Handling Large Structural Costs in Neural Networks with Slack Rescaling

Heejin Choi [1] **Karl Stratos** [1]

Abstract

A cost-sensitive learning objective is often important for a neural model to achieve good performance in many structured problems. An ideal objective should penalize an incorrect label by its structural discrepancy and a correct label by zero. Since this is non-convex and non-differentiable, one typically turns to a convex surrogate loss in practice (Tsochantaridis et al., 2004). A widely used surrogate is *margin rescaling* which promotes the score of the true label to be greater than the loss-augmented score of the best label. However, for problems with large structural costs, this formulation is not faithful to the ideal objective: even when all structures are correctly classified, the loss may remain high. Consequently, the model wastefully updates parameters on correct instances during training. In this work, we focus on an alternative cost-sensitive objective called *slack rescaling*. Unlike margin rescaling, slack rescaling is invariant to the absolute values of structural costs and ignores labels that are already well-separated. This can be seen as a tighter approximation to the ideal objective. Inference with slack rescaling is in general intractable, but we adapt recent development of an efficient inference algorithm to the deep network. We evaluate our approach on neural graph-based dependency parsing and report promising results.

1. Introduction

In many structured problems, it is often important to carefully account for structural properties of the label space to achieve good performance. This need arises in neural network modeling across many disciplines such as natural language processing (NLP), computer vision, and speech recognition (Bakir et al., 2007). As a concrete example, in the graph-based neural dependency parser of Kiperwasser & Goldberg (2016) which is trained using structured hinge loss, the accuracy of the parser plunges from 91 to 79.4 when the training objective disregard structural costs.

Structured prediction concerns many interesting real world problems where there exists an internal structure within its output(Bakir et al., 2007). A few examples are dependency parsing in natural language processing, segmentation task in computer vision, and speech recognition task. In such examples, output label is consists of micro-labels, and its relationship within the micro-label is expressed by a structure, for instance, a tree or a sequence.

In the era of deep networks due to surprising success in various fields (Krizhevsky et al., 2012)(Goodfellow et al., 2014), building a deep structured model to deal with the structured task is an important task, and it is well motivated by the success in many fields(Chen et al., 2015; Schwing & Urtasun, 2015).

Incorporating the structure in the model is the essence of the structured models. One of the most well-studied approaches is to incorporate the structure into the loss function. This is called cost-sensitive learning. Compare to zero one flat loss where the loss is one if predicted label is different from the true label or zero if correct, in cost-sensitive learning, different loss incurs respect to the structure of the labels, i.e. the loss is calculated by modeling how the labels are different. (Tsochantaridis et al., 2004) introduced convex surrogate loss for the cost-sensitive losses, called margin rescaling and slack rescaling.

Two formulations differ how the most violating label is calculated. While margin rescaling criteria is a sum of the feature score and the label score, the slack rescaling criteria is a product between the two. This enables margin rescaling to be computationally efficient since the interplay between the two scores is not global, and can be decomposed respect to the substructure. However, margin rescaling can be dominated by one score alone, feature score or label score. This is problematic for a large structured problem since even for the label which is well separated, it can be considered as a violating label if label score is high. Therefore, the constraints of the margin rescaling is hard to be satisfied. On the other hand, criteria of slack rescaling is a product of the two scores, and it is difficult for the label to

[1]TTIC, Chicago, USA. Correspondence to: Heejin Choi <heejincs@ttic.edu>.

Proceedings of the ICML 17 Workshop on Deep Structured Prediction, Sydney, Australia, PMLR 70, 2017.

be considered as the most violating label if it has only one high score. Specifically, labels with a margin larger than a constant margin of one cannot be the most violating label. However, the interplay between the two score is global, and does not decompose with respect to the substructure. Thus, loss augmented inference in slack rescaling formulation is computationally intensive, and intractable for large structures.

To reduce the computational burden of loss augmented inference, Sarawagi & Gupta (2008) introduced a method utilizing margin rescaling inference as an oracle, and by calling this oracle iteratively to obtain an approximated slack rescaling argmax label. However, the drawback of the approach that makes the approach impractical is that it involves binary search over scalar, which can take tens of iterations. A dynamic programming approach is proposed In (Bauer et al., 2014) for a simple structures. Choi et al. (2016) improves upon (Sarawagi & Gupta, 2008) for efficient optimization and extends to finding exact optimum with a small modification of the margin rescaling oracle. In this paper, we show that the iterative oracle apporach can be efficiently used in deep network. We demonstrate that in practice approximated slack rescaling can be done almost as same computational complexity as margin rescaling, calling oracle only 2 to 3 times in average in the later stage of the optimization.

We mainly consider the dependency parsing problem in NLP investigated in (Kiperwasser & Goldberg, 2016). We demonstrate that this slack rescaling approach can be utilized in deep network efficiently to achieve a slighter higher performance. This paper is organized as follows: Firstly, we review two convex surrogate losses, margin rescaling and slack rescaling, and compare the pros and the cons of the two approaches. Secondly, we briefly describe how we adapt slack rescaling to the deep network. Thirdly, we visit our main focused application of dependency parsing. Fourthly, we show the empirical evaluation of our approach. We conclude with discussions.

2. Two Surrogate Losses

In this section, we review the two surrogate losses mostly based on (Choi et al., 2016). Structural SVM(Tsochantaridis et al., 2004; Taskar et al., 2003) with a margin rescaling formulation is defined as

$$\min_{w,\xi} \frac{C}{2}\|w\|_2^2 + \frac{1}{n}\sum_i \xi_i \tag{1}$$
$$\text{s.t. } f_i(y_i) - f_i(y) \geq \Delta(y, y_i) - \xi_i \qquad \forall i, y \in \mathcal{Y}$$
$$\xi_i \geq 0 \qquad \forall i$$

where C is the regularization constant. The slack rescaling formulation is

$$\min_{w,\xi} \frac{C}{2}\|w\|_2^2 + \frac{1}{n}\sum_i \xi_i \tag{2}$$
$$\text{s.t. } f_i(y_i) - f_i(y) \geq 1 - \frac{\xi_i}{\Delta(y, y_i)} \qquad \forall i, y \in \mathcal{Y}$$
$$\xi_i \geq 0 \qquad \forall i$$

The important property of the surrogate losses are it is a convex upper bound of the task loss $\Delta(y, y_i)$. Too see the difference of the two directly, let $m(y) = f(y_i) - f(y)$ be the margin of label of y_i for a certain instance. Then, the loss occurred by label y in both formulations are

$$\text{Margin: } \xi_M(y) = \Delta(y, y_i) - m(y) \tag{3}$$
$$\text{Slack: } \xi_S(y) = \Delta(y, y_i)\,(1 - m(y)) \tag{4}$$

And the empirical loss for the instance is defined as max over the label

$$\text{Margin: } \xi_M = \max_y \xi_M(y) \tag{5}$$
$$\text{Slack: } \xi_S = \max_y \xi_S(y) \tag{6}$$

Since $\xi_M(y_i) = \xi_S(y_i) = 0$, the loss is non-negative. Two main differences of the two surrogate losses are the tightness and the constraint set of label $\mathcal{Y}$. To be specific, slack rescaling is tighter to the task loss $\Delta(y, y_i)$ than margin rescaling when the instance is on the correct side of the classification boundary, $m(y) > 0$, and the other way when the instance is on the wrong side of the classification boundary . Also, slack rescaling considers a much smaller set of constraint set of $\mathcal{Y}$ disregarding labels which are already well separated.

The first argument can be shown easily by showing following, given $L(y, y_i) \geq 1$

$$\Delta(y, y_i) \leq \xi_S(y) \leq \xi_M(y) \qquad \text{if } m(y) > 0.$$
$$\Delta(y, y_i) \leq \xi_M(y) \leq \xi_S(y) \qquad \text{if } m(y) < 0.$$

Since the deep network is a powerful model, by increasing the power of the deep network, we can expect all the training margin to be well classified. Then, the slack rescaling loss is much closer to the target loss than margin rescaling, especially for large structures with a large target loss $\max_{y,y_i} \Delta(y, y_i)$. Therefore, slack rescaling more preferable objective than the margin rescaling for the deep network.

The other difference is the label constraint set. Since $\xi_M(y_i) = \xi_S(y_i) = 0$ and objective function takes maximum over the labels, we can disregard y such that $\xi_m(y) < 0$ or $\xi_s(y) < 0$ not changing the ξ_i. Writing down this label set explicitly,

$$\text{Margin: } \tilde{\mathcal{Y}}_m = \{y | \Delta(y, y_i) < m(y)\} \tag{7}$$
$$\text{Slack: } \tilde{\mathcal{Y}}_s = \{y | 1 < m(y)\} \tag{8}$$

We can immediately see that $\tilde{\mathcal{Y}}_m \subseteq \tilde{\mathcal{Y}}_s$, and while slack rescaling removes all well separated labels from the consideration, margin rescaling is more conservative by removing the labels by only removing labels separated with a large margin. This implies that slack rescaling has less constraints to satisfy, and in turn easy to satisfy by removing unnecessary labels, and margin rescaling is hard to satisfy.

However, the main caveat for the slack rescaling is the computational cost. To see this, denote $h(y)$ as score function and error function as $g(y)$. For margin rescaling $h(y) = f(y) - f(y_i)$ and for slack rescaling $h(y) = 1 + f(y) - f(y_i)$, and $g(y) = \Delta(y, y_i)$ for both. Then, the slack rescaling objective is the product between the two, $h(y)g(y)$ and for the margin rescaling it is the sum, $h(y) + g(y)$. Since often for tractable structural problems, label decomposes over the substructure, and finding the maximum label in margin rescaling can benefit from the decomposition since the objective function also decomposes over the substructure. However, slack rescaling cannot benefit from the decomposition since the objective function is the product between the two, and it does not decompose over the substructure. Therefore, it is often intractable to do max slack rescaling label search for a large structure. We discuss this problem in the next section.

3. Adaptation to Deep Network

Here we describe the method to learn with slack rescaling formulation in the deep network. Moving from the margin rescaling formulation to slack rescaling formulation can be straightforward. This method can be widely applicable to the models that is using margin rescaling by changing the loss from (5) to (6). Since the update is mostly done via SGD, we only need to change argmax label of (5) to that of (6). This is the approach in (Choi et al., 2016) described in Algorithm 2. We can scale the label error function by λ and find the argmax label in margin rescaling formulation iteratively. Method to find the next λ depends on the algorithm. We use variant of Bisecting search in (Choi et al., 2016), Then, we update the network according to the maximum label in slack rescaling formulation among the labels we found varying λ noted as S in the Algorithm 2.

3.1. Expending label set for generalization

In the previous section, we briefly described that slack rescaling deals with the smaller label set by removing well-separated labels from the consideration. While there is a benefit to this, the small label set might result failing to find the most violating label, resulting zero gradients for instances in the early stage of the learning. This encourages to expand the label set so that deep net can learn more from the data in case of the zero gradients. We accomplish this by first finding slack rescaling label, and in the case when there is no violating slack rescaling label, we learn with the margin rescaling argmax label. This can be viewed as expending required margin. We call this procedure slack-margin rescaling.

Algorithm 1 Slack rescaling argmax search

$S \leftarrow \emptyset, \lambda \leftarrow 0$
for $t = 1, ...,$ **do**
  $\lambda \leftarrow f(S, \lambda)$
  $y_t = \arg\max h(y) + \lambda g(y)$
  if $y_t \in S$ **then**
    return $\arg\max_{y \in S} h(y)g(y)$
  end if
  $S \leftarrow S \cup \{y_t\}$
end for

Algorithm 2 Slack-margin rescaling argmax search

$y = \arg\max h(y)g(y)$
if $y = y_i$ **then**
  return $\arg\max h(y) + g(y)$
else
  return y
end if

4. Experiments

We experimented with the-state-of-the-art neural dependency parser described in (Kiperwasser & Goldberg, 2016). It uses margin scaling formulation. In the problem, exploiting the structure is found crucial in for the high performance as described in the introduction. For the argmax search algorithm, we used a variant of the bisecting search of (Choi et al., 2016), and by efficiently making use of the label cache, we only need to search argmax margin scaling label less than 3 times in the later epoch. We trained for a full dataset for 30 epochs, and report result on the dev set.

5. Discussion

We adapted slack rescaling approach to the deep neural network to achieve higher performance than the margin rescaling. We showed that a careful choosing of λ can lead to efficient slack rescaling so that it can be deployed in the large deep network. This implies that slack rescaling can be widely used for other structural tasks where the deep network is used.

Algorithm	Margin	Slack	Slack-Margin
Labeled attachment	91.0	90.7	91.2
Unlabeled attachment	92.7	92.4	92.9

Table 1. : Results on the Treebank dataset

References

Bakir, Gükhan H., Hofmann, Thomas, Schölkopf, Bernhard, Smola, Alexander J., Taskar, Ben, and Vishwanathan, S. V. N. *Predicting Structured Data*. The MIT Press, 2007.

Bauer, Alexander, Gornitz, N, Biegler, Franziska, Muller, K-R, and Kloft, Marius. Efficient algorithms for exact inference in sequence labeling svms. *Neural Networks and Learning Systems, IEEE Transactions on*, 25 (5):870–881, 2014.

Chen, Liang-Chieh, Schwing, Alexander, Yuille, Alan, and Urtasun, Raquel. Learning deep structured models. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1785–1794, 2015.

Choi, Heejin, Meshi, Ofer, and Srebro, Nathan. Fast and scalable structural svm with slack rescaling. In *Artificial Intelligence and Statistics*, pp. 667–675, 2016.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Kiperwasser, Eliyahu and Goldberg, Yoav. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327, 2016.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Sarawagi, Sunita and Gupta, Rahul. Accurate max-margin training for structured output spaces. In *Proceedings of the 25th international conference on Machine learning*, pp. 888–895. ACM, 2008.

Schwing, Alexander G and Urtasun, Raquel. Fully connected deep structured networks. *arXiv preprint arXiv:1503.02351*, 2015.

Taskar, B., Guestrin, C., and Koller, D. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*. MIT Press, 2003.

Tsochantaridis, Ioannis, Hofmann, Thomas, Joachims, Thorsten, and Altun, Yasemin. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 104. ACM, 2004.