Complex network in text mining

Alneu de Andrade Lopes and Alan Valejo

University of São Paulo (campus at São Carlos)

Laboratório

Algoritmos

- IMBHN (supervisionado)
- TPBG (semi-supervisionado)
- PBG (não supervisionado)

Ferramentas

- Python
- Scikit Learn
- NLTK

Tarefas

- Kfold e supervisionado
- Semi-supervisionado e número de rótulos
- Não supervisionado, pré-processamento e tópicos

Dados

- Sintéticos
- Reais

Tipos de dados

- Arff
- Ncol
- Coleções de documentos

BNOC - Dados sintéticos

Valejo, Alan and Goes, F. and Romanetto, L. M. and Oliveira, Maria C. F. and Lopes, A. A., A benchmarking tool for the generation of bipartite network models with overlapping communities, in *Knowledge and information systems*, accepted paper, 2019
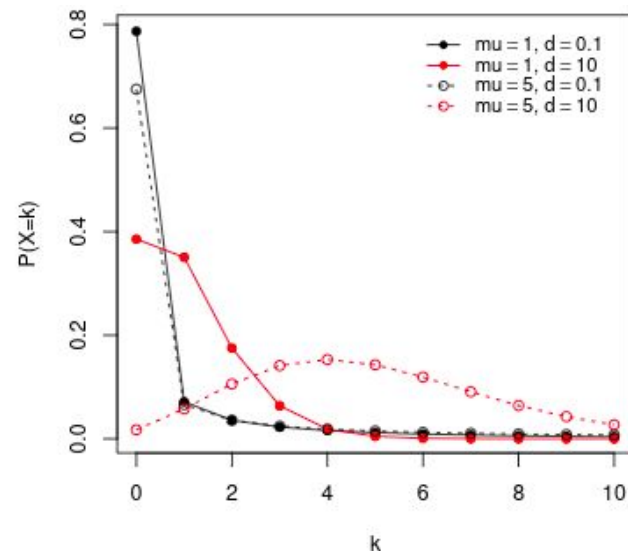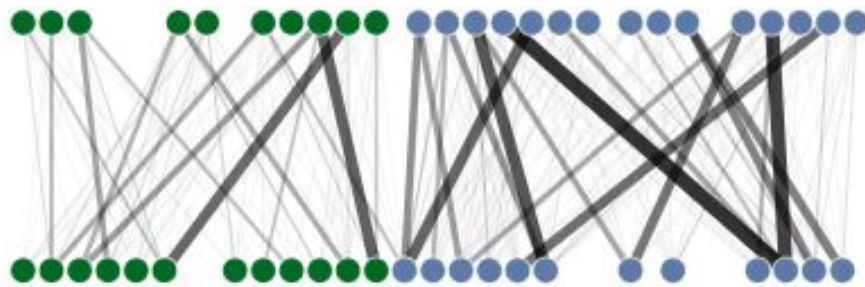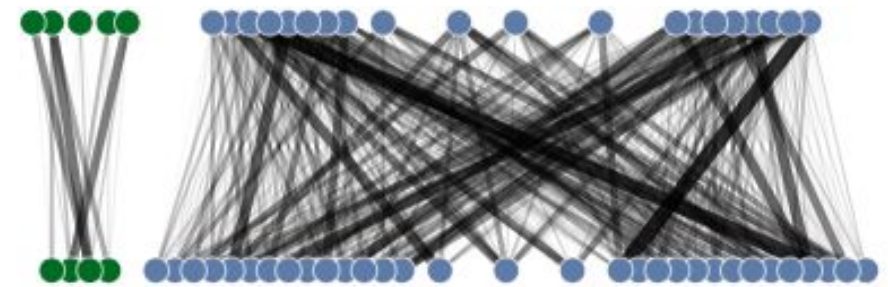
Fig. 1 Negative binomial distribution for distinct values of parameters *mu* and *d*.

Uma distribuição de probabilidade discreta que representa o número de possíveis falhas em uma sequência de ensaios de Bernoulli antes de atingir um número alvo de sucessos.
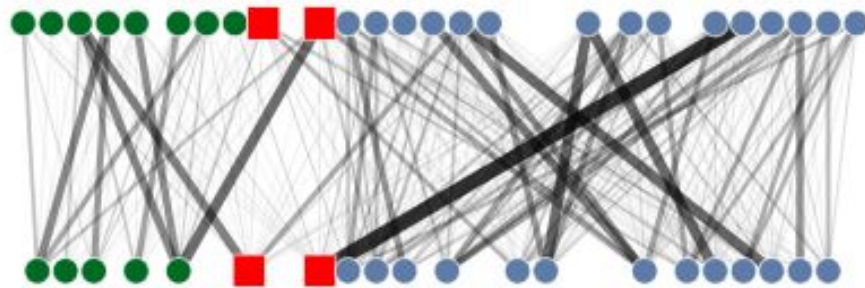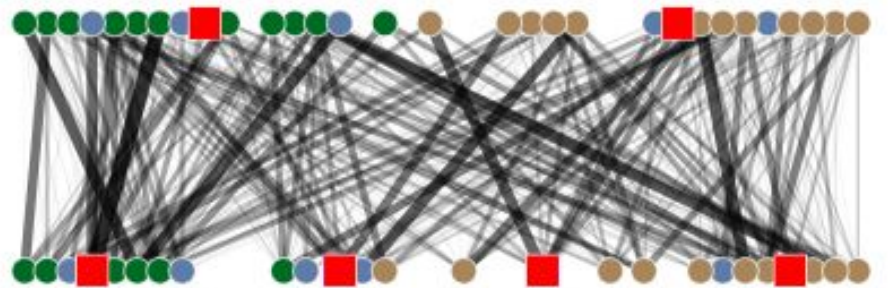
42

BNOC - Dados sintéticos

(a) $v = [25, 15]$, $c = [2, 2]$, $d = 0.5$, b

(b) $v = [35, 25]$, $c = [2, 2]$, $d = 0.9$, $p0 = [0.2, 0.8]$, $p0 = p1$

(c) $v = [20, 25]$, $c = [2, 2]$, $p0 = [0.4, 0.6]$, $p0 = p1$, $x = 2$, $y = 2$, $z = 2$, $d = 0.6$

(d) $v = [25, 30]$, $c = [2, 2]$, $d = 0.8$, $x = 4$, $y = 2$, $z = 2$, $n = 0.2$, b

43

BNOC - Dados sintéticos

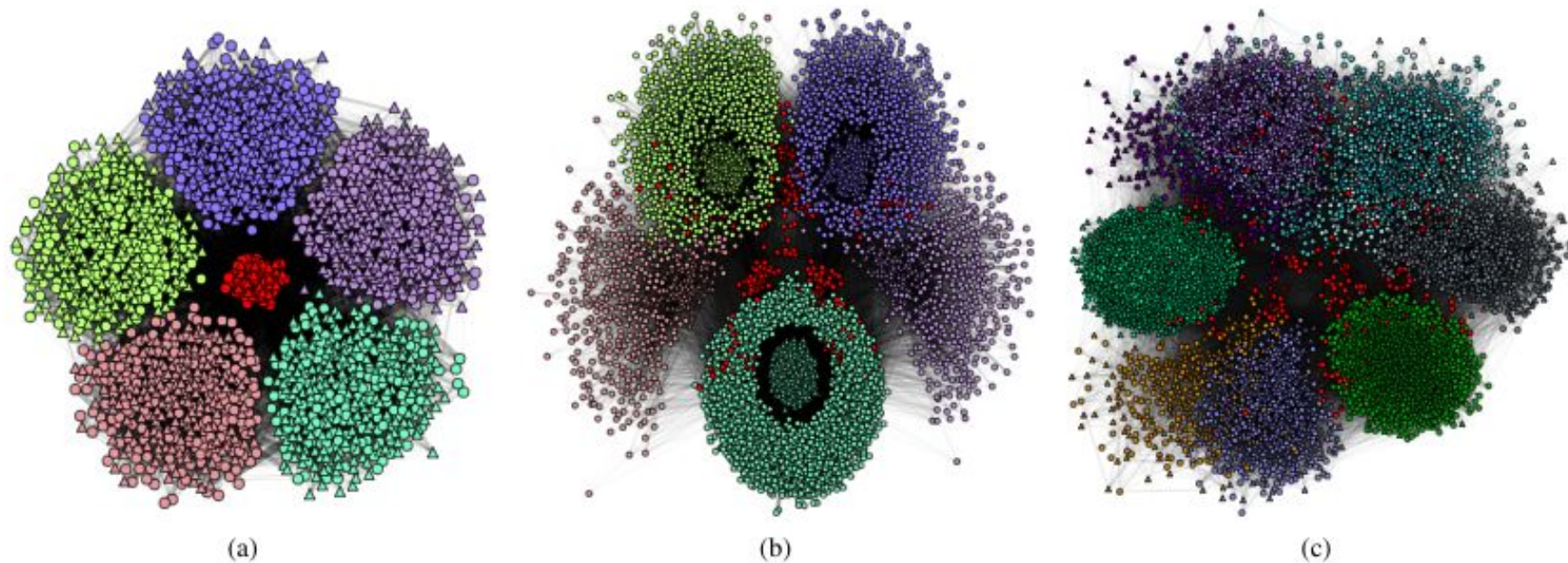(a)   (b)   (c)

Fig. 7 Bipartite networks generated with BNOC illustrating the effect of varying parameter z, which controls the degree of community overlapping. Red markers depict overlapping vertices, whereas other colors indicate the assigned communities of the non-overlapping vertices. **(a)** a network built with five communities ($c = [5, 5]$) and $z = 5$ (strong overlapping); **(b)** a network built with $c = [5, 5]$, $z = 3$, and $x = y = 80$; **(c)** a network built with $c = [10, 10]$, $z = 2$, and $x = y = 150$.
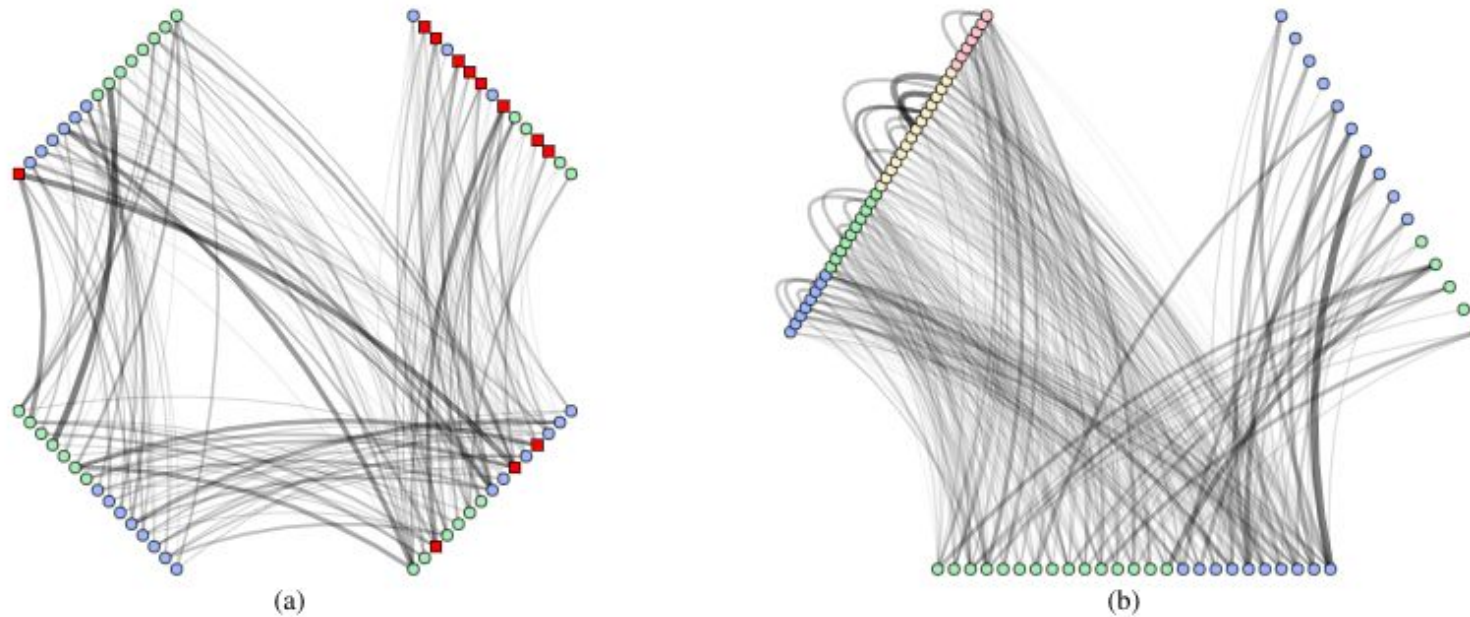
BNOC - Dados sintéticos

Fig. 20 Heterogeneous networks generated with HNOC presenting distinct topological structures and properties: red squares depict overlapping vertices and colored circles indicate non-overlapping vertices and their assigned community; line widths reflect the corresponding edge weights. (a) illustrates a 4-partite network with $v = [15, 15, 15, 15]$, $e = [(0, 1), (1, 2), (2, 3), (3, 1)]$ and $x = [8, 3, 0, 1]$; (b) depicts a heterogeneous network obtained with settings $v = [40, 25, 15]$, $e = [(0, 1), (1, 2), (2, 2)]$, $c = [2, 2, 4]$, and $d = [0.45, 0.85, 0.15, 0.15]$. The network drawings were obtained based on the technique described by Uslu and Mehler (2018).

Coleções de texto reais

CSTR

- Composta por resumos e relatórios técnicos publicados no Departamento de Ciência da Computação da Universidade de Rochester, de 1991 a 2007. Os documentos pertencem a 4 áreas: Processamento de Linguagem Natural, Robótica/Visão, Sistemas e Teoria

Documents	Terms	Terms	Systems	Theory	Robotics	ArtificialIntelligence
299	1726	54.27	25	46	100	128

SyskillWebert

- Composta por páginas da web com assuntos variados, desde bandas e músicas até textos da área de biomedicina.

Documents	Terms	Terms	Bands	Sheep	Goats	Biomedical
334	4340	93.16	61	65	71	137

Formatos de dados

.ncol

vertice   vertice   peso

vertice   vertice   peso

vertice   vertice   peso

vertice   vertice   peso

….…...

Formatos de dados

.arff (weka)

@relation CSTR

@attribute plastic numeric

....

@attribute class_atr {Theory,ArtificiallIntelligence,Robotics,Systems}

@data

0,1,0, …, Theory

3,0,0, …, ArtificiallIntelligence

…

Formatos de dados

texto

Códigos

Github

https://github.com/alanvalejo/icmc2019bigdata

Instalação

Máquina virtual "Matemática"

Terminal Anaconda

$ pip install unidecode

$ pip install pypdf2

$ pip install pandas

$ pip install scikit-learn

$ pip install nltk

$ python

>> import nltk

>> nltk.download('stopwords')

CTRl+d

Experimento 1

Dados sintéticos usando o BNOC

- Simular redes bipartidas sintéticas de documentos e termos, com características topológicas simples ou complexas

Scripts

- $ python bnoc.py -cnf input/input_document_term_easy.json
- $ python bnoc.py -cnf input/input_document_term_hard.json
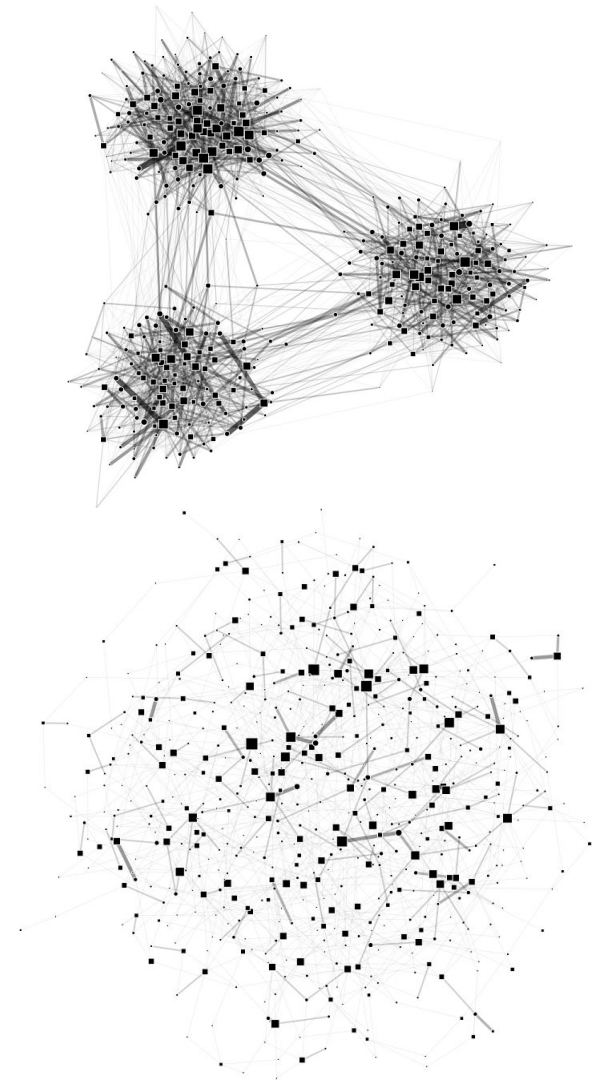
Experimento 2

Problema supervisionado: Classificação de documentos

- IMBHN

Conceitos

- kfold
- sklearn

Script

- $ python imbhn-bnoc-supervised.py

Experimento 2

```
>>> import numpy as np
>>> from sklearn.model_selection import KFold
>>> X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
>>> y = np.array([1, 2, 3, 4])
>>> kf = KFold(n_splits=2)
>>> kf.get_n_splits(X)
2
>>> print(kf)
KFold(n_splits=2, random_state=None, shuffle=False)
>>> for train_index, test_index in kf.split(X):
...     print("TRAIN:", train_index, "TEST:", test_index)
...     X_train, X_test = X[train_index], X[test_index]
...     y_train, y_test = y[train_index], y[test_index]
TRAIN: [2 3] TEST: [0 1]
TRAIN: [0 1] TEST: [2 3]
```

Experimento 3

Problema semi-supervisionado

- TPBG

Conceitos

- Quantidade de dados rotulados

- scipy

- sklearn

Scripts

- $ python pbg-bnoc-semi-supervised.py (base sintética)
- $ python pbg-cstr-semi-supervised.py (base real)

Experimento 3

- Pré-processamento
  - Transformar .ncol em matriz esparsa

.ncol

vertice	vertice	peso
vertice	vertice	peso
vertice	vertice	peso
vertice	vertice	peso

....

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 6 & 0 \\ 0 & -3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 \\ 5 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Experimento 4

Problema não supervisionado: Encontrar tópicos em coleções de documentos

- PBG
- LDA

Conceitos

- Pré-processamento
- NLTK
- sklearn

Scripts

- $ python pbg-syskillwebert-unsupervised.py

Experimento 4

Script: pbg-syskillwebert-unsupervised.py, linha 18

d = l.from_files('input/parsed/syskillwebert')

d = l.from_files('C:\\Users\\ICMC\\Downloads\\icmc2019bigdata-master\\pbg\\input\\parsed\\syskillwebert')

Pré-processamento (NLTK e sklearn)

- Remover stopwords ('and', ',' ...)
- Stemmers (remover plurais, gênero ..., manter apenas o radical)
- Regular expression operations
- Rede bipartida
  - CountVectorize: Cria um vetor de vocabulário e a frenquência de cada palavra em cada documento

Experimento 4

```
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> corpus = [
...     'This is the first document.',
...     'This document is the second document.',
...     'And this is the third one.',
...     'Is this the first document?',
... ]
>>> vectorizer = CountVectorizer()
>>> X = vectorizer.fit_transform(corpus)
>>> print(vectorizer.get_feature_names())
['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third', 'this']
>>> print(X.toarray())
[[0 1 1 1 0 0 1 0 1]
 [0 2 0 1 0 1 1 0 1]
 [1 0 0 1 1 0 1 1 1]
 [0 1 1 1 0 0 1 0 1]]
```

Thank you!