Belief Propagation Network for Hard Inductive Semi-Supervised Learning

Jaemin Yoo
Seoul National Univ.
jaeminyoo@snu.ac.kr

Hyunsik Jeon
Seoul National Univ.
jeon185@snu.ac.kr

U Kang
Seoul National Univ.
ukang@snu.ac.kr

Summary

- **Given:** Graph-structured data with partially observed labels
- **Problem:** Train a robust classifier in a semi-supervised setting that works independently for each node without neighbors
- **Main idea:** Propose a belief propagation network (BPN), which uses a classifier to compute the priors of nodes and then diffuse them through the graph, independently from the priors
- **Homepage (+ source codes):** https://datalab.snu.ac.kr/bpn

Belief Propagation Network (BPN)

- **Idea 1:** Separate the classification of each node and diffusion of the predictions (or priors) over the graph
- **Idea 2:** Model the diffusion as a parameter-free operation
- **Idea 3:** Use an induction loss to make the classifier mimic the diffusion by its own structure, without an actual graph
- **Overview:** BPN is an algorithm to train a given classifier \( f \) by diffusing its predictions over the given graph

Forward Propagation

1. **Prior Computation**
   Use a classifier \( f \) to compute the prior \( \phi_i \) of node \( i \)
   \[
   \phi_i = f(\mathbf{x}_i; \theta)
   \]
   - \( \mathbf{x}_i \) is the feature vector of node \( i \)
   - \( \theta \) is the set of parameters of the classifier \( f \)
   - \( \phi_i \) is a probability vector that sums to one

2. **Diffusion by Loopy Belief Propagation**
   Diffuse the predicted priors by loopy belief propagation
   \[
   \mathbf{m}_{ij}^t = [\psi(\mathbf{b}_{ij}^{t-1} \odot \mathbf{m}_{ij}^{t-1})]
   \]
   \[
   \mathbf{b}_{ij}^t = \text{softmax}(\log \phi_i + \sum_{j \in \mathcal{N}_i} \log \mathbf{m}_{ij}^t)
   \]
   - \( \psi \) is an edge potential matrix that imposes adjacent nodes to have the same label (correlation by the graph structure)
   - \( \mathbf{m}_{ij}^t \) is a message from node \( i \) to node \( j \) at iteration \( t \)
   - \( \mathbf{b}_{ij}^t \) is a belief (diffused prediction) of node \( j \) at iteration \( t \)

Backward Propagation

1. **Classification Loss**
   How well the observed labels are predicted by the beliefs
   \[
   l_c(\theta) = -\sum_{i \in \mathcal{V}_o} \log \mathbf{b}_{ij}(y_i)
   \]

2. **Induction Loss**
   How well the diffused beliefs are predicted by the priors
   \[
   l_d(\theta) = -\sum_{i \in \mathcal{V}_o} \sum_{s \in \mathcal{S}} \mathbf{b}_{ij}(s)(\log \phi_i(s) - \log \mathbf{b}_{ij}(s))
   \]

3. **Overall Loss Function**
   Combine the two losses by a hyperparameter \( \beta \)
   \[
   l(\theta) = (1 - \beta)l_c(\theta) + \beta l_d(\theta) + \lambda \| \theta \|^2
   \]
   \( \beta \) is set to a value between 0 and 1 (0.5 in experiments), \( \lambda \) is an L2 regularization parameter for the training

Experiments

- **Classification accuracy (vs. competitors)**
  - Planetoid (Yang et al., 2016)
  - GCN (Graph convolutional networks, Kipf et al., 2017)
  - SEANO (Liang et al., 2018)
  - GAT (Graph attention networks, Velickovic et al., 2018)

<table>
<thead>
<tr>
<th>Method</th>
<th>Pubmed</th>
<th>Cora</th>
<th>Citeseer</th>
<th>Amazon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planetoid</td>
<td>74.6 ± 0.5</td>
<td>66.2 ± 0.9</td>
<td>66.8 ± 1.0</td>
<td>70.1 ± 1.9</td>
</tr>
<tr>
<td>GCN-I</td>
<td>74.1 ± 0.2</td>
<td>67.8 ± 0.6</td>
<td>63.6 ± 0.5</td>
<td>76.5 ± 0.3</td>
</tr>
<tr>
<td>SEANO</td>
<td>75.7 ± 0.4</td>
<td>64.5 ± 1.2</td>
<td>66.3 ± 0.8</td>
<td>78.6 ± 0.6</td>
</tr>
<tr>
<td>GAT</td>
<td>76.5 ± 0.4</td>
<td>70.1 ± 1.0</td>
<td>66.7 ± 1.0</td>
<td>77.5 ± 0.4</td>
</tr>
<tr>
<td>BPN (ours)</td>
<td>78.3 ± 0.3</td>
<td>72.2 ± 0.5</td>
<td>70.1 ± 0.9</td>
<td>81.5 ± 1.3</td>
</tr>
</tbody>
</table>

- **Training process (loss values)**
  - Classification loss decreases continuously
  - Induction loss increases at first then decreases

This is because \( f \)'s updates change the beliefs of nodes!