label errors into the dataset by swapping bounding box coordinate between different training images. The percentage of swapped labels is a free parameter, r, that we vary between 0% (perfect labels) and 100% (random labels).

3.2. Model Selection

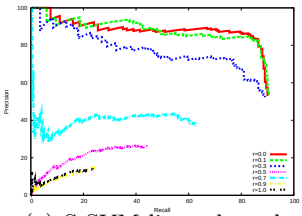
Training JKSE in the situation described takes less than a second of computation time. Prediction can be done in the order of seconds for the linear kernel function, whereas for the χ^2 kernel it takes a several minutes per image. The S-SVM has identical evaluation time, as it solves the same inference problem, but for training it requires many iterative solutions of the MAP estimate, each corresponding to an evaluation of the prediction function over many images. This procedure is only feasible for the linear kernel, and even with the fast integral image trick, the total training took approximately 5 hours. Within this time, on average close to 3,500 MAP calls were performed accounting for 97% of total running time.

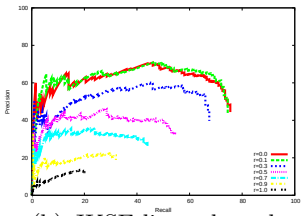
The long training time makes *model selection* by cross-validation impractical. Instead, we rely on a simplified criterion: we train one S-SVM for each parameter $C \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}\}$ on the full training set. The resulting weight vectors are treated as classifiers that we test on a held out set of 1050 smaller images that are also part of the original dataset, but could not be used for localization since they are pre-cropped. For testing, we adopt the C parameter that yields the largest *area under ROC curve* in this setup. For comparability, we follow the same procedure for JKSE to select $\nu \in \{0.05, 0.1, \ldots, 1.0\}$. Note, however, that JKSE would in fact be fast enough to perform full cross-validation, and this could be expected to improve the localization performance to certain extent.

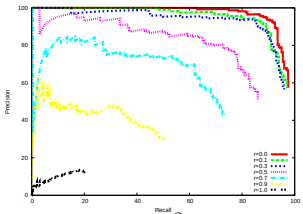
3.3. Results

Localization performance on the `UIUCcars` dataset is generally measured by precision-recall curves. The curves for S-SVM and JKSE are depicted in Figure 1. We also provide a table of *equal-error-rates*, i.e. the error rate where precision equal recall.

The left two plots show the results of S-SVM and JKSE with linear kernels: S-SVM achieves higher precision and recall than JKSE for noiseless data as well as when up to 30% of labels are scrambled. One can assume that S-SVM's Tikhonov regularization successfully compensates the disturbance introduced by this level of label errors. For label error rates of 50% and more, S-SVM performance takes a huge dive, and at 90% label errors, performance is basically random. Performance of JKSE with the same kernel function, though starting from a lower precision level, decreases more gradually when the amount of label errors increases. Even for 90% label noise, JKSE achieves better-than-random localization performance. We at-


(a) S-SVM linear kernel


(b) JKSE linear kernel


(c) JKSE χ^2-kernel

EER	S-SVM (lin.)	JKSE (lin.)	JKSE (χ^2)
$r = 0.0$	0.18	0.36	0.08
$r = 0.1$	0.16	0.35	0.11
$r = 0.3$	0.27	0.43	0.12
$r = 0.5$	0.76	0.60	0.14
$r = 0.7$	0.55	0.68	0.37
$r = 0.9$	0.92	0.79	0.62
$r = 1.0$	0.91	0.91	0.91

Figure 1. Precision-recall plots and *equal-error-rates* (EER) of localization performance of linear S-SVM, linear JKSE and χ^2-JKSE for different levels r of label noise. With an identical kernel, S-SVM dominates JKSE in terms of accuracy. However, with a non-linear kernel, JKSE performance surpasses S-SVM and it achieves non-trivial localization performance even with 90% label noise. See the text for further details.

tribute this to a success application of the ν-formalism: at this level $\nu = 0.95$ was chosen thereby treating a large amount of the training data as outliers.

The third plot shows results for JKSE with the χ^2 kernel function. Clearly, JKSE's localization accuracy is improved over the linear case and it also achieves better results than S-SVM. Adding up to 30% label noise hardly decreases the accuracy compared to perfect labels. For higher noise levels the performance decreases, however always staying clearly above the results for S-SVM. Even when 90% of training labels are randomly chosen, JKSE achieves a recall level of 50% and its precision lies above 40% over most of the plot. Clearly, the improved performance is a direct consequence of the use of a better kernel function. It can be assumed that S-SVM based localization would profit from using a χ^2 localization kernel as well. However, as mentioned above, training S-SVM with such a kernel is not computationally feasible with current techniques even for small training sets.

4. Conclusions

We have proposed *Joint Kernel Support Estimation* (JKSE), a technique that allows structured prediction with a training procedure that is as efficient and easy as ordinary SVM training. JKSE relies on a hybrid generative/discriminative view, modelling of the joint probability density of sample-label pairs, but doing so using a one-class SVM for margin-based support estimation. The resulting algorithm is on the one hand very fast, as no iterated inference has to performed at training time. On the other hand it is robust, because it does not assume that all label output labels for a sample are included in the training set, and it can handle mislabeled data through use of the ν-formalism.

From the point of pure classification performance with identical joint kernel functions, JKSE seems not to be as powerful as discriminative techniques. However, its strength lies in the fact that it can be applied in situation where CRF or S-SVM have problems or even fail: when training is not computationally feasible or only so by choosing a suboptimal kernel, and also, when the provided training data contain incomplete or incorrect labels. While few of the standard benchmark datasets in machine learning are of this type, this is a quite common situation in realistic pattern recognition problems, especially in domains where ground truth cannot easily be generated even by humans, *e.g.*, bioinformatics or medical imaging.

For future work, a further analysis of the assumptions and the properties of JKSE is required, in particular regarding the question: for which structured learning problems knowing the support of $p(x, y)$ is in fact sufficient to achieve good prediction performance?

References

Agarwal, S., Awan, A., & Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. *PAMI*, *26*, 1475–1490.

Blaschko, M. B., & Lampert, C. H. (2008). Learning to Localize Objects with Structured Output Regression. *ECCV*.

Bordes, A., Ertekin, S., Weston, J., & Bottou, L. (2005). Fast kernel classifiers with online and active learning. *JMLR*, *6*.

Collins, M. (2002). Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. *EMNLP*.

Hsieh, C.-J., Chang, K.-W., Lin, C.-J., Keerthi, S. S., & Sundararajan, S. (2008). A dual coordinate descent method for large-scale linear SVM. *ICML*.

Joachims, T. (2006). Training linear SVMs in linear time. *ACM KDD*.

Kashima, H., & Tsuboi, Y. (2004). Kernel-based discriminative learning algorithms for labeling sequences, trees, and graphs. *ICML*.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML*.

Lafferty, J., Zhu, X., & Liu, Y. (2004). Kernel conditional random fields: Representation and clique selection. *ICML*.

Lin, C.-J., Weng, R. C., & Keerthi, S. S. (2008). Trust region Newton method for logistic regression. *JMLR*, *9*.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, *13*, 1443–1471.

Shalev-Shwartz, S., Singer, Y., & Srebro, N. (2007). Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. *ICML*.

Taskar, B., Guestrin, C., & Koller, D. (2003). Max-margin Markov networks. *NIPS*.

Tsochantaridis, I., Joachims, T., Hofmann, T., & Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *JMLR*, *6*.