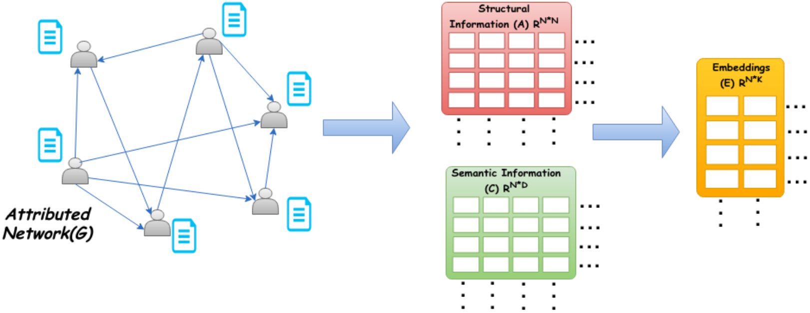
Outlier Aware Network Embedding for Attributed Networks

Sambaran Bandyopadhyay, Lokesh N, M. N. Murty

The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)
January 27, 2019

Attributed Network Embedding

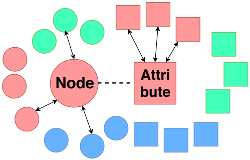


Motivation

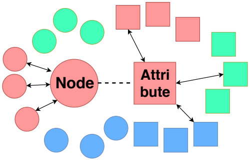
- Attributed network embedding algorithms, in general, exploit homophily property in the topological structure and attributes in a network.
- They perform reasonably well when the network is consistent in its structure and content.
- Unfortunately real world networks are noisy and there are different outliers which even affect the embeddings of normal nodes.
- Hence it is important to minimize the effect of outlier nodes during the generation of embeddings.
- Towards this end, we propose an unsupervised embedding algorithm called **ONE** (**O**utlier aware **N**etwork **E**mbedding) for attributed networks

Notion of an outlier

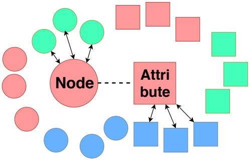
- We define outlier as a node that is



Type 1



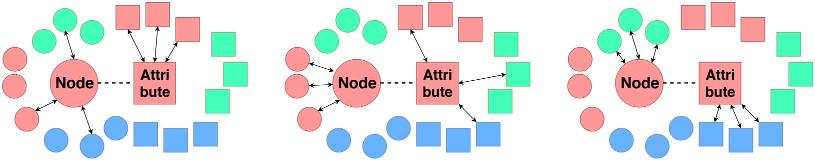
Type 2



Type 3

Notion of an outlier

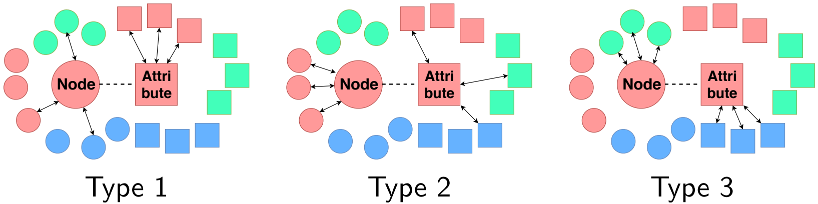
- We define outlier as a node that is
  - Type 1 : Structurally inconsistent, Semantically consistent



Type 1 Type 2 Type 3

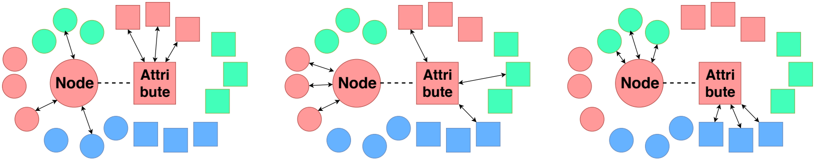
Notion of an outlier

- We define outlier as a node that is
  - Type 1 : Structurally inconsistent, Semantically consistent
  - Type 2 : Semantically inconsistent, Structurally consistent



Notion of an outlier

- We define outlier as a node that is
  - Type 1 : Structurally inconsistent, Semantically consistent
  - Type 2 : Semantically inconsistent, Structurally consistent
  - Type 3 : Structurally and Semantically consistent, but inconsistent across structure and semantics



Type 1 Type 2 Type 3

Problem Statement

- Learn a lower dimensional representation for nodes in an attributed network $\mathcal{G}$ with Structural information $A \in R^{N \times N}$ and Semantic information $C \in R^{N \times D}$
- Mathematically we should learn a function f : $\mathbb{R}^{N \times (N+D)} \mapsto \mathbb{R}^{N \times K}$: $k<N$

Related work

- We categorize Network embedding learning algorithms into two types

Related work

- We categorize Network embedding learning algorithms into two types
- Ones that use just structure
  - DeepWalk
  - Node2Vec
  - Line

Related work

- We categorize Network embedding learning algorithms into two types
- Ones that use just structure
  - DeepWalk
  - Node2Vec
  - Line
- Ones that use both structure and attributes
  - TADW
  - AANE
  - GraphSage
  - **SEANO** - semi-supervised algorithm to consider outliers while generating the embeddings

Our Contributions

- **ONE** - an iterative approach to find lower dimensional embedding of the nodes, such that the **outliers contribute less** to the overall cost function
- First work to propose a **completely unsupervised algorithm** for attributed network embedding integrated with outlier detection, also we propose a novel method to combine structure and attributes efficiently.
- **Experimentation** on the outlier seeded versions of publicly available network datasets to show the efficiency of our approach to **detect outliers**, as well as on network mining tasks such as **node clustering** and **classification**.

ONE - Solution Approach

- Our solution primarily has three components

ONE - Solution Approach

- Our solution primarily has three components
- Learning from Structure
  - $\mathcal{L}_{str} = \sum_{i=1}^{N} \sum_{j=1}^{N} \log\left(\frac{1}{O_{1i}}\right)(A_{ij} - G_{i\cdot} \cdot H_{\cdot j})^2$
  - $\sum_i o_{1i} = 1$
  - Essentially we factorize structure matrix A into two matrices G, H

ONE - Solution Approach

- Our solution primarily has three components
- Learning from Structure
  - $\mathcal{L}_{str} = \sum_{i=1}^{N} \sum_{j=1}^{N} \log\left(\frac{1}{O_{1i}}\right)(A_{ij} - G_{i\cdot} \cdot H_{\cdot j})^2$
  - $\sum_i o_{1i} = 1$
  - Essentially we factorize structure matrix A into two matrices G, H
- Learning from attributes
  - $\mathcal{L}_{attr} = \sum_{i=1}^{N} \sum_{d=1}^{C} \log\left(\frac{1}{O_{2i}}\right)(C_{id} - U_{i\cdot} \cdot V_{\cdot j})^2$
  - $\sum_i o_{2i} = 1$
  - Essentially we factorize attribute matrix C into two matrices U, V

ONE - Solution Approach

- Our solution primarily has three components
- Learning from Structure
  - $\mathcal{L}_{str} = \sum_{i=1}^{N} \sum_{j=1}^{N} \log\left(\frac{1}{O_{1i}}\right)(A_{ij} - G_{i\cdot} \cdot H_{\cdot j})^2$
  - $\sum_i o_{1i} = 1$
  - Essentially we factorize structure matrix A into two matrices G, H
- Learning from attributes
  - $\mathcal{L}_{attr} = \sum_{i=1}^{N} \sum_{d=1}^{C} \log\left(\frac{1}{O_{2i}}\right)(C_{id} - U_{i\cdot} \cdot V_{\cdot j})^2$
  - $\sum_i o_{2i} = 1$
  - Essentially we factorize attribute matrix C into two matrices U, V
- o_{*i} reflects the outlierness of a node

ONE - Solution Approach

- We provide novel solutions for the following two problems in the loss formulation
  - Problem 1 : Type-3 outliers must be accounted for
  - Problem 2 : The dimensions (features) of G and U should be aligned

ONE - Solution Approach

- We provide novel solutions for the following two problems in the loss formulation
  - Problem 1 : Type-3 outliers must be accounted for
  - Problem 2 : The dimensions (features) of G and U should be aligned
- Solution to Problem 1
  - To this end, we add coupling component in the loss function
  - $\mathcal{L}_{combined} = \sum_{i=1}^{N} \sum_{k=1}^{K} \log\left(\frac{1}{O_{3i}}\right)\left(G_{ik} - U_{i\cdot}\right)^2$
  - We are essentially trying to make the embeddings from structure and content consistent with each other

ONE - Solution Approach

- We provide novel solutions for the following two problems in the loss formulation
  - Problem 1 : Type-3 outliers must be accounted for
  - Problem 2 : The dimensions (features) of G and U should be aligned
- Solution to Problem 1
  - To this end, we add coupling component in the loss function
  - $\mathcal{L}_{combined} = \sum_{i=1}^{N} \sum_{k=1}^{K} \log\left(\frac{1}{O_{3i}}\right)\left(G_{ik} - U_{i\cdot}\right)^2$
  - We are essentially trying to make the embeddings from structure and content consistent with each other
- Solution to Problem 2
  - To align the dimensions we add a Linear transformation matrix in the third loss component
  - $\mathcal{L}_{combined} = \sum_{i=1}^{N} \sum_{k=1}^{K} \log\left(\frac{1}{O_{3i}}\right)\left(G_{ik} - U_{i\cdot} \cdot (W^T)_{\cdot k}\right)^2$
  - Constraint : W should be orthogonal as scaling of dimensions is not appreciable

ONE - Solution Approach

- **Final loss function** is $\mathcal{L} = \mathcal{L}_{str} + \alpha\mathcal{L}_{attr} + \beta\mathcal{L}_{combined}$.

ONE - Solution Approach

- **Final loss function** is $\mathcal{L} = \mathcal{L}_{str} + \alpha\mathcal{L}_{attr} + \beta\mathcal{L}_{combined}$.
- Except for W, we obtain closed form rules for all the variables by equating gradient of the Lagrangian to zero.

ONE - Solution Approach

- **Final loss function** is $\mathcal{L} = \mathcal{L}_{str} + \alpha\mathcal{L}_{attr} + \beta\mathcal{L}_{combined}$.
- Except for W, we obtain closed form rules for all the variables by equating gradient of the Lagrangian to zero.
- We use the solution concept from **Procrustes problem** to derive a closed form update rule for W. In Procrustes problem in Linear Algebra, we are asked to find a matrix R such that $A = RB$ subject to the condition that $R^T R = \mathcal{I}$

ONE - Solution Approach

- **Final loss function** is $\mathcal{L} = \mathcal{L}_{str} + \alpha\mathcal{L}_{attr} + \beta\mathcal{L}_{combined}$.
- Except for W, we obtain closed form rules for all the variables by equating gradient of the Lagrangian to zero.
- We use the solution concept from **Procrustes problem** to derive a closed form update rule for W. In Procrustes problem in Linear Algebra, we are asked to find a matrix R such that $A = RB$ subject to the condition that $R^T R = \mathcal{I}$
- **The joint cost function $\mathcal{L}$ decreases after each iteration of update rules of ONE.**

ONE - Solution Approach

- **Final loss function** is $\mathcal{L} = \mathcal{L}_{str} + \alpha\mathcal{L}_{attr} + \beta\mathcal{L}_{combined}$.
- Except for W, we obtain closed form rules for all the variables by equating gradient of the Lagrangian to zero.
- We use the solution concept from **Procrustes problem** to derive a closed form update rule for W. In Procrustes problem in Linear Algebra, we are asked to find a matrix R such that $A = RB$ subject to the condition that $R^T R = \mathcal{I}$
- **The joint cost function $\mathcal{L}$ decreases after each iteration of update rules of ONE.**
- Output representation is $\frac{G+UW^T}{2}$

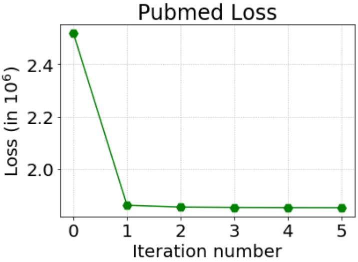
Datasets

Summary of the datasets (after planting outliers)

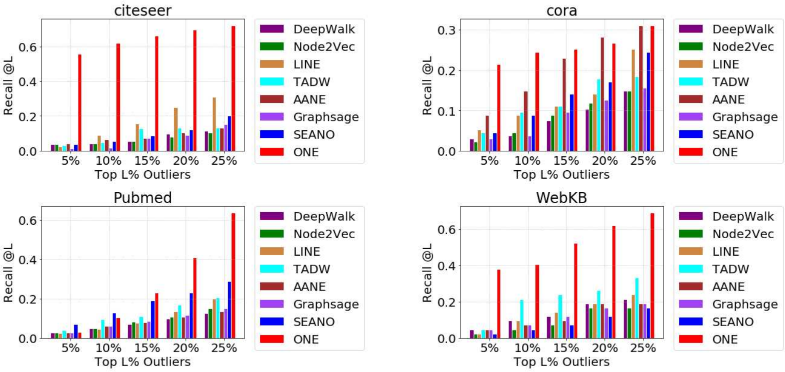
Dataset	#Nodes	#Edges	#Labels	#Attributes
WebKB	919	1662	5	1703
Cora	2843	6269	7	1433
Citeseer	3477	5319	6	3703
Pubmed	20701	49523	3	500

- We synthesize 5% outliers, with equal numbers for each of the 3 outlier types

Beauty of closed form solutions



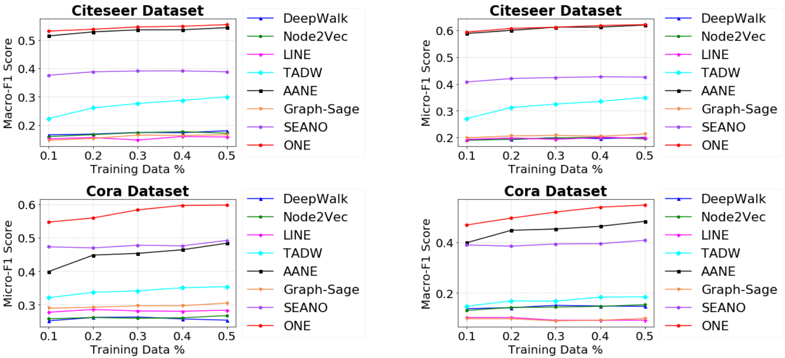
Outlier Detection Perfromance

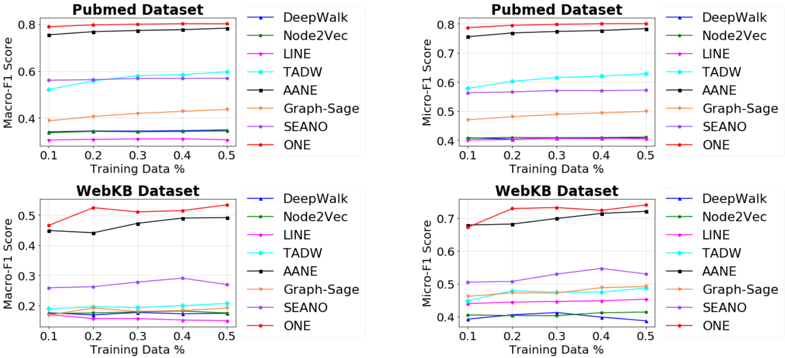


Note : ONE and SEANO have explicit outlier interpretations readily available. For rest of the algorithms we train the embeddings first and then run Isolation Forest on the embeddings to generate the outlier scores.

Classification Performance

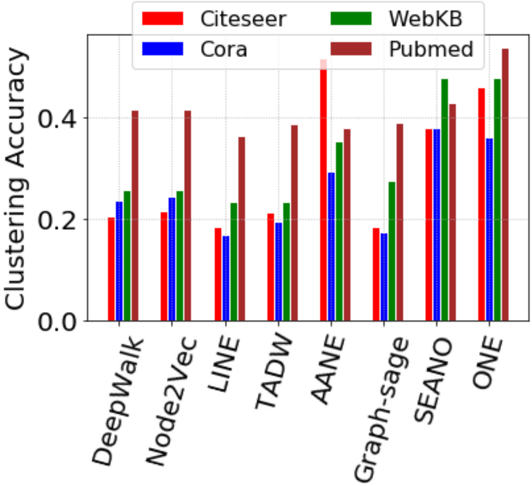
All results reported are obtained by running Random Forest algorithm on the embeddings generated by the algorithms for classification





Pubmed Dataset
Macro-F1 Score
Training Data %
Pubmed Dataset
Micro-F1 Score
Training Data %
WebKB Dataset
Macro-F1 Score
Training Data %
WebKB Dataset
Micro-F1 Score
Training Data %
DeepWalk
Node2Vec
LINE
TADW
AANE
Graph-Sage
SEANO
ONE

Clustering Perfromance



Conclusion

- We proposed ONE, which learns robust embeddings of nodes in an attributed network with outliers
- Even though, the loss function involves computation of $O(N^2)$ terms, the loss values take less than 3-4 iterations to converge
- In future we wish to:
  - approximate the update rules of the variables to make it computationally faster
  - extend it for dynamic networks