Comp790-166: Computational Biology

Lecture 25

April 12, 2022

Good Morning Question

- What was the main idea behind the MNC (matched neighborhood consistency score?)
- What was a trick that the graph refinement approach used to better scale the methods to larger graphs (hint : it had to do with updating alignments). Recall $\mathbf{A}_1\mathbf{M}_0\mathbf{A}_2$

Today

- Graph Neural Networks vs Label Propagation vs LP + Correct and Smooth

Announcements

- Homework 2 is online, fixed a couple of typos
- Project Presentations April 25 and April 27. Please visit the signup sheet. `https://docs.google.com/spreadsheets/d/1_z1NBffJF8do8JrasTQl-8pS-ATR2ScI7SutRDPIf80/edit?usp=sharing` $\rightarrow$ signed up people who didn't sign up
- Final Project LaTeX template `https://github.com/natalies-teaching/Comp790-166-CompBio-Spring2022/tree/main/Project_Final_Writeup`
- I encourage in person attendance for the project presentation days.
- Final Project Writeup is due

GNN vs Simple Things

- You all love GNNs
- Recently there has been some work to study why GNNs are outperforming simpler methods and how to incorporate this intuition back to simpler methods

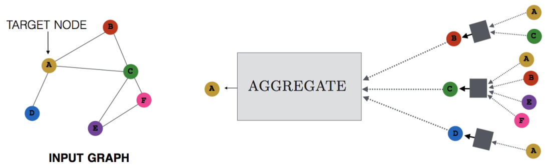


Figure: from https://www.cs.mcgill.ca/~wlh/grl_book/files/GRL_Book-Chapter_5-GNNs.pdf. Messages are aggregated from the neighborhood of some target node.

Neural Message Passing

- **Input:** $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a set of node features, $\mathbf{X} \in \mathbb{R}^{d \times |\mathcal{V}|}$
- **Output:** Node embeddings, $\mathbf{z}_u, \forall u \in \mathcal{V}$
- **Each Message-Passing Iteration:** A hidden embedding $\mathbf{h}_u^{(k)}$ is updated according to information aggregated from node u's neighborhood, $\mathcal{N}(u)$.

Update Rule

$$\begin{aligned}\mathbf{h}_u^{(k+1)} &= \text{UPDATE}^{(k)}\left(\mathbf{h}_u^{(k)}, \text{AGGREGATE}^{(k)}\left(\left\{\mathbf{h}_v^{(k)}, \forall v \in \mathcal{N}(u)\right\}\right)\right)\\ &= \text{UPDATE}^{(k)}\left(\mathbf{h}_u^{(k)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}\right)\end{aligned}$$

- $\mathbf{m}_{\mathcal{N}_{(u)}}^{(k)}$ is the *message* that is aggregated from u's graph neighborhood, $\mathcal{N}(u)$
- At each iteration, the AGGREGATE function takes as input the set of embeddings of the nodes in u's graph neighborhood, $\mathcal{N}(u)$ and applies it to a previous embedding $\mathbf{h}_u^{(k-1)}$ to generate the updated embedding $\mathbf{h}_u^{(k)}$

Illustrated..

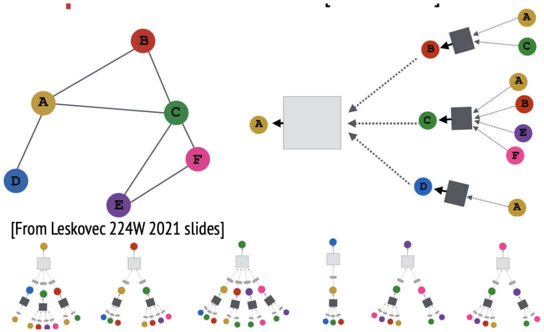
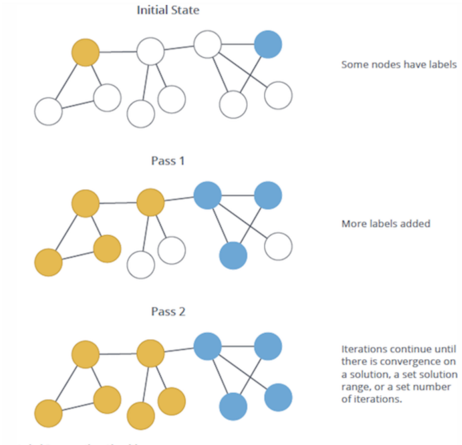


Figure: from https://www.cs.cornell.edu/~arb/slides/2021-03-12-northeastern.pdf

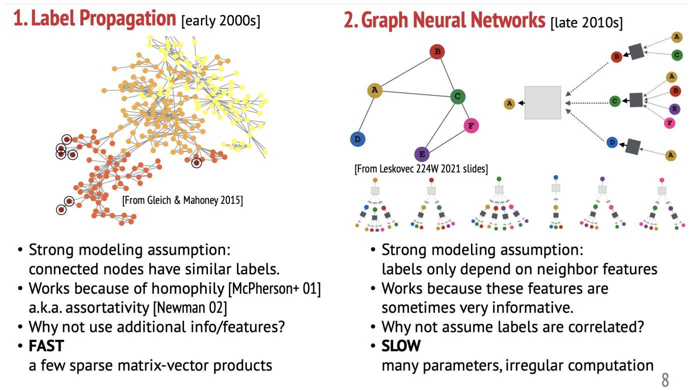
What is Label Propagation?

Some of your nodes are labeled, others are unlabeled, and you predict labels of unlabeled nodes based on the structure of the graph.


Initial State
Some nodes have labels
Pass 1
More labels added
Pass 2
Iterations continue until there is convergence on a solution, a set solution range, or a set number of iterations.

Label Propagation Algorithm

The Debate



1. Label Propagation [early 2000s]

[From Gleich & Mahoney 2015]

- Strong modeling assumption: connected nodes have similar labels.
- Works because of homophily [McPherson+ 01] a.k.a. assortativity [Newman 02]
- Why not use additional info/features?
- **FAST**
  a few sparse matrix-vector products

2. Graph Neural Networks [late 2010s]

[From Leskovec 224W 2021 slides]

- Strong modeling assumption: labels only depend on neighbor features
- Works because these features are sometimes very informative.
- Why not assume labels are correlated?
- **SLOW**
  many parameters, irregular computation

8

Figure: from https://www.cs.cornell.edu/~arb/slides/2021-03-12-northeastern.pdf

Tradeoffs

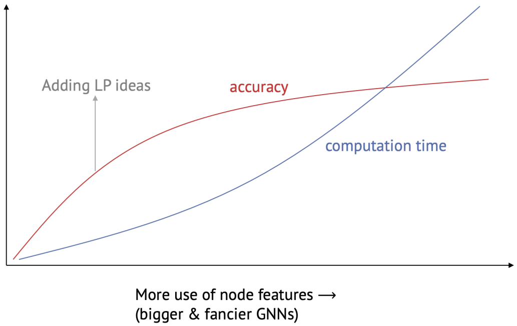


Figure: from `https://www.cs.cornell.edu/~arb/slides/2021-03-12-northeastern.pdf`

Correct and Smooth Approach

- The goal is to compare how a couple of simple methods/intuition can be strung together can be used to classify nodes
- The main idea is to start with a cheap base prediction based on node features (e.g. attributes or coordinates of a spectral embedding), and clean up graph structure through label propagation (**correct and smooth**).

Overview of Correct and Smooth Approach

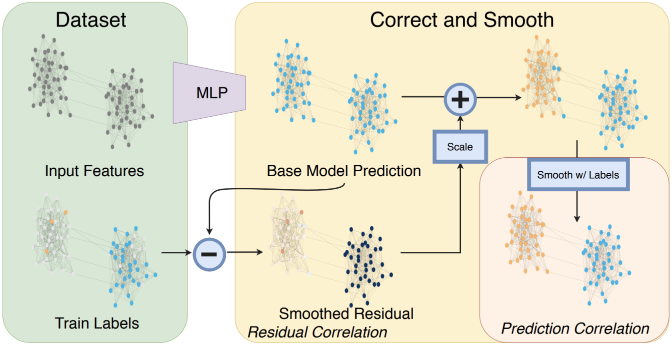


Figure: from Huang *et al.* ICLR. 2021

Notation Preliminaries

- Let there be n nodes.
- Assume we have a feature vector for each node, such that node features are encodes in an $n \times p$ matrix, X.
- Similarly, let A be the adjacency matrix of the graph
- Split nodes into labeled (L) and unlabeled (U) sets
- Define an $n \times c$ matrix, Y with a binary indicator for whether node i is in class c.

Simple Base Predictor

Given the matrix of features for each node, X and labels, Y, train a simple model to minimize,

$$\sum_{i \in L_t} \ell\left(f\left(x_i\right), y_i\right)$$

- ℓ is some loss
- Here L_t denotes the set of labeled training nodes
- Specify a matrix, Z containing these base predictions.

Error Correlation

- The intuition is that errors are expected to be correlated across edges in the graph. Hence, spread uncertainty across the edges.

Define and error matrix, $E \in \mathbb{R}^{n\times c}$ as,

$$E_{L_t,:} = Y_{L_t,:} - Z_{L_t,:}, \quad E_{L_v,:} = 0, \quad E_{U,:} = 0$$

This means that the only non-zero entries are those that correspond to labeled training nodes!

Smooth the Error Using a Label Spreading Technique

The errors are smoothed as follow with a label spreading technique,

$$\hat{E} = \underset{W \in \mathbb{R}^{n \times c}}{\arg\min}\ \text{trace}\left(W^T(I - S)W\right) + \mu\|W - E\|_F^2$$

- S is the normalized adjacency matrix, $D^{-1/2}AD^{-1/2}$
- The first term encourages smoothness of the error over the graph
- The second term keeps W close to the initial estimate of error, E.

Solution

Given

$$\hat{E} = \underset{W \in \mathbb{R}^{n \times c}}{\arg\min} \operatorname{trace}\left(W^T (I - S) W\right) + \mu \|W - E\|_F^2$$

it was previously shown that the solution can be obtained through the following iteration,

$$E^{(t+1)} = (1 - \alpha) E + \alpha S E^{(t)}$$

The quickly converges to $\hat{E}$ and therefore gives corrected predictions as,

$$Z^r = Z + \hat{E}$$

Smoothing Final Predictions with Prediction Correlation

- The next assumption to be used for correction is that adjacent nodes in the graph are likely to have similar labels (e.g. homophily)
- Another round of label propagation will be used to encourage smoothness over distribution of labels.

Starting with the best guess of the labels, H, with $H_{L_t,:} = Y_{L_t,:}$ and $H_{L_v \cup U,:} = Z^{(r)}_{L_v \cup U,:}$, propagate labels as,

$$H^{(t+1)} = (1-\alpha)H + \alpha S H^{(t)}$$

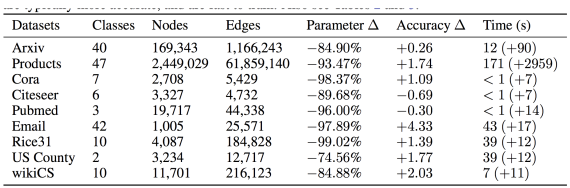
Final Prediction

The following has now been applied

- Base prediction
- Residual correction
- Label smoothing

After convergence of $H^{(t+1)} = (1-\alpha)H + \alpha S H^{(t)}$, get a final prediction, $\hat{Y} \in \mathbb{R}^{n \times c}$, and assign node to the class with the max predicted probability.

Results



Datasets	Classes	Nodes	Edges	Parameter Δ	Accuracy Δ	Time (s)
Arxiv	40	169,343	1,166,243	-84.90%	$+0.26$	12 $(+90)$
Products	47	2,449,029	61,859,140	-93.47%	$+1.74$	171 $(+2959)$
Cora	7	2,708	5,429	-98.37%	$+1.09$	< 1 $(+7)$
Citeseer	6	3,327	4,732	-89.68%	-0.69	< 1 $(+7)$
Pubmed	3	19,717	44,338	-96.00%	-0.30	< 1 $(+14)$
Email	42	1,005	25,571	-97.89%	$+4.33$	43 $(+17)$
Rice31	10	4,087	184,828	-99.02%	$+1.39$	39 $(+12)$
US County	2	3,234	12,717	-74.56%	$+1.77$	39 $(+12)$
wikiCS	10	11,701	216,123	-84.88%	$+2.03$	7 $(+11)$

Figure: from Table 1. Performance is in reported to SOTA GNN.

Accuracy vs Number of Parameters

Higher accuracy with less parameters on one of the datasets (and training is also significantly faster)

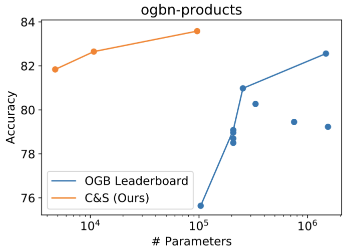


Figure: from Fig. 2

Visualizing which correction step fixed error

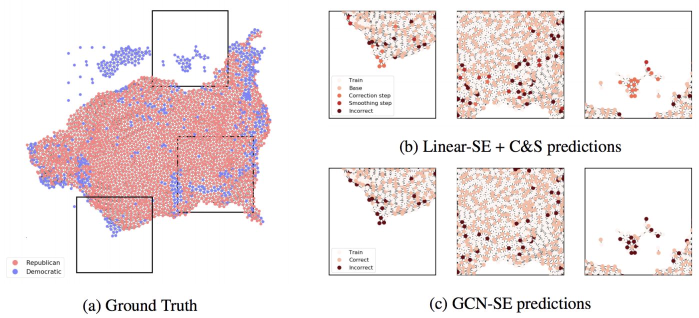


Figure: from Fig. 3.

Summary

- Simple LP, diffusion, and GNN are fundamentally related
- Augmenting graph information with attributes, spectral features, etc. can be helpful for classifying nodes
- A base prediction is corrected according to smoothing over residual errors and encouraging closely connected nodes to have similar labels.