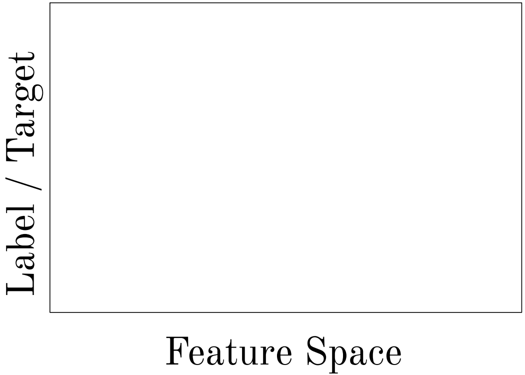
GEEM : An algorithm for Active Learning on Attributed Graphs

Florence Regol*
Soumyasundar Pal*, Yingxue Zhang**, Mark Coates*

* McGill University Compnet Lab
**Huawei Noah's Ark Lab, Montreal Research Center

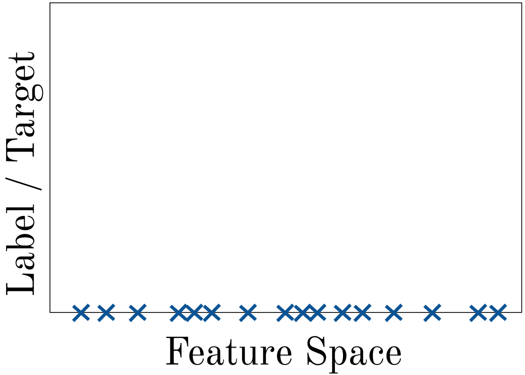
July 14th 2020

What is active learning?



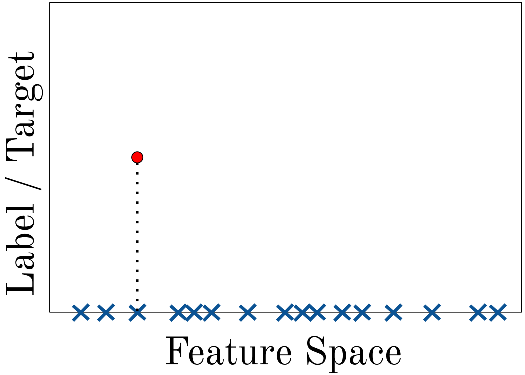
What is active learning?

- Access to **unlabelled** data.



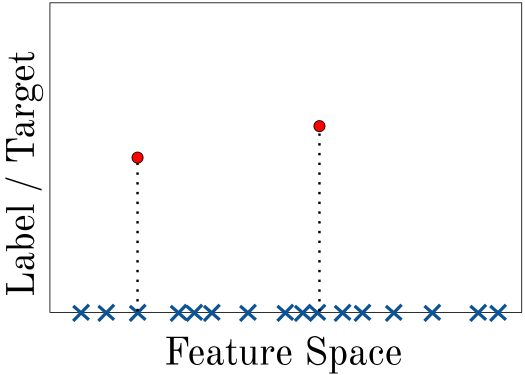
What is active learning?

- Access to **unlabelled** data.
- **Query** an oracle for **labels/targets**.



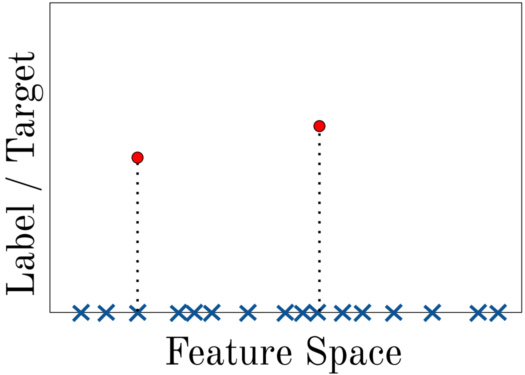
What is active learning?

- Access to **unlabelled** data.
- **Query** an oracle for **labels/targets**.



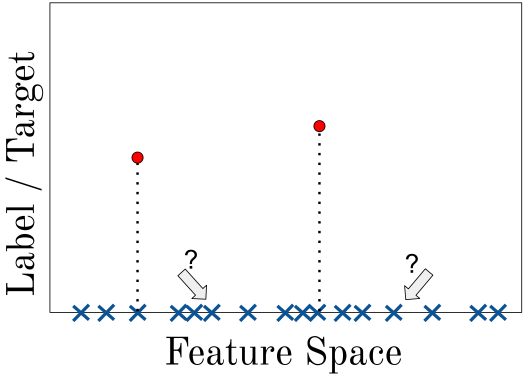
What is active learning?

- Access to **unlabelled** data.
- **Query** an oracle for **labels/targets**. → **Expensive** process.



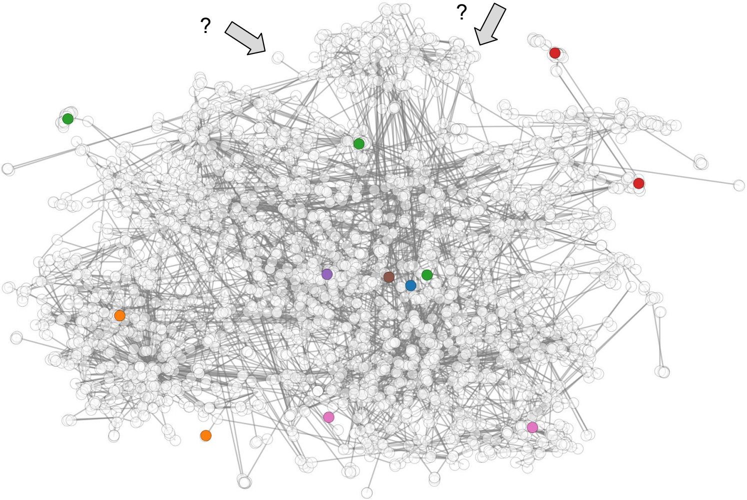
What is active learning?

- Access to **unlabelled** data.
- **Query** an oracle for **labels/targets**. $\rightarrow$ **Expensive** process.



Goal: Choose optimal queries to maximize performance.

Active Learning for Node Classification



Pool-based active learning algorithm steps :

Active Learning Process

Pool-based active learning algorithm steps :

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.

Active Learning Process

Pool-based active learning algorithm steps :

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   Trained on $(\mathbf{X}, \mathbf{Y}_{\mathcal{L}_t})$ **current labelled set** $\mathcal{L}_t$.

Pool-based active learning algorithm steps :

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   Trained on $(\mathbf{X}, \mathbf{Y}_{\mathcal{L}_t})$ **current labelled set** $\mathcal{L}_t$.

2. **QUERY** : Select q from the unlabelled set $\mathcal{U}_t$.

Pool-based active learning algorithm steps :

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   Trained on $(\mathbf{X}, \mathbf{Y}_{\mathcal{L}_t})$ **current labelled set** $\mathcal{L}_t$.

2. **QUERY** : Select q from the unlabelled set $\mathcal{U}_t$.
   Update $\mathcal{L}_{t+1} = \mathcal{L}_t \cup \{q_t\}$ and $\mathcal{U}_{t+1} = \mathcal{U}_t \setminus \{q_t\}$.

Pool-based active learning algorithm steps :

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   Trained on $(\mathbf{X}, \mathbf{Y}_{\mathcal{L}_t})$ **current labelled set** $\mathcal{L}_t$.

2. **QUERY** : Select q from the unlabelled set $\mathcal{U}_t$.
   Update $\mathcal{L}_{t+1} = \mathcal{L}_t \cup \{q_t\}$ and $\mathcal{U}_{t+1} = \mathcal{U}_t \setminus \{q_t\}$.

Repeat until the query budget $\mathcal{B}$ has been reached.

GCN-based models

GCN-based models

SOTA Active leaning strategies based on **GCN output**.
(AGE [1] and ANRMAB [2])

GCN-based models

SOTA Active leaning strategies based on **GCN output**.
(AGE [1] and ANRMAB [2])

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.

GCN-based models

SOTA Active leaning strategies based on **GCN output**.
(AGE [1] and ANRMAB [2])

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   $\rightarrow$ Run one epoch of **GCN**.

GCN-based models

SOTA Active leaning strategies based on **GCN output**.
(AGE [1] and ANRMAB [2])

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   $\rightarrow$ Run one epoch of **GCN**.
   $\rightarrow$ Save the **node embeddings output** from the **GCN**.

GCN-based models

SOTA Active leaning strategies based on **GCN output**.
(AGE [1] and ANRMAB [2])

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   $\rightarrow$ Run one epoch of **GCN**.
   $\rightarrow$ Save the **node embeddings output** from the **GCN**.

2. **QUERY** Select $q \in \mathcal{U}_t$.

GCN-based models

SOTA Active leaning strategies based on **GCN output**.
(AGE [1] and ANRMAB [2])

1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   → Run one epoch of **GCN**.
   → Save the **node embeddings output** from the **GCN**.

2. **QUERY** Select $q \in \mathcal{U}_t$.
   → Select q based on metrics derived from **GCN** output.

GCN-based models

SOTA Active leaning strategies based on **GCN output**.
(AGE [1] and ANRMAB [2])

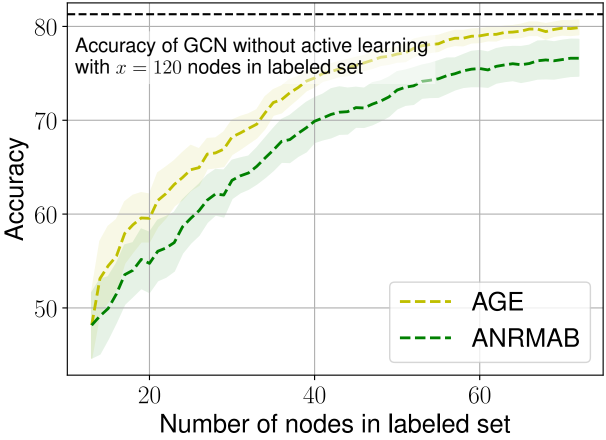
1. **PREDICT** : Infer $\hat{\mathbf{Y}} = f_t(\mathbf{X})$.
   → Run one epoch of **GCN**.
   → Save the **node embeddings output** from the **GCN**.

2. **QUERY** Select $q \in \mathcal{U}_t$.
   → Select q based on metrics derived from **GCN** output.

[1] Cai et al. "Active learning for graph embedding" arXiv 2017
[2] Gao et al. "Active discriminative network representation learning" IJCAI 2018

Existing work - Results

GCN-based algorithms on Cora.



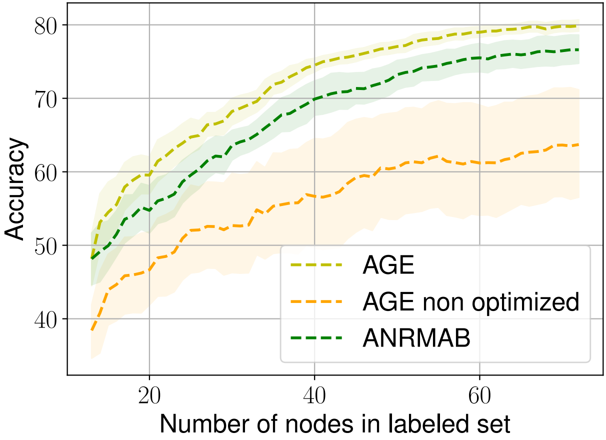
Limitation : Deep learning models generally rely on **sizable validation set** for hyperparameters tuning.

Limitation : Deep learning models generally rely on **sizable validation set** for hyperparameters tuning.

Results with **non-optimized GCN hyperparameter** **highlight this dependence**.

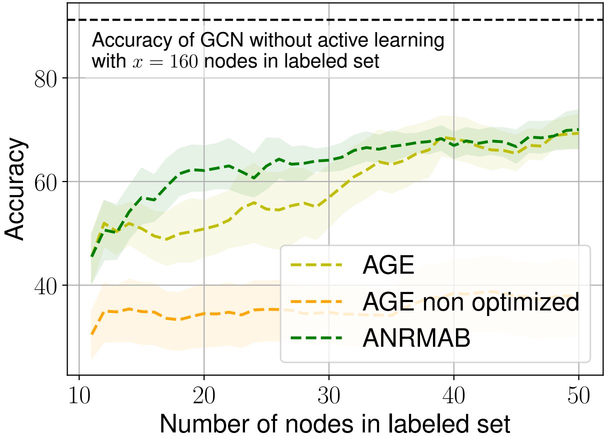
Existing work - Non optimized model

Cora with non-optimized version of **AGE**.



Existing work - Unseen dataset

Amazon-photo. Hyperparameters not fine-tuned to the dataset.



Proposed Algorithm :
Graph Expected Error Minimization (GEEM)

Expected Error Minimization (EEM)

Expected Error Minimization (EEM)

- **Risk** of q : The **expected 0/1 error once added** to $\mathcal{L}_t$.

Expected Error Minimization (EEM)

- **Risk** of q : The **expected 0/1 error once added** to $\mathcal{L}_t$.
- Denoted by $R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$.

Expected Error Minimization (EEM)

- **Risk** of q : The **expected 0/1 error once added** to $\mathcal{L}_t$.
- Denoted by $R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$.
- **EEM** selects the query q that **minimizes** this **risk**.

Expected Error Minimization (EEM)

- **Risk** of q : The **expected 0/1 error once added** to $\mathcal{L}_t$.
- Denoted by $R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$.
- **EEM** selects the query q that **minimizes** this **risk**.

$$q^* = \arg\min_{q \in \mathcal{U}_t} R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$$

Expected Error Minimization (EEM)

- **Risk** of q : The **expected 0/1 error once added** to $\mathcal{L}_t$.
- Denoted by $R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$.
- **EEM** selects the query q that **minimizes** this **risk**.

$$q^* = \arg\min_{q \in \mathcal{U}_t} R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$$

$$R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}} = \sum_{k \in K} \frac{1}{|\mathcal{U}_t^{-q}|} \sum_{i \in \mathcal{U}_t^{-q}} \Big(1 - \max_{k' \in K} p(y_i = k' | Y_{\mathcal{L}_t}, y_q = k)\Big) p(y_q = k | Y_{\mathcal{L}_t})$$

Expected Error Minimization (EEM)

- **Risk** of q : The **expected 0/1 error once added** to $\mathcal{L}_t$.
- Denoted by $R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$.
- **EEM** selects the query q that **minimizes** this **risk**.

$$q^* = \arg\min_{q \in \mathcal{U}_t} R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$$

$$R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}} = \sum_{k \in K} \frac{1}{|\mathcal{U}_t^{-q}|} \sum_{i \in \mathcal{U}_t^{-q}} \left(1 - \max_{k' \in K} p(y_i = k' | Y_{\mathcal{L}_t}, y_q = k)\right) p(y_q = k | Y_{\mathcal{L}_t})$$

Expected Error Minimization (EEM)

- **Risk** of q : The **expected 0/1 error once added** to $\mathcal{L}_t$.
- Denoted by $R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$.
- **EEM** selects the query q that **minimizes** this **risk**.

$$q^* = \arg\min_{q \in \mathcal{U}_t} R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}}$$

$$R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}} = \sum_{k \in K} \frac{1}{|\mathcal{U}_t^{-q}|} \sum_{i \in \mathcal{U}_t^{-q}} \left(1 - \max_{k' \in K} p(y_i = k' | Y_{\mathcal{L}_t}, y_q = k)\right) p(y_q = k | Y_{\mathcal{L}_t})$$

All that remains is to define $p(y|\cdot)$

Proposed Algorithm - $p(y|\cdot)$

All that remains is to define $p(y|\cdot)$

- **Simplified GCN** [3] : **Removes non-linearities** of **GCNs** to obtain a **linearized logistic regression** model.

All that remains is to define $p(y|\cdot)$

- **Simplified GCN** [3] : **Removes non-linearities** of **GCNs** to obtain a **linearized logistic regression** model.
- Set

$$p(y_j = k|\mathbf{Y}_{\mathcal{L}}) = \sigma(\mathbf{\tilde{x}_j}\mathbf{W}_{\mathbf{Y}_{\mathcal{L}}})^{(k)}$$

Proposed Algorithm - $p(y|\cdot)$

All that remains is to define $p(y|\cdot)$

- **Simplified GCN** [3] : **Removes non-linearities** of **GCNs** to obtain a **linearized logistic regression** model.
- Set

$$p(y_j = k|\mathbf{Y}_{\mathcal{L}}) = \sigma(\tilde{\mathbf{x}}_{\mathbf{j}}\mathbf{W}_{\mathbf{Y}_{\mathcal{L}}})^{(k)}$$

- **GEEM** :

$$R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}} = \sum_{k \in K} \frac{1}{|\mathcal{U}_t^{-q}|} \sum_{i \in \mathcal{U}^{-q}} (1 - \max_{k' \in K} \sigma(\tilde{\mathbf{x}}_{\mathbf{i}}\mathbf{W}_{\mathcal{L}_t,+q,y_k})^{(k')})\sigma(\tilde{\mathbf{x}}_{\mathbf{q}}\mathbf{W}_{\mathbf{Y}_{\mathcal{L}_t}})^{(k)}$$

Proposed Algorithm - $p(y|\cdot)$

All that remains is to define $p(y|\cdot)$

- **Simplified GCN** [3] : **Removes non-linearities** of **GCNs** to obtain a **linearized logistic regression** model.
- Set

$$p(y_j = k|\mathbf{Y}_{\mathcal{L}}) = \sigma(\tilde{\mathbf{x}}_\mathbf{j}\mathbf{W}_{\mathbf{Y}_{\mathcal{L}}})^{(k)}$$

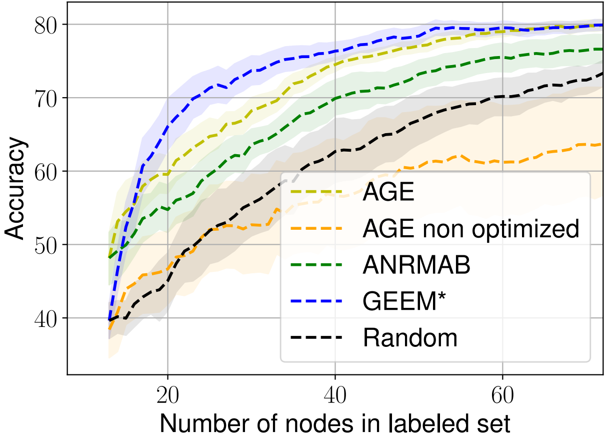
- **GEEM** :

$$R^{+q}_{|\mathbf{Y}_{\mathcal{L}_t}} = \sum_{k\in K} \frac{1}{|\mathcal{U}_t^{-q}|} \sum_{i\in\mathcal{U}^{-q}} (1 - \max_{k'\in K} \sigma(\tilde{\mathbf{x}}_\mathbf{i}\mathbf{W}_{\mathcal{L}_t,+q,y_k})^{(k')})\sigma(\tilde{\mathbf{x}}_\mathbf{q}\mathbf{W}_{\mathbf{Y}_{\mathcal{L}_t}})^{(k)}$$

Results

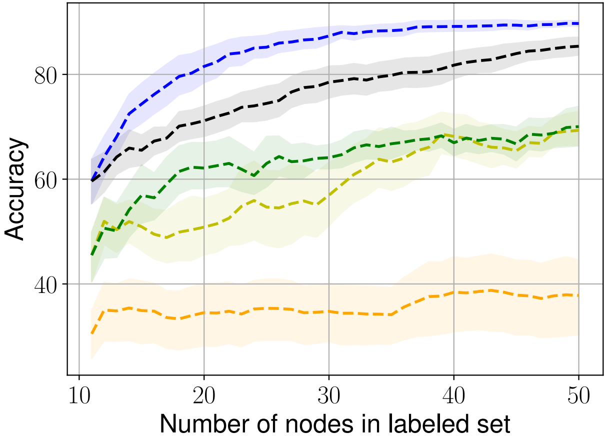
Results - GEEM

Cora. GEEM outperforms GCN-based methods even when GCN hyperparameters are fine-tuned.



Results - GEEM

Amazon-photo. GEEM significantly outperforms GCN-based methods.



Conclusion

The proposed GEEM algorithm:

Conclusion

The proposed GEEM algorithm:

- Offers SOTA performance.

Conclusion

The proposed GEEM algorithm:

- Offers SOTA performance.
- Does not rely on validation set $\rightarrow$ More realistic scenario.

Conclusion

The proposed GEEM algorithm:

- Offers SOTA performance.
- Does not rely on validation set $\rightarrow$ More realistic scenario.

Additional contributions :

Conclusion

The proposed GEEM algorithm:

- Offers SOTA performance.
- Does not rely on validation set $\rightarrow$ More realistic scenario.

Additional contributions :

- **Combined GEEM** : Hybrid mixed with **LP** covers more cases.

Conclusion

The proposed GEEM algorithm:

- Offers SOTA performance.
- Does not rely on validation set $\rightarrow$ More realistic scenario.

Additional contributions :

- **Combined GEEM** : Hybrid mixed with **LP** covers more cases.
- **Preemptive GEEM (PreGEEM)** : Take advantage of **oracle delay** with approximations.

Conclusion

The proposed GEEM algorithm:

- Offers SOTA performance.
- Does not rely on validation set $\rightarrow$ More realistic scenario.

Additional contributions :

- **Combined GEEM** : Hybrid mixed with **LP** covers more cases.
- **Preemptive GEEM (PreGEEM)** : Take advantage of **oracle delay** with approximations.
  $\rightarrow$ Provide bounds on the approximation error.

References

[1] H. Cai, V. W. Zheng, and K. C. Chang, "Active learning for graph embedding," *arXiv preprint arXiv:1705.05085*, 2017.

[2] L. Gao, H. Yang, C. Zhou, J. Wu, S. Pan, and Y. Hu, "Active discriminative network representation learning," in *Proc. Int. Joint Conf. Artificial Intell.*, 2018, pp. 2142–2148.

[3] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. Int. Conf. Machine Learning*, Long Beach, California, USA, Jun. 2019, pp. 6861–6871.