Summary for Structured Prediction Cascades

Ruotian Luo

1 Structured prediction

Structured prediction is a supervised machine learning technique that involves predicting structured objects, rather than scalar discrete or real values(which correspond to classifications and regressions)[1].

Structured data is data that consists of several parts, and not only the parts themselves contain information, but also the way in which the parts belong together[2].

Under this definition, many tasks can be regarded as a structured prediction problem, like part-of-speech tagging, machine translation, parsing in NLP, and segmentation, pose estimation in computer vision. Thus, a fast and accurate structured prediction algorithm is appealing because it can help many tasks in different fields.

2 Motivation

Approximation vs. computation Usually increasing the complexity of the model can achieve better prediction performance, however, with the cost of computation time. For example, for OCR, a first order conditional random field (CRF)[3] is fast to evaluate but may not be an accurate model while a fifth order model is more accurate, but is more expensive in both learning and prediction.

The author borrows the coarse-to-fine idea from [4], and propose to learn a cascade of structured models of increasing complexity that reduce later layers' search space so that exact inference can be done efficiently for later complex models.

The following sections will introduce the model they use for each layer of the cascade and how they do the filtering and how they train the cascade model.

3 Linear model for structured prediction

Notation: Input space $\mathcal{X}$; output space $\mathcal{Y}$; joint distribution of X and Y, $D(X,Y)$; training set $S = \{\langle x^1, y^1\rangle, \dots, \langle x^n, y^n\rangle\}$ of n IID random samples drawn from $D(X,Y)$; non-negative loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$.

The standard supervised learning task is to learn a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected loss $\mathbb{E}_D[\mathcal{L}(h(x), y)]$.

In this paper, they consider y as a sequence of discrete variables with length l, $y = \{y_1, \cdots, y_l\}$, where $y_i \in \{1, \dots, K\}$. For example, in OCR, y is the whole word(a sequence of characters),

3.1 Markov model

In this paper, they use a probabilistic graphical model to define the conditional distribution $P(y|x)$. With the graphical model, instead of exhaustively searching the solution over $\mathcal{Y}$ which is exponential in l, one can do tractable inference.

The graphical model they use in this paper is Markov network, as in CRF[3] and M3N[5]. The Markov network defines the conditional distribution Y given X as:

$$P(y|x) \propto \prod_{c \in \mathcal{C}} \psi_c(x, y_c) \tag{1}$$

The posterior probability is proportional to the product of factors; each factor is defined on a subset of x and y, which are called cliques. y_c denotes the subset of variables involved in clique c, $y_c \triangleq \{y_i | i \in c\}$. And $\mathcal{C}$ is the set of cliques in the network.

The authors use log-linear model, so

$$P(y|x) \propto \prod_{c \in \mathcal{C}} e^{\phi_c(x, y_c)} \tag{2}$$

where $\phi_c(x, y_c)$ is a linear function:

$$\phi_c(x, y_c) = \mathbf{w}^\intercal \mathbf{f}_c(x, y_c)$$

For simplicity, they define $\mathbf{f}(x, y) = \sum_c f_c(x, y_c)$ and $\theta_x(y) = \mathbf{w}^\intercal \mathbf{f}(x, y)$. Then,

$$P(y|x) \propto e^{\mathbf{w}^\intercal \mathbf{f}(x,y)} = e^{\theta_x(y)}$$

The hypothesis class for structured prediction in this model is defined as

$$h_w(x) = \operatorname*{argmax}_y P(y|x) = \operatorname{argmax} \theta_x(y)$$

. We can treat $\theta_x(y)$ as a score function for a (x, y) pair.

Here is an example of first order linear-chain markov model. It has cliques over $\{y_i, y_{i+1}, x\}$, and thus $\theta_x(y) = \sum_i \mathbf{w}^\intercal f_i(y_i, y_{i+1}, x)$.

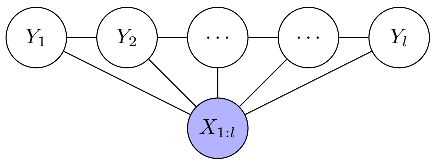

Figure 1: First order linear-chain markov model

In the next section, I will introduce how the authors use the above linear model for filtering, and how they train the model.

4 Structured cascades

The cascade model is a stack of linear models described as in Sec.3 with increasing complexity. Each layer has its own set of $\mathbf{w}^i$, $\mathbf{f}^i$ and $\mathcal{C}^i$. Instead of predicting a single output, each layer now prunes easy incorrect clique assignments accroding to max-marginals and provide a refined valid assignments set to higher layer.

4.1 Inference

The filtering is a feed-forward process. Each layer tries to reduce the set of feasible assignments to y and passes it to the next layer until the last layer does prediction.

Algorithm 1 Structured cascades inference

```
1: Input $x$, $L$ is the number of layers
2: $\mathcal{V}^1 = \mathcal{Y}$
3: for $i$=1, $L$-1 do
4:     Get filtered $\mathcal{V}^{i'}$ from $\mathcal{V}^i$ using the i-th layer model.
5:     Generate $\mathcal{V}^{i+1}$ for next layer.
6: end for
7: Get $y^* = \operatorname{argmax} \theta_x^L(y)$
```

The way they do filtering is by running inference to compute max-marginals for each clique and eliminate low-scoring clique assignments. More details in the following section.

4.2 Filtering in each layer

First we define max-marginal as $\theta_x^*(y_c)$ to be the maximum score of any output y that is consistent with the assignment y_c

$$\theta_x^*(y_c) = \max_{y' \in \mathcal{Y}} \{\theta_x(y') : y'_c = y_c\}$$

All the max-marginals can be calculated at once by max-sum.

At each layer i of the cascade, the model receives a set of possible clique assignments $\mathcal{V}_c^i$ for each clique from the previous layer. (For the first layer, $\mathcal{V}_c^1 = \mathcal{Y}_c$). The model in current layer further filters $\mathcal{V}_c^i$ to $\mathcal{V}_c^{i'}$ and generates a set of possible clique assignments $\mathcal{V}_c^{i+1}$ which are passed as input to the next level. At the end of the cascade, the most complex model chooses a single prediction as usual, but it only needs to consider the clique assignments that have not already been pruned.

Pruning is done by using clique max-marginals. At each layer, we can calculate max-marginals for all $y_c \in \mathcal{V}_c^i$ by max-sum (corresponding to max-product in CRF). If the max-marginal of y_c is less than some threshold t, then y_c is pruned; so $\mathcal{V}_c^{i'} = \{\theta_x^*(y_c) \geq t | y_c \in \mathcal{V}_c^i\}$. Then the corresponding set of valid cliques of the layer $i+1$ is defined as:

$$\mathcal{V}_c^{i+1} = \{y_c \in \mathcal{Y}_c | c \in \mathcal{C}^{i+1}, \forall c' \in \mathcal{C}^i, c' \subseteq c, y_{c'} \in \mathcal{V}_c^{i'}\} \tag{3}$$

This is the set of clique assignments y_c in the layer $i+1$ for which all consistent clique assignments $y_{c'}$ for subcliques $c' \in \mathcal{C}_i$ have not been pruned by the i_{th} model.

This filter rules have two properties:

Lemma 4.1. *If $\theta_x(y) > t$, then $\forall c, \theta_x^*(y_c) > t$.*

Lemma 4.2. *If $\max_{y'} \theta_x(y') > t$, then $\exists y, \forall c, \theta_x^*(y_c) > t$.*

The first property gives a sufficient condition that an output y is not filtered, which helps define the filter loss later. The second property guarantees if the threshold is less than maximum score, then it's guaranteed to have some assignment y which is not pruned. (These properties don't hold for sum-product marginals in CRF)

Further, the author defines a data-dependent threshold which is a convex combination of max score and the mean of max marginals.

$$t_x(\alpha) = \alpha\theta_x^* + (1-\alpha)\frac{1}{|\mathcal{V}|}\sum_{c \in \mathcal{C}, y_c \in V_c} \theta_x^*(y_c) \tag{4}$$

where $\theta_x^* = \max_y \theta_x(y)$, $\mathcal{V} = \cup_{c \in \mathcal{C}} \mathcal{V}_c$, and $\alpha \in [0, 1)$.

They derive this formula to approximate quantile threshold. Similar to quantile, the threshold has the same scale as the scores; unlike quantile, it's faster to get, and also continuous and convex, which are good for training.

4.3 Learning

The cascade is learned layer-by-layer. In general, for each layer, the more you prune, the more efficient the inference is, but also the correct answers would be more likely to be pruned. So, the trade-off here is between accuracy and efficiency. To quantify this trade-off, authors define two loss functions.

Definition 4.1. *Filtering loss.* $\mathcal{L}_f(y, \theta_x) = \mathbf{1}[\theta_x(y) \leq t_x(\alpha)]$

The expected filtering loss $\mathbb{E}[\mathcal{L}_f(Y, \theta_X)]$ is the proportion of ground truth output being eliminated. If filtering loss is lower, that means the ground truth is less likely to be pruned.

Definition 4.2. *Efficiency loss.* $\mathcal{L}_e(y, \theta_x) = \frac{1}{|\mathcal{V}|}\sum_{c \in \mathcal{C}, y_c \in \mathcal{V}_c} \mathbf{1}[\theta_x^*(y_c) \geq t_x(\alpha)]$

The expected efficiency loss $\mathbb{E}[\mathcal{L}_e(Y, \theta_X)]$ is the mean of remaining cliques proportion. Lower efficiency loss means more clique assignments are pruned.

The filters are learned by minimizing the efficiency loss while constraining the filtering loss to be below a desired tolerance ϵ.

$$\min_{\mathbf{w}, \alpha} \mathbb{E}[\mathcal{L}_e(Y, \theta_X)] \text{ s.t. } \mathbb{E}[\mathcal{L}_f(Y, \theta_X)] \leq \epsilon \tag{5}$$

Objective (5) is solved by a two-step procedure: learn a $\mathbf{w}$ for a given set of α; choose the α which optimize Eq. (5).

Learn the w given α. The author defines a convex surrogate loss for filtering loss which is similar to max-margin loss for prediction defined in [5].

$$\min_{\mathbf{w}} L(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_i H(\mathbf{w}; (x^i, y^i)) \tag{6}$$

, where $H(\mathbf{w}; (x^i, y^i)) = \max\{0, 1 + t_{x^i}(\alpha) - \mathbf{w}^T\mathbf{f}(x^i, y^i)\}$.

In practice, they use stochastic sub-gradient descent to optimize Eq. (6).

Choose α. After they get the $\mathbf{w}$s for different αs, they fix the $\mathbf{w}$s and search for an α which optimizes Eq. (5) on dev set. The final α doesn't have to be in the candidate α set. They claim this approach can get better result.

Algorithm 2 Structured cascades training

```
1: $L$ is the number of layers, $\{\epsilon_i\}$ is the loss threshold for each layer, $\mathbf{A}$ is the candidate set for $\alpha$.
2: for $i$=1, $L$-1 do
3:     for $\alpha \in \mathbf{A}$ do
4:         Learn $\mathbf{w}_i$ using stochastic sub-gradient descent on the objective Eq. (6).
5:     end for
6:     Find the $(\mathbf{w}_i, \alpha)$ pair ($\mathbf{w}_i$s are the previous step and are fixed) which gets the highest efficiency and satisfies
       the filtering loss less than $\epsilon_i$.
7: end for
8: Use Structured Perceptron to train $\mathbf{w}_L$.
```

4.4 Generalization bound

The authors provide theoretical justification for the definitions of the loss functions $\mathcal{L}_e$ and $\mathcal{L}_f$ and the structured cascade objective.

Theorem 4.1.

$$\mathbb{E}[\mathcal{L}_f(Y, \theta_X)] \le \hat{\mathbb{E}}[\phi_f(Y, \theta_X)] + O\left(\frac{m\sqrt{l}B}{\gamma\sqrt{n}}\right) + \sqrt{\frac{8\ln(2/\delta)}{n}}, \tag{7}$$

This theorem says: if the model works well on a n sample training data, it is likely that the performance of the model on unseen test data will not be too much worse as n gets large.

5 Experiments

5.1 Hand-writing recognition

This experiment proves that higher-order model can achieve better word accuracy and character accuracy, which convinces us the usage of higher-order model is necessary. Meanwhile, the filter mechanism is able to prune a large portion of the states: $26^2 \to 123.8$ at the second layer, $26^3 \to 88.6$ at the third layer, $26^4 \to 5.4$ at the fourth layer (the numbers on the right is the average number of feasible states of each clique). In this way, the higher-order is not only accurate but also efficient.

5.2 Part-of-speech Tagging

In this task, Structured Cascades(SC) model is compared to a filter version CRF and an unfiltered Structured Perceptron(SP) model. Relative to unfiltered SP inference and filtered CRF inference, SC model can achieve 100x and 2.6x increased inference efficiency, without harming the performance.

Authors also compare their trained filter to a naive filter baseline in which only POS tags associated with a given word in the training set were searched during inference. It's shown that the SC is more efficient with similar filter loss.

To better investigate the efficiency vs. filtering accuracy trade-off of different models, they plot the $\mathcal{L}_e - \mathcal{L}_f$ curve of when CRF, SC, SP work as the first layer filter and the second layer filter. In nutshell, SC has better efficiency with the same accuracy than other methods in both cases.

6 Conclusion

Authors propose a cascade model for structured prediction, a way to use complex model with less computation cost. They also propose the training method which can explicitly balance accuracy and efficiency. Finally, they evaluated their method on several datasets.

References

[1] Gökhan Bakir. *Predicting structured data.* MIT press, 2007.

[2] Sebastian Nowozin and Christoph H Lampert. Structured prediction and learning in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3-4):3–4, 2011.

[3] John Lafferty, Andrew McCallum, Fernando Pereira, et al. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning, ICML*, volume 1, pages 282–289, 2001.

[4] Paul Viola and Michael Jones. Robust real-time object detection. *International Journal of Computer Vision*, 4(34–47), 2001.

[5] BTCGD Roller. Max-margin markov networks. *Advances in neural information processing systems*, 16:25, 2004.