Active Learning on Graphs - Sampling the Initial Set

Florence Regol
Dep. Electrical and Computer Engineering
McGill University
Montréal, Québec, Canada

McGill

Computer Networks Research Laboratory

How do we study for a test?

- We **choose** what to study
- As we go, we **re-evaluate** what we should focus on
- Our goal is to **optimize our grade**

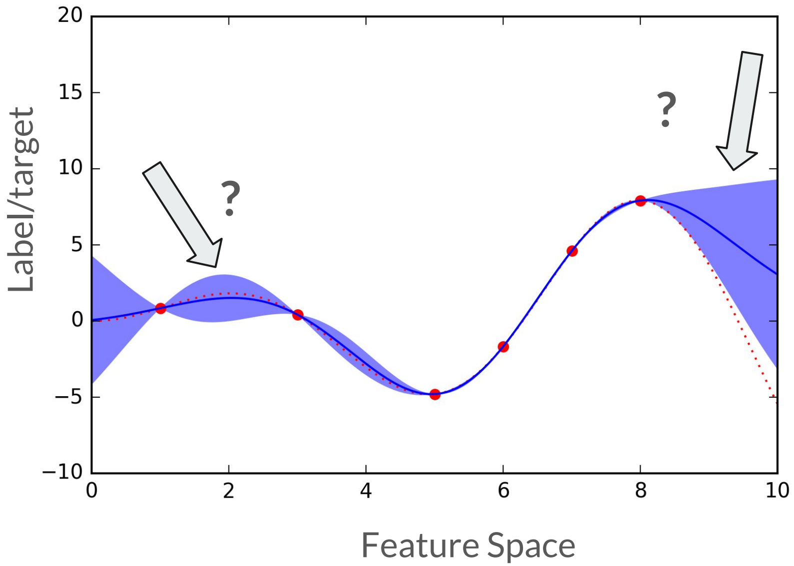
... Active Learning

Active Learning

Problem Setting :

- **Unlabeled data**
- Able to **query** an oracle to obtain labels

Goal : **Choose the optimal queries to maximize performance**



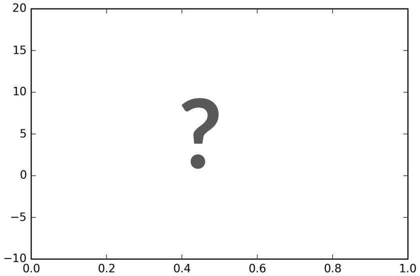
Active Learning Strategies - Graph Context

Strategies	Graph context
Heterogeneity (Uncertainty, query by committee)	Nodes far from labeled nodes
Performance (Expected error/variance reduction)	Directly optimize the cost function of the graph learning algorithm
Representativeness (choose better representation of underlying distribution)	Use embeddings to run K-mean and compute distances to the centroides

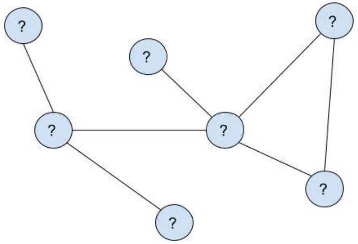
[1] H. Cai, V. W. Zheng, and K. C. Chang, “Active learning for graph embedding,” CoRR, vol. abs/1705.05085, 2017

Strategies for choosing the initial set

- Assuming IID on Euclidean space
  → Nothing better than random



- Data on graph, with some smoothness assumptions
  → Can leverage graph structure



Sampling Methods relying only on graph structure

Max Degree Sampling

- Order nodes by highest degree
- If we have to choose, sample uniformly.

```
G = {}
for # nodes to select
    Max Degree Set = maxDegree(remaining nodes)
    if |G| + |Max Degree Set| > # nodes to select:
        selected_nodes ~ Uniform(Max Degree Set)
        G = G U selected_node
        break
    G = G U (Max Degree Set)
```

Intuition : Nodes with more connections are more representative, “central” to the graph

Experimentally Designed Sampling (EDS)

- Sampling to recover a **k-sparse signal** :
- Sample node relative to their sampling score :

$$p_i = ||u_{ki}||_2 / \sum_{j \in Training} ||u_{kj}||_2$$

$$Adj = V \Lambda U$$
$$\hat{x}_k = U_k x$$
$$x = V_k \hat{x}_k$$

Intuition : Nodes selected to fully recovers a signal are more important and more representative of the graph

[3] S. Chen, R. Varma, A. Singh, and J. Kovacevic, "Signal recovery on graphs: Random versus experimentally designed sampling," 2015 International Conference on Sampling Theory and Applications (SampTA), pp. 337–341, 2015.

Greedy Sampling - Problem Setting

- **Bayesian Estimation problem**
- Goal : Estimate a signal z from a noisy observation y of a **k-sparse signal** x $\Longrightarrow$ $y = x + w$
- The recovered signal can be obtained through a linear transformation $\Longrightarrow$ $z = Hx$
- **Prior** on initial signal and noise $\Longrightarrow$ Λ, Λ_w
- Estimator is a linear interpolation from **sampled** observation y_s $\Longrightarrow$ $\hat{z} = Ly_s$

[3] L. F. O. Chamon and A. Ribeiro, "Greedy sampling of graph signals," IEEE Transactions on Signal Processing, vol.66, no. 1, pp. 34–47, 2018.

Greedy Sampling - Defining MSE

- The **Optimal** interpolation operator can be found by minimizing the **Interpolation Error Covariance Matrix :**

$$K[\hat{z}(s)] = E[(z - \hat{z}(s))(z - \hat{z}(s)^H)|x, w]$$

- The error is only dependant on the set

$$K^*(S) = HV_K(\Lambda^{-1} + \sum_{i\in s} \lambda_{w,i}^{-1} v_i v_i^H)^{-1} V_K^H H^H$$

- Can define the Mean Square Error on a set

$$MSE(S) = Tr[K^*(S)]$$

Greedy Sampling

- Still a **Combinatorial Problem** $\binom{n}{s}$
- Use a Greedy algorithm instead to minimize the MSE
- Derive **bounds** on the performance; function of the sparsity, the size of the set and α-supermodularity

```
G = {}
for # nodes to select
    selected_node = argmin MSE(G U {i})
    G = G U ({selectd_node})
```

$$f(Greedy) \leq (1 - e^{-\alpha s/k}) fopt^*$$

Taking the identity matrix as the transformation makes the results hard to interpret. (Same goes for EDS)

Problem Setting
- Experiment Description

Graph Convolutional Network(GCN)

Feature matrix X

Adjacency matrix A

GCN

- Built with layers
  - $H^{l+1} = \sigma(\hat{A}H^{l}W^{l})$
- First layer
  - $H^{1} = \sigma(\hat{A}XW_{0})$

$$\hat{A} = \hat{D}^{-1/2}(A+I)\hat{D}^{-1/2}$$

Label vector Z

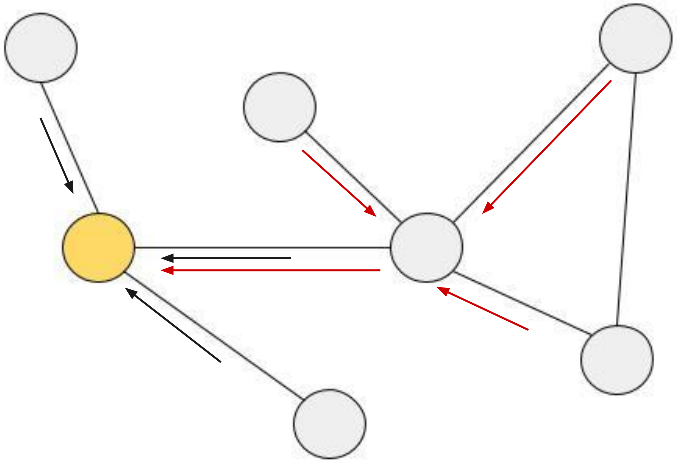
Architecture used : $f(X, A) = softmax(\hat{A}ReLU(\hat{A}XW^{0})W^{1})$

[2] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,”arXiv preprint arXiv:1609.02907, 2016.

Role of Â

- Recall $H^{l+1} = \sigma(\hat{A}H^lW^l)$
- The identity matrix ensures that we keep the features of the "main node"
- Regularized to avoid vanishing/exploding gradient as we add layers
- No Edges -> Neural Networks

$$\hat{A} = \hat{D}^{-1/2}(A + I)\hat{D}^{-1/2}$$

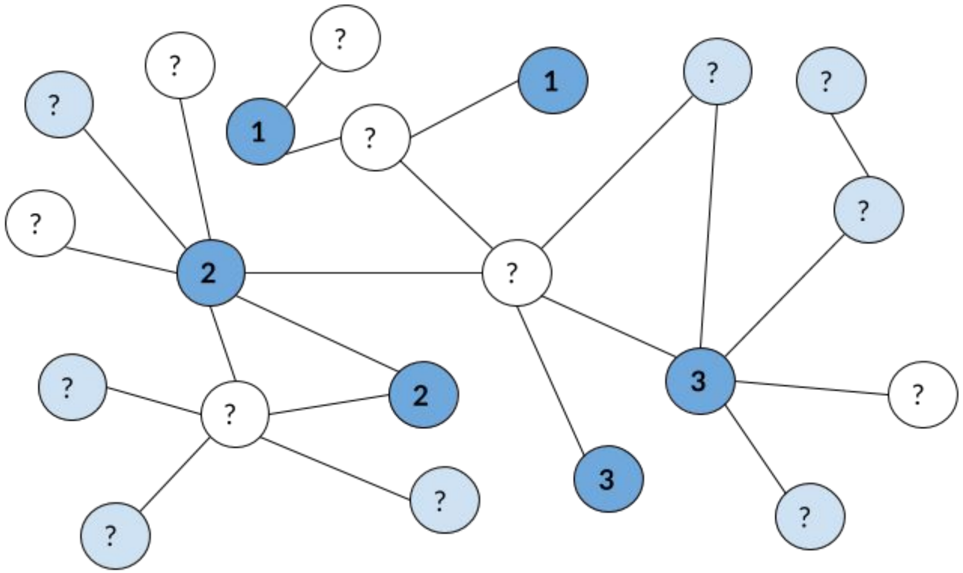


Sampling (Semi-Supervised)

Testing/Validation set

Unlabeled Training set

Labeled Training set



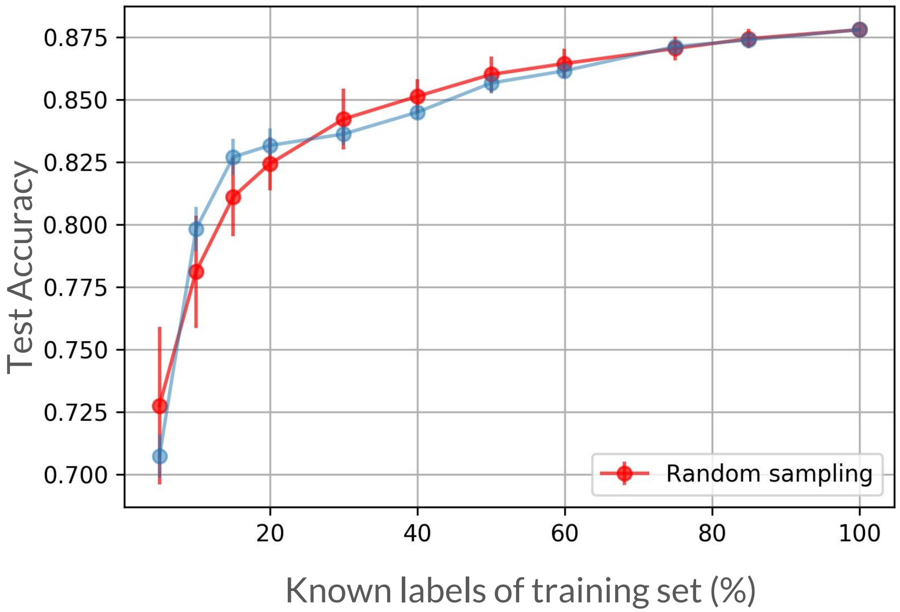
Sampling Technique Choosing the labeled training set

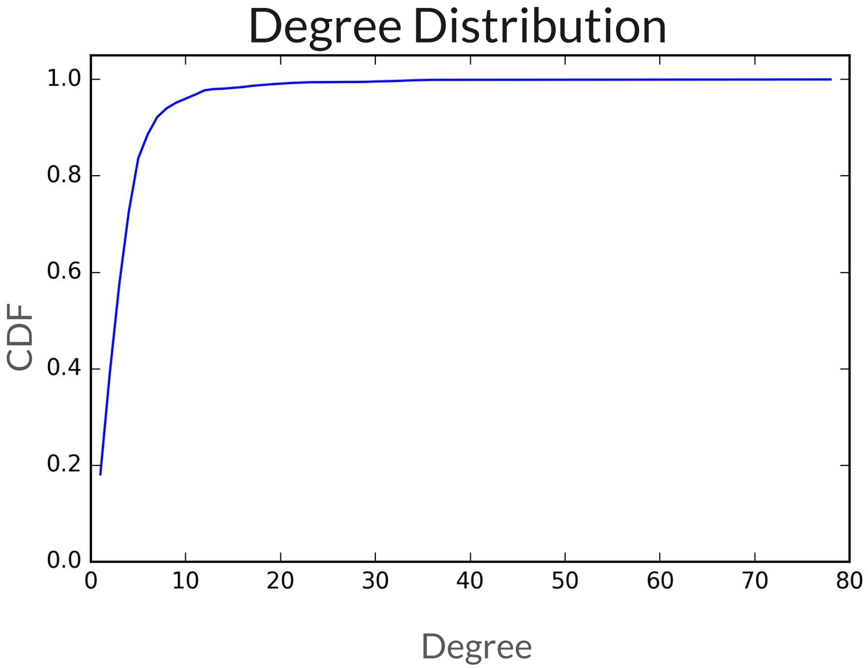
Experiment Description

Experiment Parameter	Values
Sampled Node (%)	5, 10, 15, 20, 30, 40, 50, 60, 75, 85, 100
Noise Covariance Prior (when applicable)	0.01, 1, 100
Num Eigenvector k (when applicable)	5, 10, 100
Dataset	Cora (2708 feature, 7 classes)
Num. of Sampling Trials (when applicable)	20
Num. of Cross Validation (when applicable)	4

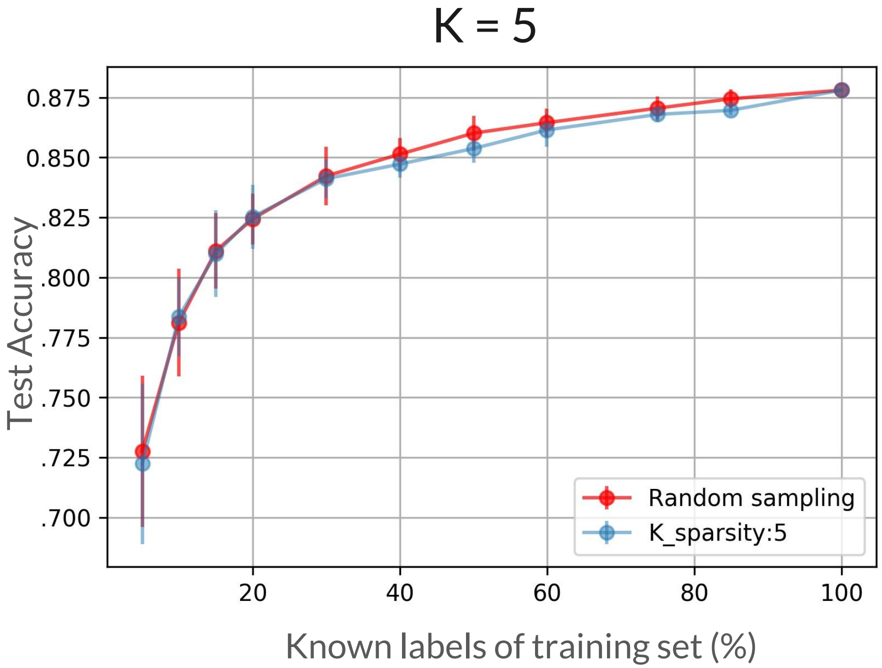
Results

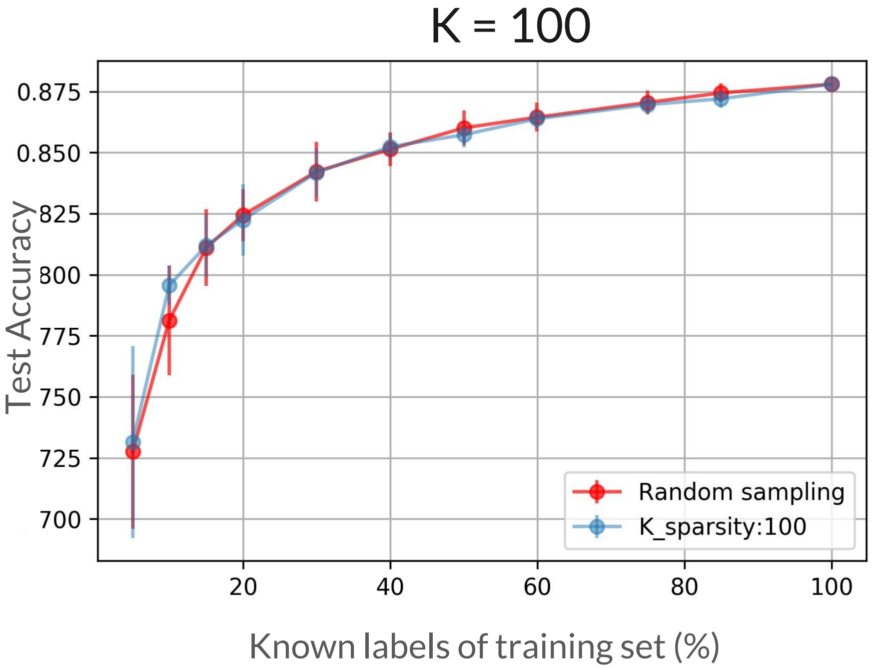
Max Degree


Test Accuracy
0.875
0.850
0.825
0.800
0.775
0.750
0.725
0.700
20
40
60
80
100
Random sampling
Known labels of training set (%)

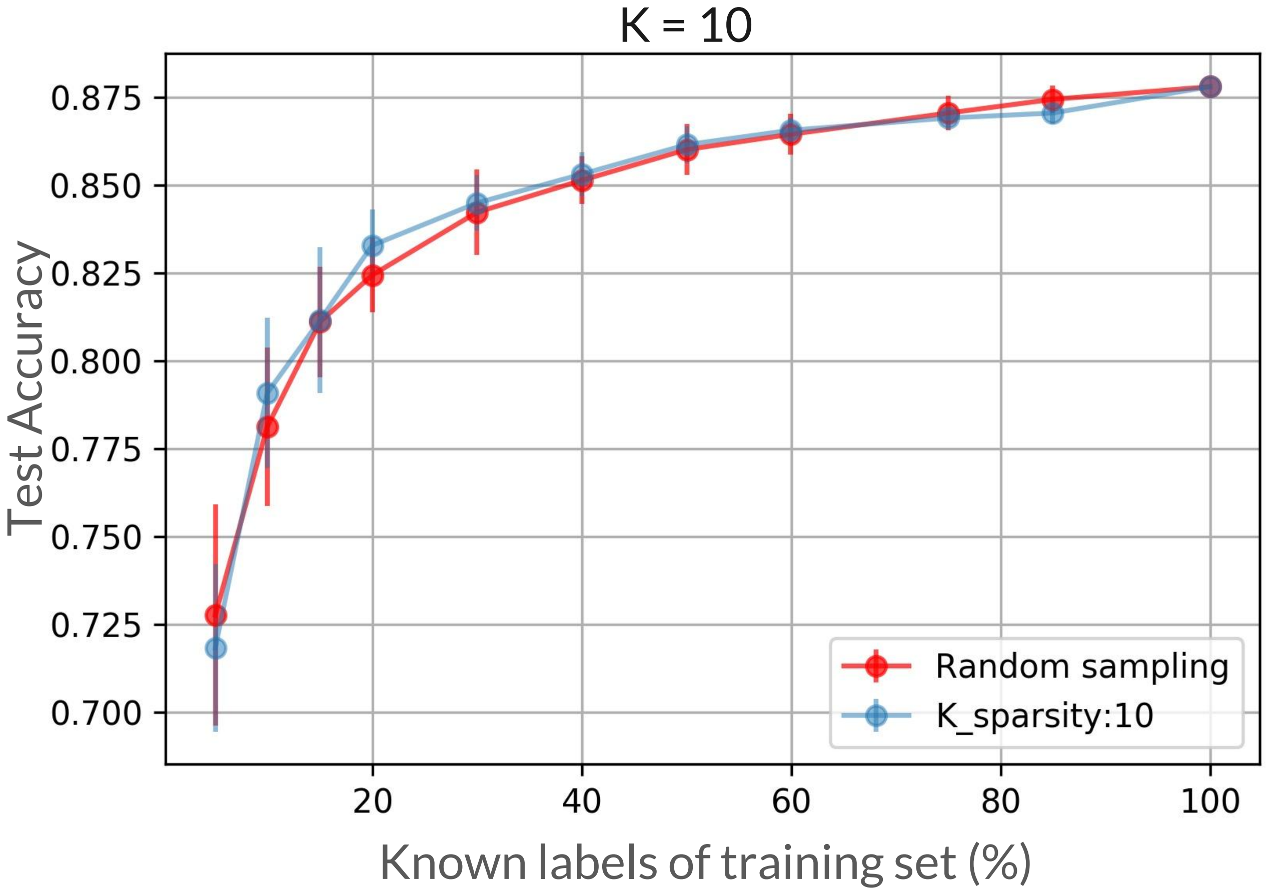

Degree Distribution
CDF
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
60
70
80
Degree

EDS


K = 5
0.875
0.850
.825
.800
.775
.750
.725
.700
Test Accuracy
20
40
60
80
100
Random sampling
K_sparsity:5
Known labels of training set (%)

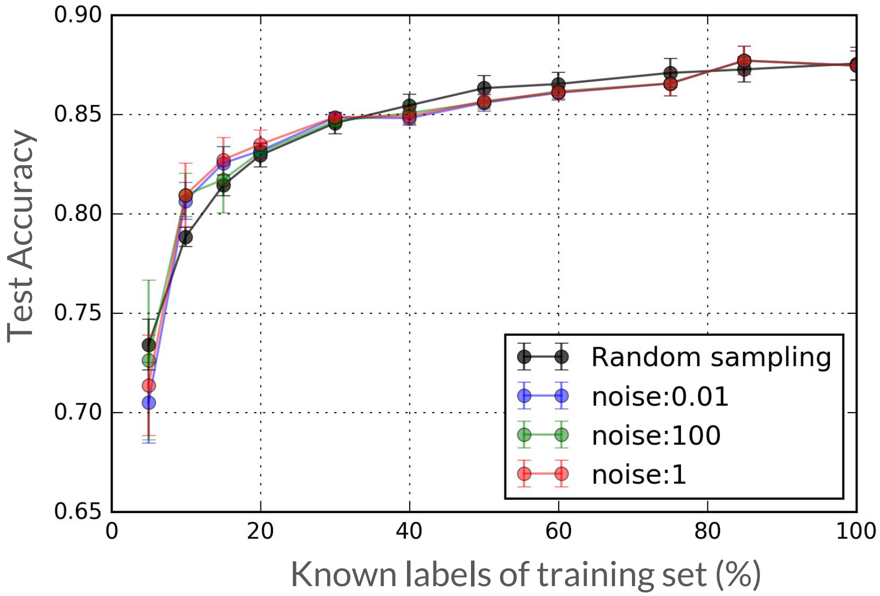

K = 100
0.875
0.850
825
800
775
750
725
700
Test Accuracy
20
40
60
80
100
Random sampling
K_sparsity:100
Known labels of training set (%)

EDS

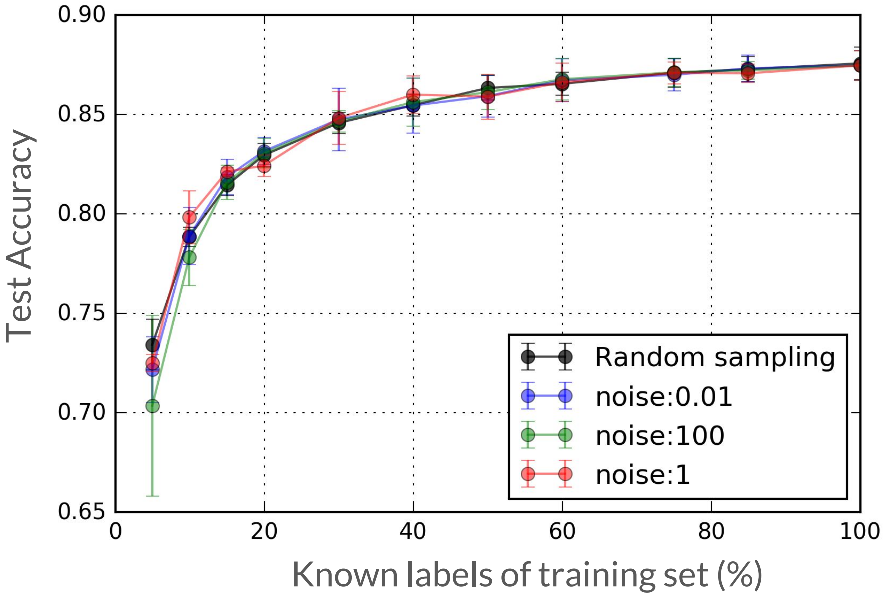

K = 10
Test Accuracy
0.875
0.850
0.825
0.800
0.775
0.750
0.725
0.700
20
40
60
80
100
Known labels of training set (%)
Random sampling
K_sparsity:10

Greedy Sampling

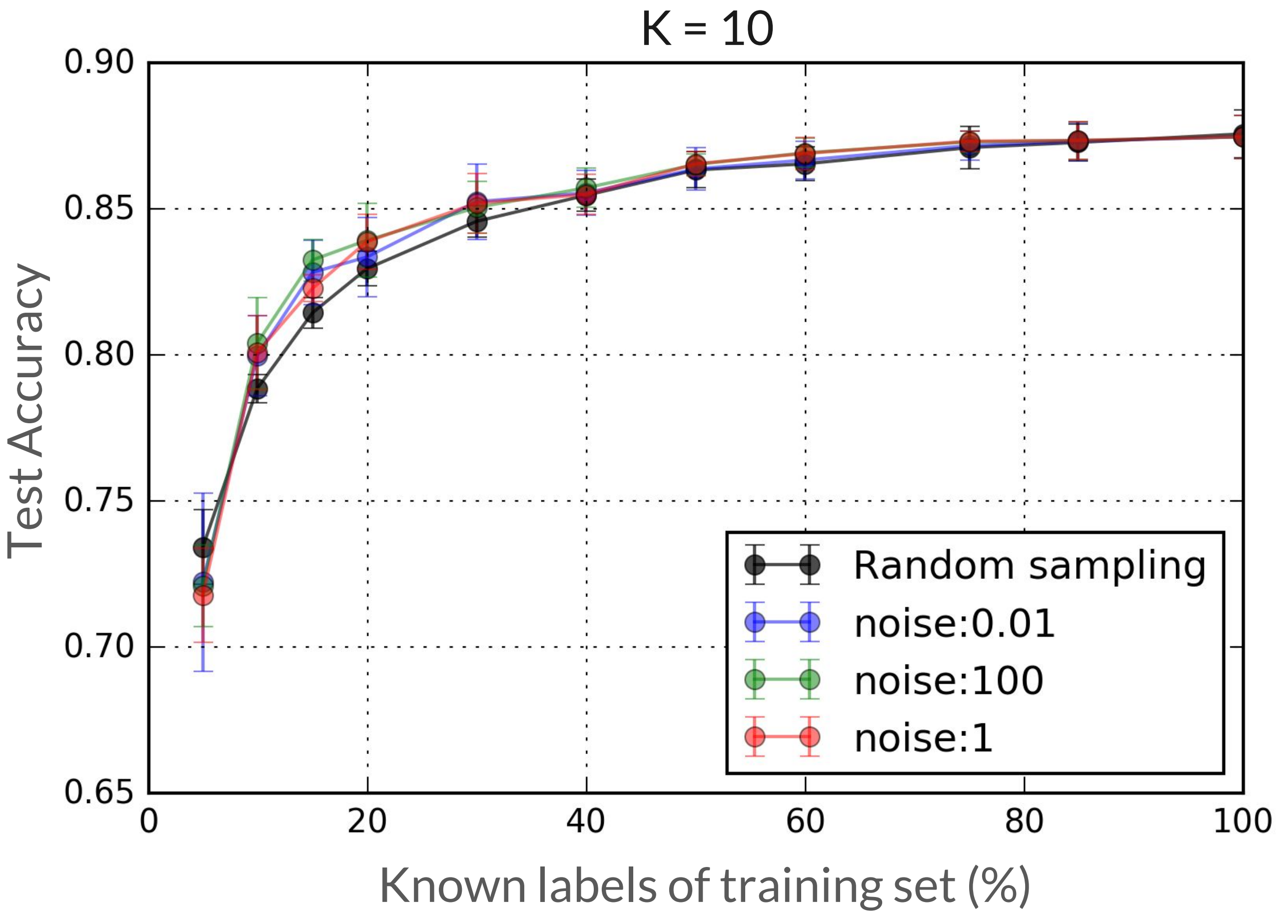
K = 5



K = 100

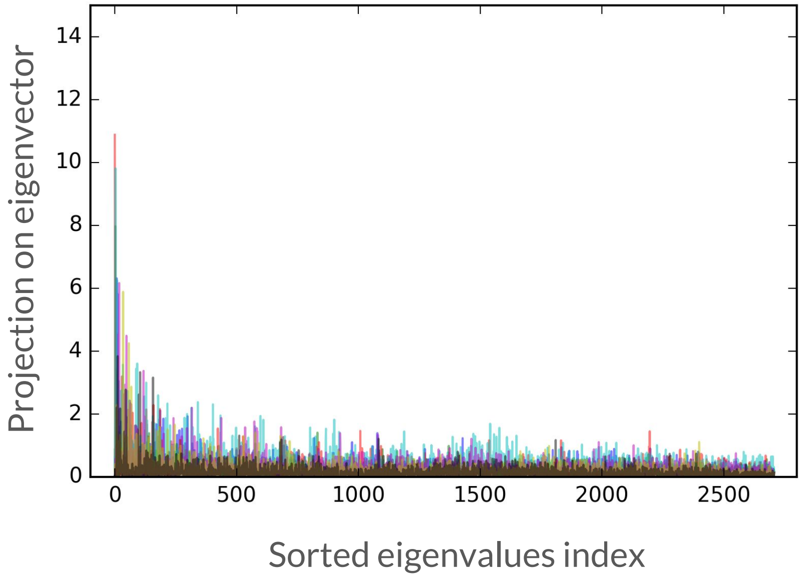


Greedy Sampling

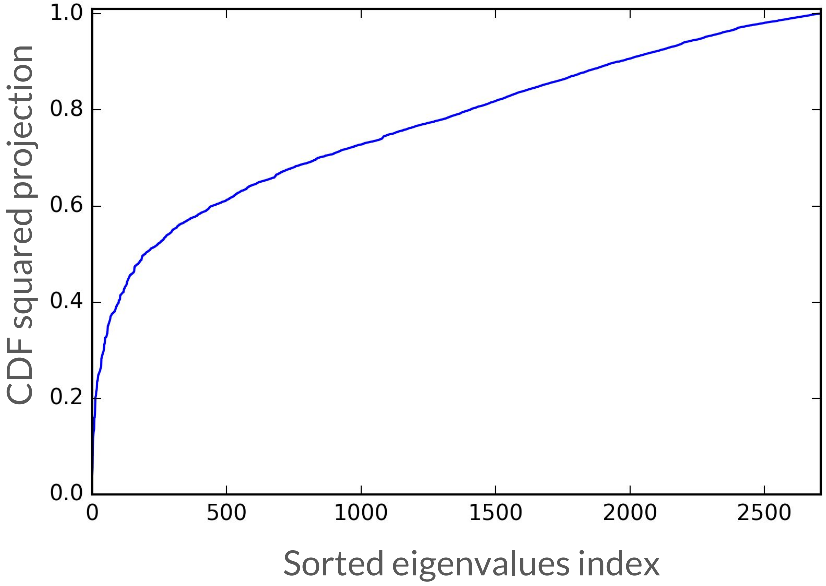


Binary Label Signals in Frequency domain

Binary Signals - Frequency Domain



Energy distribution of the Signals

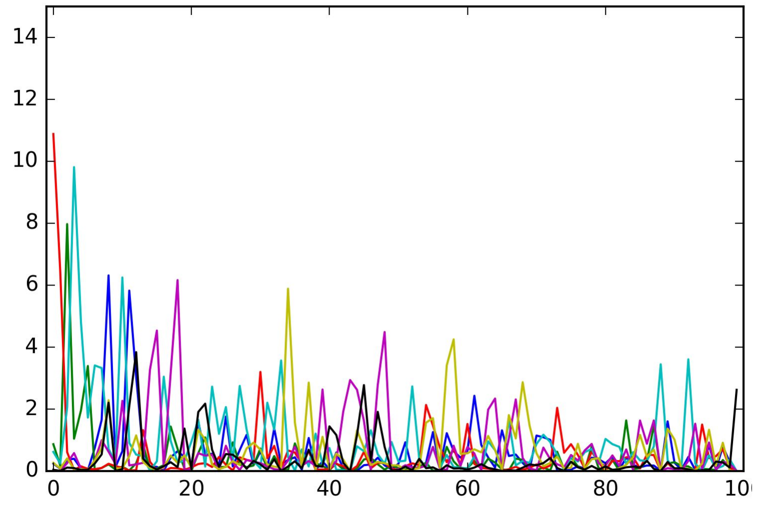


Top 100 eigenvectors

7 Binary Signals on Spectral Domain

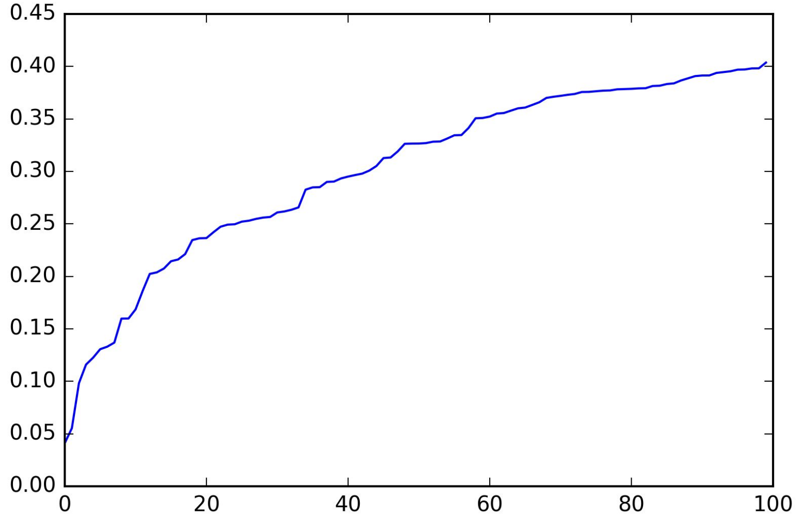
Energy distribution of the Signals

Projection on eigenvector



Sorted eigenvalues index

CDF squared projection



Sorted eigenvalues index

Conclusion and Future Work

- Greedy sampling gave a small increase of performance
    - We could try to bring closer the H matrix and the GCN
- Active learning to select following nodes

References

[1] H. Cai, V. W. Zheng, and K. C. Chang, “Active learning for graph embedding,”CoRR, vol. abs/1705.05085, 2017.

[2] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,”arXiv preprint arXiv:1609.02907, 2016.

[3] S. Chen, R. Varma, A. Singh, and J. Kovacevic, “Signal recovery on graphs: Random versus experimentally designed sampling,” 2015 International Conference on Sampling Theory and Applications (SampTA), pp. 337–341, 2015.

[4] L. F. O. Chamon and A. Ribeiro, “Greedy sampling of graph signals,” IEEE Transactions on Signal Processing, vol.66, no. 1, pp. 34–47, 2018.