Belief Propagation Network
for Hard Inductive Semi-Supervised Learning

Jaemin Yoo, Hyunsik Jeon and U Kang
Seoul National University

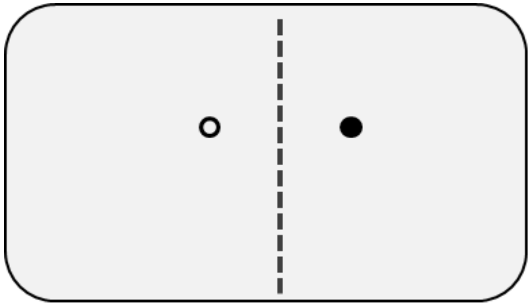
IJCAI 2019

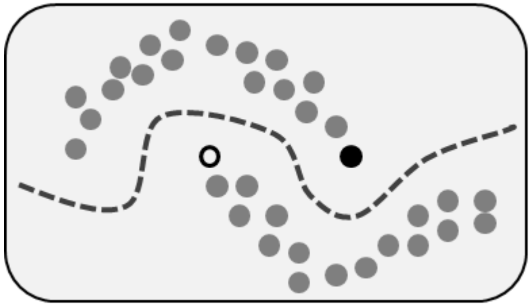
Outline

- **Introduction**
- Proposed Method
- Experiments
- Conclusion

Semi-Supervised Learning

- **Semi-supervised learning**
  - Leverage unlabeled data for better performance



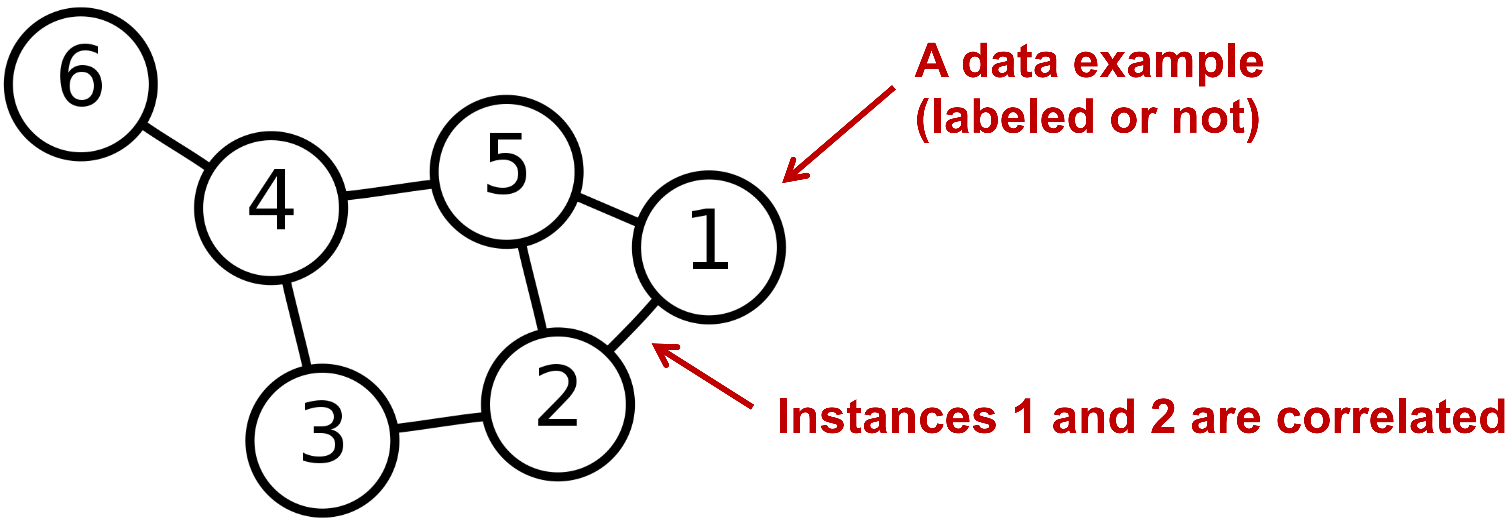


- **Graph-based semi-supervised learning**
  - Focus on the data represented as graphs
  - Easy to capture relationships between examples

Image from https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph-Based Learning

- Basic assumption of graph-based learning
  - Nearby examples are correlated positively
- Model the **correlations** by neural networks
  - GCN (ICLR'17), GAT (ICLR'18), …



Motivation

- Previous works have naïve assumptions
- 1) A graph is given also at the test time
  - **No graphs** in real-time classification
- 2) Every node has enough neighborhoods
  - **No neighbors** for fresh users or items

How to address these limitations?

Problem Definition

- **Problem: hard inductive learning**
- **Given**
  - An undirected graph G
    - Each node is an example $(\mathbf{x}, y)$ or $(\mathbf{x}, \cdot)$
  - Labels of only a subset of nodes
- **Learn**
  - A classifier $f : \mathbf{x} \to y$
    - Predicts each example independently
    - Does not require the graph at the test time

Outline

- Introduction
- **Proposed Method**
- Experiments
- Conclusion

Belief Propagation Network

- Novel approach for hard inductive learning
- Separates classification and diffusion steps
- **Classification**
  - Classify each node i independently by $f(\mathbf{x}_i)$
  - The graph structure is not considered at all
- **Diffusion**
  - Diffuse the predictions through the graph
  - Update f based on the results of diffusion

Classification: MLP

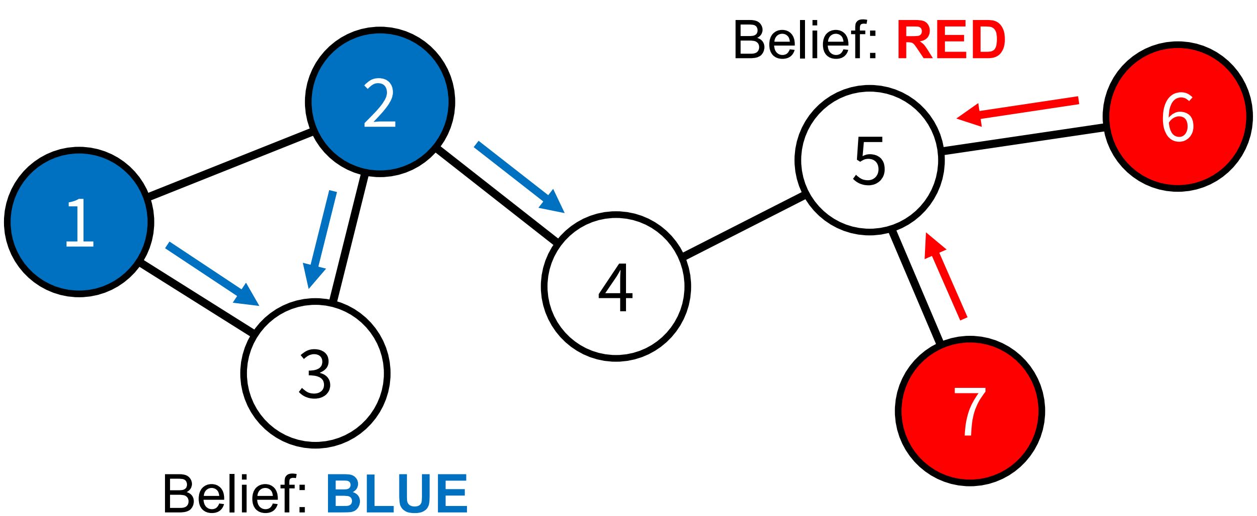
- Any model can be used as a classifier f
- Our choice is a multilayer perceptron (MLP)
  - Single hidden layer of 64 units
  - Tanh as an activation function

$$\phi_i = \mathrm{softmax}(\mathbf{W}_2(\tanh(\mathbf{W}_1\mathbf{x}_i + \mathbf{b}_1)) + \mathbf{b}_2)$$

- Produce a probability vector ϕ_i for node i

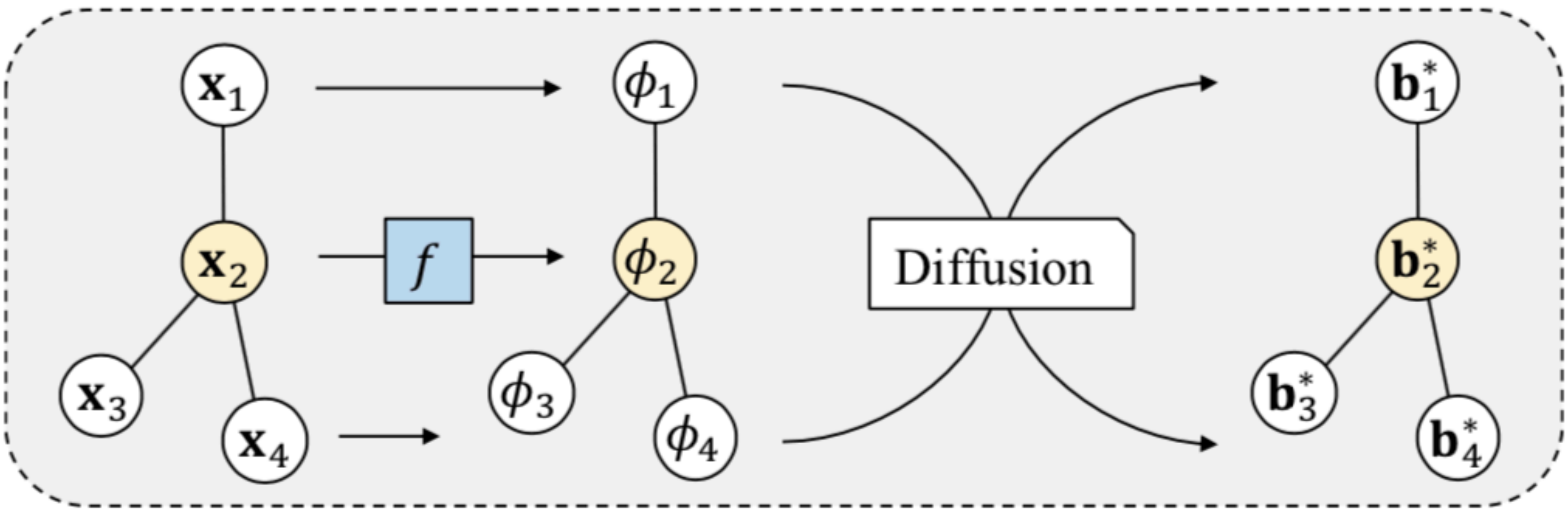
Diffusion: LBP

- Run **loopy belief propagation** for diffusion
  - Takes predictions of f as **priors** of nodes
  - Propagates the priors and computes **beliefs**



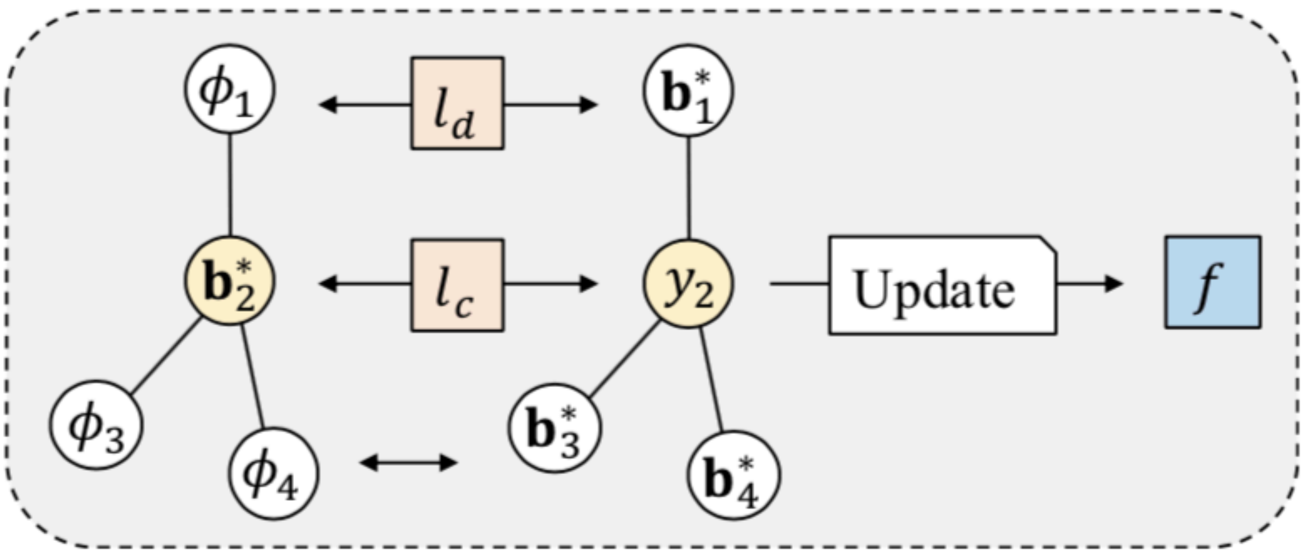
Forward Propagation

- Computes the **prior** $\phi_i = f(\mathbf{x}_i)$ of node i
- Diffuses the priors by running LBP
  - The **belief** $\mathbf{b}_i^*$ of each node i is computed



Backward Propagation

- Compute **two loss functions** l_c and l_d
  - Classification loss l_c for the labeled nodes
  - Induction loss l_d for the unlabeled nodes
- f is updated to minimize the sum of l_c and l_d



Loss Functions

- **Classification loss**
  - Typical loss function for classification
  - Cross-entropy between labels and beliefs
- **Induction loss**
  - Our proposed loss for unlabeled nodes
  - KL-divergence between beliefs and priors
  - Make f learn the results of LBP as soft labels

Outline

- Introduction
- Proposed Method
- **Experiments**
- Conclusion

Datasets

- Solve node classification on four datasets
  - Three datasets are citation graphs
    - Classify the area of each research article
  - The other is an Amazon graph of items
- 20 labeled examples for each class

Name	Nodes	Edges	Attributes	Labels
Pubmed[1]	19,717	44,324	500	3
Cora[1]	2,708	5,278	1,433	7
Citeseer[1]	3,327	4,552	3,703	6
Amazon	32,966	63,285	3,000	3

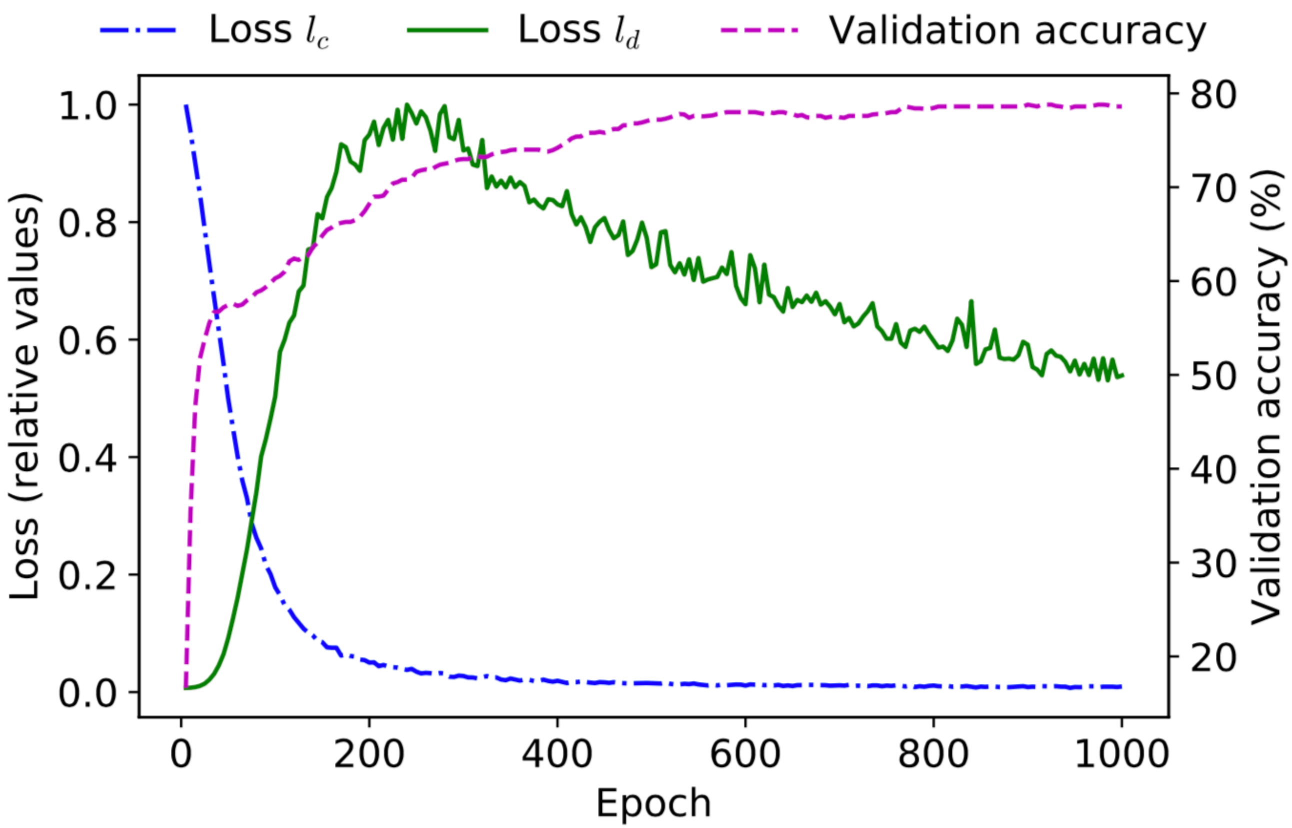
Classification Accuracy

- BPN shows the best classification accuracy

Method	Pubmed	Cora	Citeseer	Amazon
Planetoid	74.6 ± 0.5	66.2 ± 0.9	66.8 ± 1.0	70.1 ± 1.9
GCN-I	74.1 ± 0.2	67.8 ± 0.6	63.6 ± 0.5	76.5 ± 0.3
SEANO	75.7 ± 0.4	64.5 ± 1.2	66.3 ± 0.8	78.6 ± 0.6
GAT	76.5 ± 0.4	70.1 ± 1.0	66.7 ± 1.0	77.5 ± 0.4
BPN (ours)	$\mathbf{78.3 \pm 0.3}$	$\mathbf{72.2 \pm 0.5}$	$\mathbf{70.1 \pm 0.9}$	$\mathbf{81.5 \pm 1.3}$

Loss Values

- BPN minimizes the two loss functions



Outline

- Introduction
- Graph-based Learning
- Proposed Method
- **Conclusion**

Conclusion

- We solve the **hard inductive leaning**
  - A graph is not given at the test time
- We propose a **belief propagation network**
  - Classify each node by a classifier f
  - Diffuse the predictions (or priors) by LBP
  - Update f by minimizing two loss functions
- BPN **outperforms** the SOTA approaches

Thank you !

https://datalab.snu.ac.kr/bpn