About me

- **Name**: 李博浪
- **Student ID**: 21821210
- **Topic**: Neural Networks

Schedule

Task	Due	Done
1.选择论文	Mar.14	Y
2.精读论文	Mar.21	Y
3.复现论文	Apr.4	N
4. 完成实验	Apr.11	N
5.撰写报告	Apr.18	N

选择论文

[Deep Neural Networks for Learning Graph Representations AAAI 2016]

- **摘要** > 本文提出了一种新的模型,通过捕获图的结构特征,从而能够学习图的表示,为图中每个节点生成低维向量.与以往的基于采样的模型不同,DNGR主要通过随机搜索从而直接捕获图的结构信息.本文主要使用堆叠去噪自编码器去抽取PMI矩阵中的复杂的非线性特征.为了证明模型的有效性,作者利用了模型所学习到的节点的向量,来完成聚类及可视化任务等. ### 精读论文 >本文通过捕获图的特征结构来学习图的表示,为图中每个节点生成低维向量.本文可用于带权图,而且还能捕获图中的非线性关系.
  本文算法主要分为三个步骤:
  1.使用random surfing模型捕获图的结构,并获得过线概率矩阵PCO
  2.基于PCO来计算PPMI矩阵
  3.利用堆叠去噪自编码器来学习节点的低维表示
  1.random surfing
  $p_{k}=\alpha p_{k-1}A+(1-\alpha)p_{0}$
  $r=\sum_{k=1}^{K}p_{k}$
  即通过转移概率矩阵来求得节点的k跳所能到达节点的概率,然后通过加权和获得节点的概率共现矩阵PCO.
  2.基于PCO来计算PPMI矩阵
  $PMI_{w,c}=log(\frac{\#(w,c)\cdot \left| D \right|}{\#(w)\cdot \#(c)})$
  $$PPMI_{w,c}=max(PMI_{w,c},0)$$
  通过节点的概率共现矩阵获得PPMI矩阵

3.利用堆叠去噪自编码器来学习节点的低维表示
将PPMI矩阵中节点对应的向量作为输入,放入SDAE里,获取节点的低维向量表示.并使用逐层的贪婪预训练.最终获得节点的低维向量表示

模型结构与实验结果

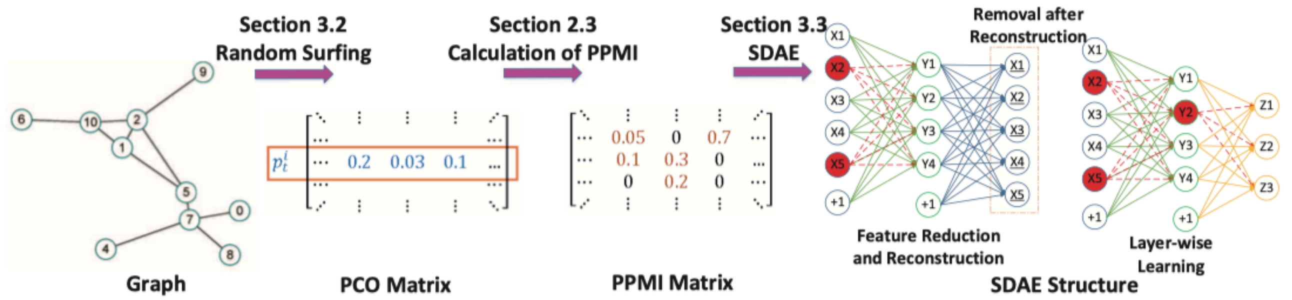
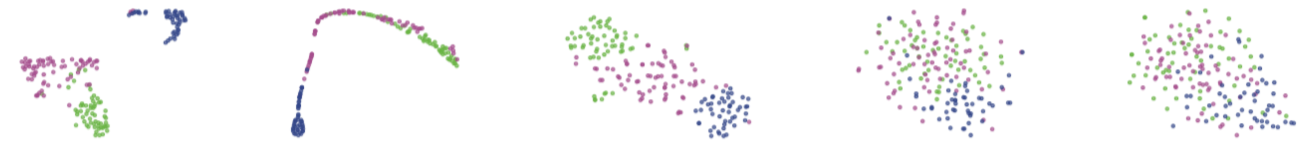


Figure 1: Main components: random surfing, calculation of PPMI matrix and feature reduction by SDAE



(a) DNGR (α=0.98) (b) DNGR(α=1) (c) SVD (α=0.98) (d) DeepWalk (e) SGNS

Figure 3: Visualization Results of Wine. Vector representations serve as features to feed into t-SNE tool and points in the same color indicate instances of the same type.

复现论文

random surfing

```
def random_surfing(matrix,alpha=0.98,steps=10):
    nodes_count=len(matrix)
    p0=np.eye(nodes_count,dtype='float64')
    result=np.zeros((nodes_count,nodes_count),dtype='float64')
    p=p0.copy()
    for i in range(steps):
        p=alpha*np.dot(p,matrix)+(1-alpha)*p0
        result=result+p
    #result=result/result.sum(axis=1).reshape(-1,1)
    result=scale_sim_mat(result)
    return result
```

PPMI

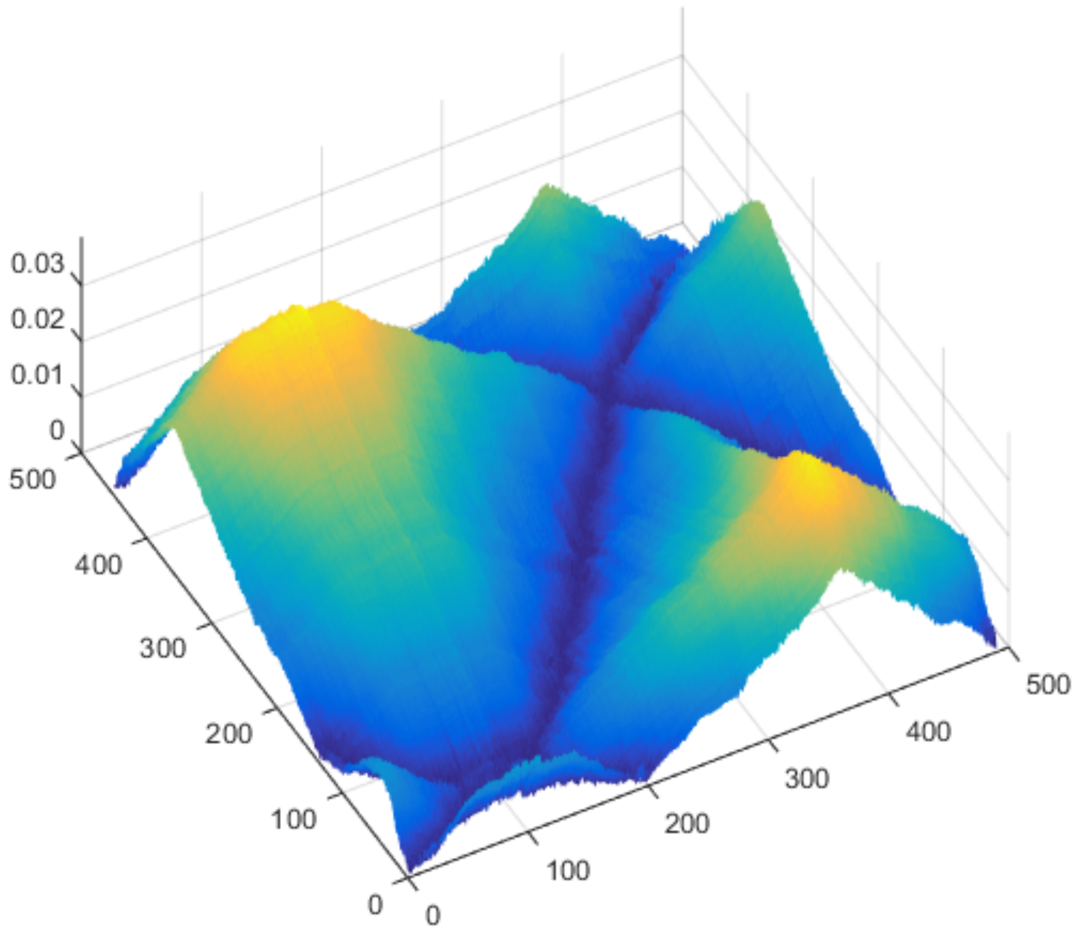
```
def get_PPMI(matrix):

    node_counts=len(matrix)
    D=matrix.sum()
    row_sum=matrix.sum(axis=1).reshape(-1,1)
    column_sum=matrix.sum(axis=0).reshape(1,-1)
    PPMI=np.log(np.divide(D*matrix,np.dot(row_sum,column_sum)))
    PPMI[np.isnan(PPMI)]=0
    PPMI[np.isinf(PPMI)] = 0.0
    PPMI[np.isneginf(PPMI)] = 0.0
    PPMI[PPMI<0]=0
    return PPMI
```

train

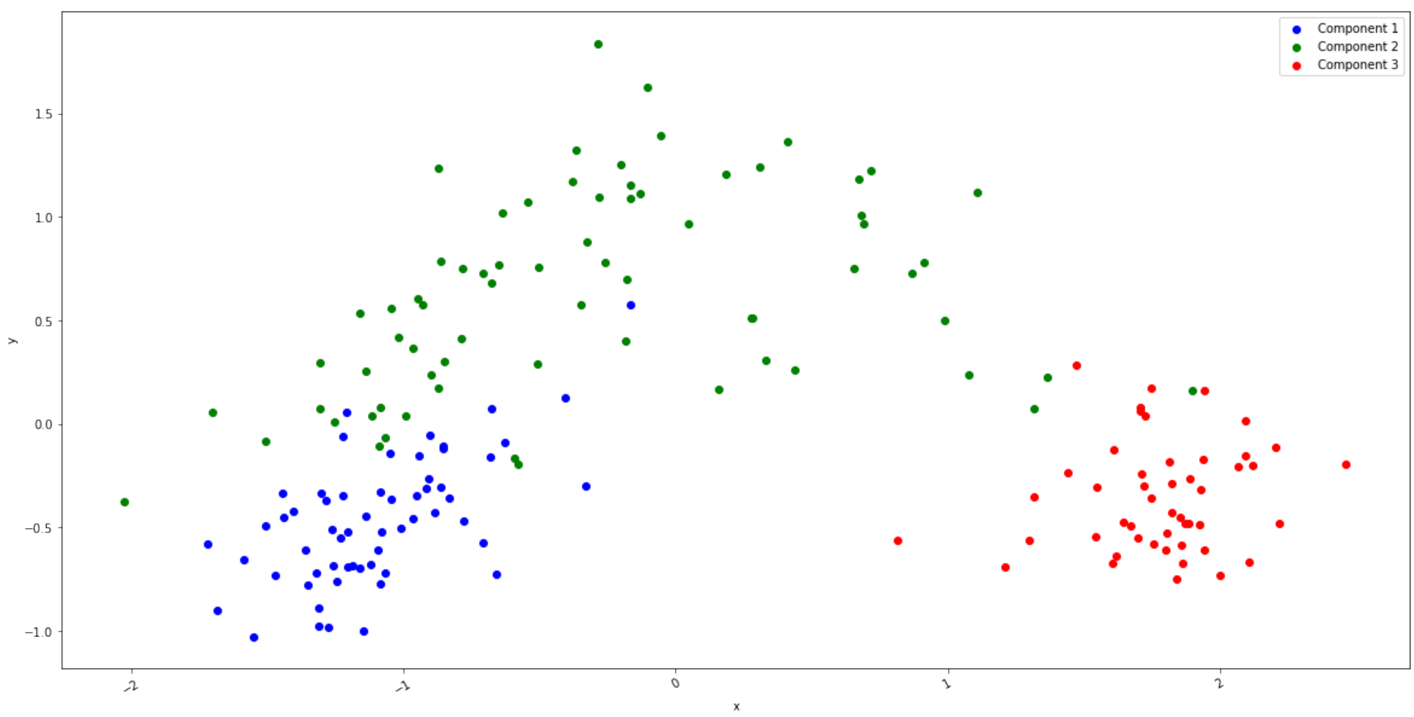
```
def train(data,hidden_layer=3,hidden_neurons=[128,64,32]):
    input_layer=Input(shape=(data.shape[1],))
    encoder=noise.GaussianNoise(0.2)(input_layer)
    for i in range(hidden_layer):
        encoder=Dense(hidden_neurons[i],activation='relu')(encoder)
        encoder=noise.GaussianNoise(0.2)(encoder)
    decoder=Dense(hidden_neurons[-2],activation='relu')(encoder)
    for j in range(hidden_layer-3,-1,-1):
        decoder=Dense(hidden_neurons[j],activation='relu')(decoder)
    decoder=Dense(data.shape[1],activation='sigmoid')(decoder)

    sdae=Model(input=input_layer,output=decoder)
    sdae.summary()
    sdae.compile(optimizer='adadelta',loss='mse')
    sdae.fit(data,data,steps_per_epoch=20,epochs=10)
    return Model(input=input_layer,output=encoder)
```

完成实验

loss仿真


0.03
0.02
0.01
0
500
400
300
200
100
0
0
100
200
300
400
500

分类结果


Component 1
Component 2
Component 3
1.5
1.0
0.5
0.0
−0.5
−1.0
y
−2
−1
0
1
2
x

实验结果

Table 3: Clustering Results on 20-NewsGroup

NMI (%)	3NG	6NG	9NG
DNGR(α=0.98)	**80.44**	68.76	**59.13**
DNGR(α=0.95)	74.30	**68.87**	57.32
DNGR(α=1)	74.63	66.70	58.22
SVD (α=0.98)	62.95	66.18	52.90
SVD (α=0.95)	62.83	65.33	50.00
SVD (α=1)	60.27	63.52	53.52
PPMI	62.84	66.52	52.30
DeepWalk (128)	55.83	58.11	46.86
DeepWalk (256)	56.34	58.40	46.62
DeepWalk (512)	59.64	59.82	46.29
SGNS (128)	64.34	62.99	48.14
SGNS (256)	62.56	63.25	48.10
SGNS (512)	63.97	63.16	49.04