Journal of Machine Learning Research 17 (2016) 1-5 Submitted 10/15; Revised 3/16; Published 5/16

StructED : Risk Minimization in Structured Prediction

Yossi Adi ADIYOSS@CS.BIU.AC.IL
Joseph Keshet JOSEPH.KESHET@BIU.AC.IL
Department of Computer Science
Bar-Ilan University
Ramat Gan, 52900, Israel

Editor: Kevin Murphy

Abstract

Structured tasks are distinctive: each task has its own measure of performance, such as the word error rate in speech recognition, the BLEU score in machine translation, the NDCG score in information retrieval, or the intersection-over-union score in visual object segmentation. This paper presents StructED, a software package for learning structured prediction models with training methods that aimed at optimizing the task measure of performance. The package was written in Java and released under the MIT license. It can be downloaded from `http://adiyoss.github.io/StructED/`.

Keywords: structured prediction, structural SVM, CRF, direct loss minimization

1. Introduction

Ultimately the objective of discriminative training is to learn the model parameters so as to optimize a desired measure of performance. In binary classification, the objective is to find a prediction function that assigns a binary label to a single object, and minimizes the error rate (0-1 loss) on unseen data. In structured prediction, however, the goal is to predict a structured label (e.g., a graph, a sequence of words, a human pose, etc.), which often cannot be evaluated using the binary error rate, but rather with a task-specific measure of performance, generally called here *task loss*. There is a voluminous amount of work on training procedures for maximizing the BLEU score in machine translation, minimizing word/phone error rate in speech recognition or maximizing NDCG in information retrieval.

The most common approaches to learning parameters for structured prediction, namely structured perceptron, structural support vector machine (SSVM) and conditional random fields (CRF) do not directly minimize the task loss. The structured perceptron (Collins, 2002) solves a feasibility problem, which is independent of the task loss. The surrogate loss of SSVM (Joachims et al., 2005) is the structured hinge loss, a convex upper bound to the task loss, which is based on a generalization of the binary SVM hinge loss. However, while the binary SVM is strongly consistent, namely, converges to the error rate of the optimal linear predictor in the limit of infinite training data, the structured hinge loss is not consistent and does not converge to the expected task loss, i.e., the risk, of the optimal linear predictor even when the task loss is the 0-1 loss (McAllester, 2006). The objective of CRF is the log loss function, which is independent of the task loss (Lafferty et al., 2001).

Several structured prediction libraries already exist, such as CRF++ (Kudo, 2005), CRFsuite (Okazaki, 2007), Dlib (King, 2009), PyStruct (Müller and Behnke, 2014), and SVM-Struct (Joachims et al., 2009). All provide a general framework for training a model with either structured perceptron, CRF, or SSVM using several optimization techniques. Nevertheless, CRF++ and CRFsuite were not designed to support discriminative training with a task-specific evaluation metric, but rather use the binary error rate as their task loss function; whereas SVM-Struct, PyStruct, and Dlib support the training of structural SVM with a user-defined task loss function, but do not include, in their current phase, other training methods that are directly aimed at minimizing the task loss.

In this work we present StructED, a software package that is focused on training methods for structured prediction models that are aimed at minimizing a given task loss. The package provides several interfaces to define and implement a structured task, and allows the user to estimate the model parameters with several different training methods. The goal is not to provide several optimization alternatives to the structured hinge loss or the log loss as was already done in the previous packages, but rather to propose an implementation of a variety of surrogate loss functions that were designed to minimize the task loss and were theoretically motivated (McAllester and Keshet, 2011).

Note that most training methods require the task loss function to be decomposable in the size of the output label. Decomposable task loss functions are required in order to solve the loss-augmented inference that is used within the training procedure (Ranjbar et al., 2013), and evaluation metrics like intersection-over-union or word error rate which are not decomposable need to be approximated when utilized in SSVM, for example. Our package provides an implementation of methods (structured probit and orbit loss) that do not expect the task loss to be decomposable and can handle it directly without being approximated.

2. Structured Prediction and Risk Minimization

Consider a supervised learning setting with input instances $\boldsymbol{x} \in \mathcal{X}$ and target labels $\boldsymbol{y} \in \mathcal{Y}$, which refers to a set of objects with an internal structure. We assume a fixed mapping $\boldsymbol{\phi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ from the set of input objects and target labels to a real vector of length d, where we call the elements of this mapping *feature functions*. Also, consider a linear decoder with parameters $\boldsymbol{w} \in \mathbb{R}^d$, such that $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ is a good approximation to the true label of $\boldsymbol{x}$, as follows:

$$\hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \ \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) \tag{1}$$

Ideally, the learning algorithm finds $\boldsymbol{w}$ such that the prediction rule optimizes the expected desired *measure of preference* or *evaluation metric* on unseen data. We define the task loss function, $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}})$, to be a non-negative measure of error when predicting $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$ instead of $\boldsymbol{y}$ as the label of $\boldsymbol{x}$. Our goal is to find the parameter vector $\boldsymbol{w}$ that minimizes this function. When the desired evaluation metric is a utility function that needs to be maximized (like BLEU or NDCG), we define the task loss to be 1 minus the evaluation metric.

Our goal is to set $\boldsymbol{w}$ so as to minimize the expected task loss (i.e., risk) for predicting $\hat{\boldsymbol{y}}_{\boldsymbol{w}}$,

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{w}} \ \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\rho}[\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{w}}(\boldsymbol{x}))], \tag{2}$$

where the expectation is taken over pairs $(\boldsymbol{x}, \boldsymbol{y})$ drawn from an unknown probability distribution ρ. This objective function is hard to minimize directly. A common practice is to replace the task loss with a surrogate loss function, denoted $\bar{\ell}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y})$, which is easier to minimize; and to replace the expectation with a regularized average with respect to a set of training examples $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^m$, where each pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is drawn i.i.d from ρ:

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{w}} \ \frac{1}{m} \sum_{i=1}^{m} \bar{\ell}(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{y}_i) + \frac{\lambda}{2} \|\boldsymbol{w}\|^2, \tag{3}$$

where λ is a trade-off parameter between the loss term and the regularization. Different training algorithms are defined by different surrogate loss functions, e.g., the surrogate loss of max-margin Markov model (Taskar et al., 2003) is the structured hinge loss with the Hamming distance, whereas the one for CRF is the log loss function. These loss functions are convex in the model parameters, $\boldsymbol{w}$.

Recently, several works proposed different surrogate loss functions which are closer to the task loss in some sense: structured ramp-loss (McAllester and Keshet, 2011), structured probit loss (Keshet et al., 2011), and orbit loss (Karmon and Keshet, 2015). Direct loss minimization (McAllester et al., 2010) is a perceptron-like method of performing direct gradient descent on the risk (2) in the case where $\mathcal{Y}$ is discrete but $\mathcal{X}$ is continuous. All of the training objectives that were proposed in these works are non-convex functions in the model parameters, are strongly consistent and perform well in general.

3. Implementation

Implementation of a structured prediction task involves defining a set of feature functions. Given a trained model, the prediction is performed by finding the output label that maximizes the weighted sum of those feature functions according to (1). Such maximization involves the enumeration over all possible output labels, which is often intractable and handled by dynamic programming or by approximated inference techniques. As a result, and in contrast to packages for binary classification, here the user has to write code that implements the feature functions, the decoder and the evaluation metric.

Currently the implemented training methods are structured perceptron (SP), SSVM, CRF, structured passive-aggressive (SPA; Crammer et al., 2006), direct loss minimization (DLM), structured ramp loss (SRL), structured probit loss (SPL), and orbit loss (Karmon and Keshet, 2015).

The package exposes interfaces for task loss functions, feature extractors, decoding algorithms and training algorithms. In order to train a new model the user has to consider four interfaces, which correspond to: (i) a set of feature functions, $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})$; (ii) a task loss function[1], $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})$; (iii) a decoder to perform the inference in (1), as well as the loss-augmented inference (needed by SPA, SRL, DLM, SSVM); and (iv) a training algorithm. The user can either implement those interfaces or use any of the already implemented interfaces.

The objective in (3) is optimized using stochastic gradient descent (SGD) for both the convex (SSVM and CRF) and non-convex (SRL, SPL, orbit) surrogate losses. DLM cannot be expressed as a surrogate loss function in objective (3), and is optimized using SGD as

1. BLUE and NDCG are currently not implemented.

well. SP and SPA are online algorithms and are implemented as batch algorithms with averaging of the weight vectors as an online-to-batch conversion (Cesa-Bianchi et al., 2004).

4. Experiments

While the library focuses on different training methods for minimizing a given task loss, it is important to verify that the implemented algorithms run in an acceptable amount of time. We compared our implementation of SSVM optimized by SGD to the corresponding implementation in PyStruct on the MNIST data set of handwritten digits. We got similar accuracy and training times (StructED : 92.6%, 149 sec; PyStruct: 90.2%, 145 sec).

We conclude the paper by demonstrating the advantage of the package with a simple structured prediction task of automatic vowel duration measurement. In this task the input is a speech segment of arbitrary duration that contains a vowel between two consonants (e.g., *bat*, *taught*, etc.), and the goal is to accurately predict the duration of the vowel. This task is a required measurement in linguistic studies and is often done manually. The training data includes speech segments that are labeled with the vowel onset and offset times, denoted y_b and y_e, respectively. The task loss is defined as $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \max\{0, |y_b - \hat{y}_b| - \tau_b\} + \max\{0, |y_e - \hat{y}_e| - \tau_e\}$, where $\hat{y}_b$ and $\hat{y}_e$ are predicted vowel onset and offset times, respectively, and τ_b and τ_e are parameters (τ_b=10 msec and τ_e=15 msec). We had a training set of 90 examples, a validation set of 20 examples and a test set of 20 examples. We extracted 21 unique acoustic features every 5 msec, including the confidence of a frame-based phoneme classifier, the first and the second formants and other acoustic features. We trained all algorithms on this task and present their performance in terms of task loss (the lower the better) and training times in Table 1 (training parameters for each of the training methods can be found in the package's Examples folder).

Algorithm	Task loss [msec]	Time [sec]
SP	72.5	13
SSVM	72.0	13
CRF	70.5	31
SPA	69.0	12
SRL	64.5	25
SPL	64.5	617
DLM	63.5	24
Orbit loss	58.5	13

Table 1: Error rate and training times on vowel duration measurement task.

Results suggest that training methods, which are designed to minimize the task loss, lead to improved performance compared to SP, SSVM, CRF or SPA. It is also evident that these methods, except orbit loss, take at least twice as much time to train: DLM and SRL need two inference operations per training iteration and SPL needs hundreds to thousands of inferences per training iteration.

The implementation of the above example is straightforward and is given in the Examples section of the package website `http://adiyoss.github.io/StructED/` along with several other usage examples and further details.

References

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Trans. on Information Theory*, 50(9):2050–2057, 2004.

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

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

I. Joachims, T. Tsochantaridis, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

T. Joachims, T. Finley, and Chun-Nam Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

D. Karmon and J. Keshet. Risk minimization in structured prediction using orbit loss. 2015. (under submission).

J. Keshet, D. McAllester, and T. Hazan. PAC-Bayesian approach for minimization of phoneme error rate. In *Proceeding of ICASSP*, 2011.

D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

T. Kudo. CRF++: Yet another CRF toolkit, 2005.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289, 2001.

D. McAllester. Generalization bounds and consistency for structured labeling. In B. Schölkopf, A. Smola, B. Taskar, and S.V.N. Vishwanathan, editors, *Predicting Structured Data*, pages 247–262. MIT Press, 2006.

D. McAllester and J. Keshet. Generalization bounds and consistency for latent structural probit and ramp loss. In *Proceedings of NIPS (25)*, 2011.

D. McAllester, T. Hazan, and J. Keshet. Direct loss minimization for structured prediction. In *Proceeding of NIPS (24)*, 2010.

A. C. Müller and S. Behnke. PyStruct - learning structured prediction in python. *Journal of Machine Learning Research*, 15:2055–2060, 2014.

N. Okazaki. CRFsuite: a fast implementation of conditional random fields (CRFs), 2007.

M. Ranjbar, T. Lan, Y. Wang, S.N. Robinovitch, Z.-N. Li, and G. Mori. Optimizing nondecomposable loss functions in structured prediction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 35(4):911–924, 2013.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Proceedings of NIPS (17)*, 2003.