Proximal Deep Structured Models

Shenlong Wang, Sanja Fidler, & Raquel Urtasun

University of Toronto

NIPS 2016
Presenter: Jack Lanchantin
1 Intro

2 Deep Structured Prediction

3 Proximal Methods
   - Proximal Methods
   - Proximal Deep Structured Models
   - Solving Proximal Deep Structured Models

4 Experiments and Results
   - Image Denoising/Depth Refinement
   - Optical Flow
Structured Prediction

- Many problems in real-world applications involve predicting a collection of random variables that are statistically related.
- Graphical models have been widely exploited to encode these interactions, but they are shallow and only a log linear combination of hand-crafted features is learned.
Deep structured models attempt to learn complex features by taking into account the dependencies between the output variables. A variety of methods have been developed in the context of predicting discrete outputs.

However, little to no attention has been given to deep structured models with continuous valued output variables. One of the main reasons is that inference is much less well studied, and very few solutions exist.
Continuous-Valued Structured Prediction

Given input $x \in \mathcal{X}$, let $y = (y_1, \ldots, y_n)$ be the set of random variables we want to predict. The output space is a product space of all the elements $y \in \mathcal{Y} = \prod_{i=1}^{N} \mathcal{Y}_i$, $\mathcal{Y}_i \subset \mathbb{R}$.

$E(x, y; w)$ is an energy function which encodes the problem:

$$E(x, y; w) = \sum_i f_i(y_i, x; w_u) + \sum_{\alpha} f_\alpha(y_\alpha, x, w_\alpha)$$

(1)

where $f_i(y_i : x, w_u) : \mathcal{Y}_i \times \mathcal{X} \rightarrow \mathbb{R}$ is a function that depends on a single variable (i.e. unary term) and $f_\alpha(y_i) : \mathcal{Y}_\alpha \times \mathcal{X} \rightarrow \mathbb{R}$ depends on a subset of variables $y_\alpha = (y_i)_{i \in \alpha}$ defined on a domain $\mathcal{Y}_\alpha \subset \mathcal{Y}$. 

Shenlong Wang, Sanja Fidler, & Raquel Urtasun

NIPS 2016 Presenter: Jack Lanchantin / 26
$E(y) = -\sum_{i} f_i(y_i) - \sum_{i,j \in E} f(y_i, y_j) - \sum_{\alpha} f_{\alpha}(y_{\alpha})$

- **Unaries**
- **Pairwise**
- **High-order**
Continuous-Valued Structured Prediction

- Given an input $x$, inference aims at finding the best configuration by minimizing the energy function:

$$y^* = \arg\min_{y \in Y} \sum_i f_{\alpha_i}(y_i; x, w_u) + \sum_{\alpha} f_{\alpha}(y_{\alpha}, x, w_{\alpha}) \quad (2)$$

- Finding the best scoring configuration $y^*$ is equivalent to maximizing the posteriori distribution:

$$p(y|x; w) = \frac{1}{Z(x; w)} \exp(-E(x, y|w)) \quad (3)$$
Performing inference in MRFs with continuous variables involves solving a challenging numerical optimization problem. If certain conditions are satisfied, inference is often tackled by a group of algorithms called proximal methods. In this paper, they use proximal methods and show that it results in a particular type of recurrent net.
Outline

1 Intro

2 Deep Structured Prediction

3 Proximal Methods
   - Proximal Methods
   - Proximal Deep Structured Models
   - Solving Proximal Deep Structured Models

4 Experiments and Results
   - Image Denoising/Depth Refinement
   - Optical Flow
The proximal operator $\text{prox}_f(x_0) : \mathbb{R} \to \mathbb{R}$ of a function is defined as:

$$\text{prox}_f(x_0) = \arg\min_y (y - x_0)^2 + f(y)$$

(4)

This involves solving a convex optimization problem, but usually there is a closed-form solution.
Outline

1 Intro

2 Deep Structured Prediction

3 Proximal Methods
   - Proximal Methods
   - Proximal Deep Structured Models
   - Solving Proximal Deep Structured Models

4 Experiments and Results
   - Image Denoising/Depth Refinement
   - Optical Flow
In order to apply proximal algorithms to tackle the inference problem defined in Eq. (2), we require the energy functions $f_i$ and $f_\alpha$ to satisfy the following conditions:

1. There exist functions $h_i$ and $g_i$ s.t. $f_i(y_i, x; w) = g_i(y_i, h_i(x, w))$, where $g_i$ is a distance function.
2. There exists a closed-form proximal operator for $g_i(y_i, h_i(x, w))$ wrt $y_i$.
3. There exist functions $h_\alpha$ and $g_\alpha$ s.t. $f_\alpha(y_\alpha, x; w)$ can be re-written as $f_\alpha(y_\alpha, x; w) = h_\alpha(x; w)g_\alpha(w_\alpha^T y_\alpha)$.
4. There exists a proximal operator for $g_\alpha()$.
If our potential functions satisfy the conditions above, we can rewrite our objective function as follows:

\[
E(x, y; w) = \sum_i g_i(y_i, h_i(x; w)) + \sum_\alpha h_\alpha(x; w)g_\alpha(w_\alpha^T y_\alpha)
\]  

(5)
The general idea of primal dual solvers is to introduce auxiliary variables $z$ to decompose the high order terms. We can then minimize $z$ and $y$ alternately through computing their proximal operator:

$$
\min_{y \in \mathcal{Y}} \max_{z \in \mathcal{Z}} \sum_i g_i(y_i, h_i(x; w)) - \sum_{\alpha} h_{\alpha}(x, w)g_\alpha^*(w_\alpha^T y_\alpha) + \sum_{\alpha} h_{\alpha}(x, w)\langle w_\alpha^T y_\alpha, z_\alpha \rangle
$$

where $g_\alpha^*$ is the convex conjugate of $g^*$
Outline

1. Intro

2. Deep Structured Prediction

3. Proximal Methods
   - Proximal Methods
   - Proximal Deep Structured Models
   - Solving Proximal Deep Structured Models

4. Experiments and Results
   - Image Denoising/Depth Refinement
   - Optical Flow
The primal-dual method solves the problem in Eq.(6) by iterating the following steps: (i) fix $y$ and minimize the energy wrt $z$; (ii) fix $z$ and minimize the energy wrt $y$; (iii) conduct a Nesterov extrapolation gradient step:

$$
\begin{align*}
    z_{\alpha}^{(t+1)} &= \text{prox}_{g_{\alpha}}^*(z_{\alpha}^{(t)} + \frac{\sigma_p}{h_{\alpha}(x;w)} w_T \bar{y}_{\alpha}^{(t)}) \\
    y_i^{(t+1)} &= \text{prox}_{g_i,h_i(x,w)}(y_i^{(t)} - \frac{\sigma_T}{h_{\alpha}(x;w)} w^T \cdot_i z^{(t+1)}) \\
    y_i^{(t+1)} &= y_i^{(t+1)} + \sigma_{ex}(y_i^{(t+1)} - y_i^{(t)})
\end{align*}
$$

where $y^{(t)}$ is the solution at the $t$-th iteration, $z^{(t)}$ is an auxiliary variable and $h(x, w_u)$ is the deep unary network.
\[
\begin{align*}
    z_{\alpha}^{(t+1)} &= \text{prox}_{g_{\alpha}^*}(z_{\alpha}^{(t)}) + \frac{\sigma_{\rho}}{h_{\alpha}(x;w)}W_{\alpha}^T \bar{y}_{\alpha}^{(t)} \\
    y_{i}^{(t+1)} &= \text{prox}_{g_{i}, h_{i}(x,w)}(y_{i}^{(t)}) - \frac{\sigma_{\tau}}{h_{\alpha}(x;w)}W_{*,i}^T z^{(t+1)} \\
    \bar{y}_{i}^{(t+1)} &= y_{i}^{(t+1)} + \sigma_{\text{ex}}(y_{i}^{(t+1)} - y_{i}^{(t)})
\end{align*}
\]
Learning

Given training pairs composed of inputs \( \{x_n\}_{n=1}^N \) and their corresponding output \( \{y_{n}^{gt}\}_{n=1}^N \), learning aims at finding parameters which minimizes a regularized loss function:

\[
w^* = \arg\min_w \sum_n \ell(y^*_n, y_n^{gt}) + \gamma ||w||_2
\]  

(7)

Where \( \ell() \) is the loss, \( y^* \) is the minimizer of RNN, and \( \gamma \) is a scalar.
Algorithm: Learning Continuous-Valued Deep Structured Models
Repeat until stopping criteria
1. Forward pass to compute $h_i(x, w)$ and $h_\alpha(x, w)$
2. Compute $y^*$ via forward pass in Eq. (5)
3. Compute the gradient via backward pass
4. Parameter update

Figure 2: Algorithm for learning proximal deep structured models.
Outline

1 Intro

2 Deep Structured Prediction

3 Proximal Methods
   - Proximal Methods
   - Proximal Deep Structured Models
   - Solving Proximal Deep Structured Models

4 Experiments and Results
   - Image Denoising/Depth Refinement
   - Optical Flow
Image Denoising

Corrupt each image with Gaussian noise and use the following energy function to denoise:

\[
y^* = \arg\min_{y \in Y} \sum_i ||y_i - x_i||_2^2 + \sum_\alpha \lambda ||w_{ho,\alpha}^Ty_\alpha||_1
\]

where \(\text{prox}_{\ell_2}(y, \lambda) = \frac{x + \lambda y}{1 + \lambda}\) and \(\text{prox}_\rho^*(z) = \min(|z|, 1) \cdot \text{sign}(z)\)
Image Denoising

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>28.56</td>
<td>28.68</td>
<td>28.70</td>
<td>28.72</td>
<td>28.75</td>
<td><strong>28.79</strong></td>
<td><strong>28.79</strong></td>
</tr>
<tr>
<td>Time (second)</td>
<td>2.57</td>
<td>108.72</td>
<td>516.48</td>
<td>5.10</td>
<td>69.25</td>
<td><strong>0.23</strong></td>
<td><strong>0.011</strong></td>
</tr>
</tbody>
</table>

Table 1: Natural Image Denoising on BSDS dataset [23] with noise variance $\sigma = 25$.

Figure 3: Qualitative results for image denoising. Left to right: noisy input, ground-truth, our result.
Depth Refinement

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>32.29</td>
<td>30.95</td>
<td>24.37</td>
<td>35.46</td>
<td>35.63</td>
<td>36.31</td>
</tr>
</tbody>
</table>

Table 3: Performance of depth refinement on dataset [10]

Figure 4: Qualitative results for depth refinement. Left to right: input, ground-truth, wiener filter, bilateral filter, BM3D, Filter Forest, Ours.
Optical Flow

Predict the motion between two image frames for each pixel

\[
y^* = \arg\min_{y \in \mathcal{Y}} \sum_{i} ||y_i - f_i(x^l, x^r, w_u)||_1 + \sum_{\alpha} \lambda ||w_{ho,\alpha}^T y_\alpha||_1
\]  

(9)
Optical Flow

<table>
<thead>
<tr>
<th></th>
<th>Flownet</th>
<th>Flownet + TV-11</th>
<th>Our proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>End-point-error</td>
<td>4.98</td>
<td>4.96</td>
<td>4.91</td>
</tr>
</tbody>
</table>

Table 4: Performance of optical flow on Flying chairs dataset [11]

Figure 5: Optical flow: Left to right: first and second input, ground-truth, Flownet [11], ours.