Graph Clustering for Natural Language Processing

Tutorial at AINL 2018

doi: 10.5281/zenodo.1161505

About Me

Dr. **Dmitry Ustalov**

- Post-Doctoral Researcher at the University of Mannheim, Germany
- Research Interests: Crowdsourcing, Computational Lexical Semantics

Outline

Introduction

- Natural Language Processing (NLP) focuses on *analysis* and synthesis of natural language
- Linguistic phenomena instantinate in linguistic data, showing interconnections and relationships
- **Graph clustering**, as an *unsupervised learning* technique, captures the *implicit structure* of the data
- In this tutorial, we will learn how to do it!

Core Idea: Graphs are a Representation

After constructing it explicitly, we can extract useful knowledge from it.

Successful Applications

Graph clustering helps in addressing very challenging NLP problems:

- word sense induction (Biemann, 2006)
- cross-lingual semantic relationship induction (Lewis et al., 2013)
- unsupervised term discovery (Lyzinski et al., 2015)
- making sense of word embeddings (Pelevina et al., 2016)
- text summarization (Azadani et al., 2018)
- entity resolution from multiple sources (Tauer et al., 2019)

Other well-known applications of graph-based methods (not clustering):

- **PageRank**, a citation-based ranking algorithm (Page et al., 1999)
- **BabelNet**, a multilingual semantic network (Navigli et al., 2012)

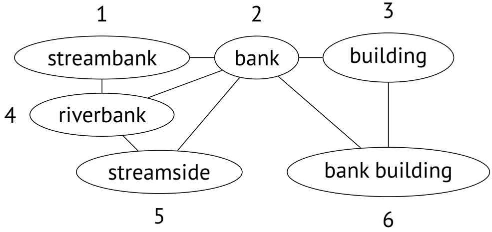
Graph Theory Recap I

- A graph is a tuple $G = (V, E)$, where V is a set of objects called *nodes* and $E \subseteq V^2$ is a set of pairs called *edges*
- Graphs can be undirected (edges are unordered) or directed (edges are called *arcs*)
  - The maximal number of edges in an *undirected* graph is $\frac{|V|(|V|-1)}{2}$
  - The maximal number of arcs in a *directed* graph is $|V|(|V|-1)$
- Graphs can be *weighted*, i.e., there is $w : (u, v) \to \mathbb{R}, \forall (u, v) \in E$
- A neighborhood $G_u = (V_u, E_u)$ is a subgraph induced from G containing the nodes *incident* to $u \in V$ without u

Graph Theory Recap II

- There is a lot of ways to represent a graph, the most common is *adjacency matrix* $A_{i,j} = \mathbb{1}_E(V_i, V_j)$:

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$



- Sparse matrices can be efficiently represented in such formats as CSC (Duff et al., 1989), CSR (Buluç et al., 2009), etc.
- A node *degree* is the number of nodes incident to this node, e.g., $\deg(\text{riverbank}) = 3$; the maximal degree Δ in this graph is 5
- In a directed graph, $\operatorname{succ}(u) \subset V$ is a set of *successors*, which are the nodes reachable from $u \in V$

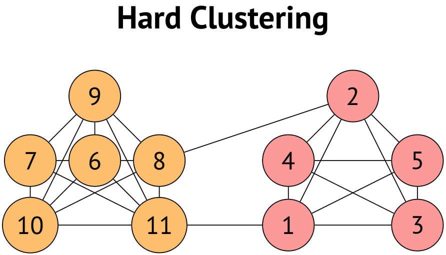
Graph Clustering: Problem Formulation

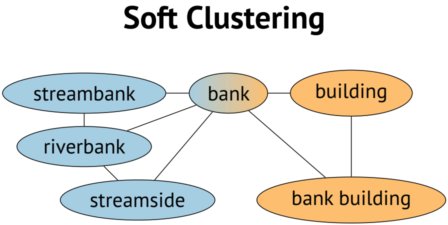
- So, given an *undirected* graph $G = (V, E)$, we are interested in obtaining a set cover for V called *clustering* C of this graph:

$$V = \bigcup_{C^i \in C} C^i$$

- *Hard* clustering algorithms (partitionings) produce non-overlapping clusters: $C^i \cap C^j = \emptyset \iff i \neq j, \forall C^i, C^j \in C$
- *Soft* clustering algorithms permit cluster overlapping, i.e., a node can be a member of several clusters: $\exists u \in V : |C^i \in C : u \in C^i| > 1$
- Like in other unsupervised learning tasks, similar objects are expected to be close, while non-similar are not
- Every algorithm defines what good clustering is

Graph Clustering: Example





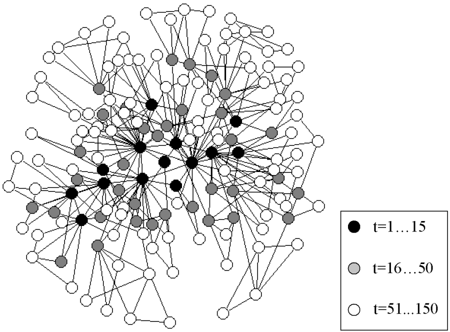
Can We Trust Graphs?

Graphs representing linguistic phenomena exhibit **small world** properties (Biemann, 2012):

- *co-occurrence networks* tend to follow the Dorogovtsev-Mendes distribution (2001),
- *semantic networks* tend to follow the scale-free properties (Steyvers et al., 2005), etc.

Yes We Can

These properties do not depend on a language w.r.t. the parameters.



Source: Steyvers et al. (2005)

Clustering Algorithms

We will focus on four different clustering algorithms:

- Chinese Whispers (CW)
- Markov Clustering (MCL)
- MaxMax
- Watset

There are *a lot* of other clustering algorithms!

Chinese Whispers (CW)

- **Chinese Whispers** (CW) is a *randomized* hard clustering algorithm for both weighted and unweighted graphs (Biemann, 2006)
- Named after a famous children's game, it uses random shuffling to induce clusters
- Originally designed for such NLP tasks as word sense induction, language separation, etc.

Source: Pixabay (2015)

Chinese Whispers: Algorithm

Input: graph $G = (V, E), \mathrm{weight} : (G_u, i) \to \mathbb{R}, \forall u \in V, 1 \leq i \leq |V|$
Output: clustering C

```
label(V_i) ← i for all 1 ≤ i ≤ |V|                                    ▷ Initialization
while labels change do              ▷ labels(G_u) is a set of node labels in G_u
    for all u ∈ V in random order do
        label(u) ← arg max_{i∈labels(G_u)} weight(G_u, i)
                                            ▷ Pick the most weighted label in G_u
C ← {{u ∈ V : label(u) = i} : i ∈ labels(G)}
return C
```

Chinese Whispers: Label Weighting

Typical strategies to weigh the labels in the neighborhood G_u of u in G:

- Sum of the edge weights corresponding to the label i (`top`):

$$\mathrm{weight}(G_u, i) = \textstyle\sum_{\{u,v\} \in E_u : \mathrm{label}(v) = i} w(u, v)$$

- Use the node degree $\deg(v)$ to amortize highly-weighted edges (`nolog`):

$$\mathrm{weight}(G_u, i) = \textstyle\sum_{\{u,v\} \in E_u : \mathrm{label}(v) = i} \frac{w(u,v)}{\deg(v)}$$

- Use log-degree for amortization (`log`):

$$\mathrm{weight}(G_u, i) = \textstyle\sum_{\{u,v\} \in E_u : \mathrm{label}(v) = i} \frac{w(u,v)}{\log(1+\deg(v))}$$

Chinese Whispers: Example

- We consider an example on a graph from Biemann (2006, Figure 2)

Chinese Whispers: Discussion

Pros:

- Very simple and non-parametric
- Very fast, the running time is $O(|E|)$
- Works well for a lot of NLP tasks

Cons:

- Every run yields different results
- Node oscillation is possible
- No convergence guarantee

Implementations:

- https://github.com/uhh-lt/chinese-whispers
- https://github.com/nlpub/chinese-whispers-python

Markov Clustering (MCL)

- **Markov Clustering** (MCL) is a *stochastic* hard clustering algorithm that simulates *flows* in a graph using **random walks** (van Dongen, 2000)
- The algorithm makes a series of adjacency matrix transformations to obtain the partitioning: *expansion* and *inflation*
- MCL has been applied in a number of different domains, mostly in bioinformatics (Vlasblom et al., 2009)
- Similar to Affinity Propagation (Frey et al., 2007)

Source: Pixabay (2013)

Markov Clustering: Algorithm

Input: graph $G = (V, E)$, adjacency matrix A,
expansion parameter $e \in \mathbb{N}$, inflation parameter $r \in \mathbb{R}^+$

Output: clustering C

$A_{i,i} \leftarrow 1$ **for all** $1 \leq i \leq |V|$ ▷ Add self-loops

$A_{i,j} \leftarrow \frac{A_{i,j}}{\sum_{1 \leq k \leq |V|} A_{k,j}}$ **for all** $1 \leq i \leq |V|, 1 \leq j \leq |V|$ ▷ Normalize

while A changes **do**

  $A \leftarrow A^e$ ▷ Expand

  $A_{i,j} \leftarrow A_{i,j}^r$ **for all** $1 \leq i \leq |V|, 1 \leq j \leq |V|$ ▷ Inflate

  $A_{i,j} \leftarrow \frac{A_{i,j}}{\sum_{1 \leq k \leq |V|} A_{k,j}}$ **for all** $1 \leq i \leq |V|, 1 \leq j \leq |V|$ ▷ Normalize

$C \leftarrow \{\{V_j \in V : A_{i,j} \neq 0\} : 1 \leq i \leq |V|, 1 \leq j \leq |V|\}$

return C

Markov Clustering: Example

- We consider an example on a graph from Biemann (2006, Figure 2)

Markov Clustering: Discussion

Pros:

- ✚ Eventually, the algorithm converges (but there is no formal proof)
- ✚ Works well for a lot of NLP tasks

Cons:

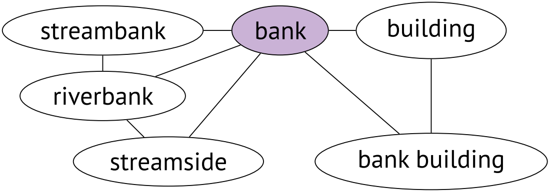
- − Relatively slow, the worst-case running time is $O(|V|^3)$
- − An efficient implementation requires sparse matrices

Implementations:

- `https://micans.org/mcl/`

This Clustering is Very Hard!

- OK, but how about the fact that the word “bank” is polysemeous?
- Hard clustering algorithms will treat this word incorrectly



Source: Pixabay (2015)

MaxMax

- **MaxMax** is a *soft* clustering algorithm designed for *weighted* graphs, such as co-occurrence graphs (Hope et al., 2013a)
- MaxMax transforms the input undirected weighted graph G into an unweighted directed graph G'
- Then, it extracts *quasi-strongly connected* subgraphs from G', which are overlapping clusters

Source: Pixabay (2016)

MaxMax: Algorithm

Input: graph $G = (V, E)$, weigthing function $w : E \to \mathbb{R}$
Output: clustering C

$E' \leftarrow \emptyset$
for all $\{u, v\} \in E$ **do**
  if $w(u, v) = \max_{v' \in V_u} w(u, v')$ **then**
    $E' \leftarrow E' \cup (v, u)$
$G' = (V, E')$
$\mathrm{root}(u) \leftarrow \mathrm{true}$ **for all** $u \in V$
for all $u \in V$ **do** ▷ Can be done using BFS
  if $\mathrm{root}(u)$ **then**
    for all $v \in \mathrm{succ}(u)$ **do** ▷ Successors of u in G'
      $\mathrm{root}(u) \leftarrow \mathrm{false}$
$C \leftarrow \{\{u\} \cup \mathrm{succ}(u) : u \in V, \mathrm{root}(u)\}$
return C

MaxMax: Example

- We consider an example from Hope et al. (2013a, Figure 3)

MaxMax: Discussion

Pros:

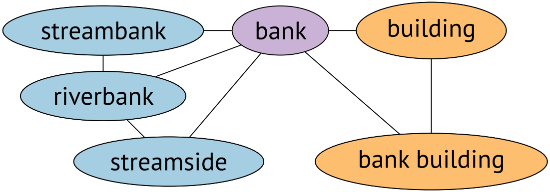
- \+ The algorithm is non-parametric
- \+ Very fast, the running time is $O(|E|)$, like CW
- \+ Works well for word sense induction (Hope et al., 2013b)

Cons:

- − Assumptions are not clear
- − Applicability seems to be limited (Ustalov et al., 2017)
- − No implementation offered by the authors

Graph-Based Word Sense Induction (WSI)

- Dorow et al. (2003) proposed a nice approach for **word sense induction** (WSI) using graphs
- Extract the *node neighborhood*, remove the node, and cluster the remaining graph
- Every cluster C^i corresponds to the *context* of the i-th sense of the node



Source: Pixabay (2016)

Watset

- **Watset** is not a clustering algorithm
- However, it is a *meta-algorithm* for turning *hard* clustering algorithms into *soft* clustering algorithms
- Watset **transforms** the input graph by replacing each node with one or more *senses* of this node (Ustalov et al., 2017)
- Under the hood Watset does word sense induction (Dorow et al., 2003) and context disambiguation (Faralli et al., 2016)

Source: Pixabay (2016)

Watset: Algorithm I

Input: graph $G = (V, E)$, algorithms $\mathrm{Cluster}_{\mathsf{Local}}$ and $\mathrm{Cluster}_{\mathsf{Global}}$, similarity measure $\mathrm{sim} : (\mathrm{ctx}(a), \mathrm{ctx}(b)) \to \mathbb{R}, \forall \mathrm{ctx}(a), \mathrm{ctx}(b) \subset V$

Output: clusters C

for all $u \in V$ **do** ▷ Local Step: Sense Induction

  $\mathrm{senses}(u) \leftarrow \emptyset$

  $V_u \leftarrow \{v \in V : \{u, v\} \in E\}$ ▷ Note that $u \notin V_u$

  $E_u \leftarrow \{\{v, w\} \in E : v, w \in V_u\}$

  $G_u \leftarrow (V_u, E_u)$

  $C_u \leftarrow \mathrm{Cluster}_{\mathsf{Local}}(G_u)$ ▷ Cluster the open neighborhood of u

  for all $C_u^i \in C_u$ **do**

    $\mathrm{ctx}(u^i) \leftarrow C_u^i$

    $\mathrm{senses}(u) \leftarrow \mathrm{senses}(u) \cup \{u^i\}$

$\mathcal{V} \leftarrow \bigcup_{u \in V} \mathrm{senses}(u)$ ▷ Global Step: Sense Graph Nodes

Watset: Algorithm II

for all $\hat{u} \in \mathcal{V}$ **do** $\triangleright$ Local Step: Context Disambiguation
  $\widehat{\mathrm{ctx}}(\hat{u}) \leftarrow \emptyset$
  for all $v \in \mathrm{ctx}(\hat{u})$ **do** $\triangleright$ $\hat{u} \in \mathcal{V}$ is a sense of $u \in V$
    $\hat{v} \leftarrow \arg\max_{v' \in \mathrm{senses}(v)} \mathrm{sim}(\mathrm{ctx}(\hat{u}) \cup \{u\}, \mathrm{ctx}(v'))$
    $\widehat{\mathrm{ctx}}(\hat{u}) \leftarrow \widehat{\mathrm{ctx}}(\hat{u}) \cup \{\hat{v}\}$
$\mathcal{E} \leftarrow \{\{\hat{u}, \hat{v}\} \in \mathcal{V}^2 : \hat{v} \in \widehat{\mathrm{ctx}}(\hat{u})\}$ $\triangleright$ Global Step: Sense Graph Edges
$\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$ $\triangleright$ Global Step: Sense Graph Construction
$\mathcal{C} \leftarrow \mathrm{Cluster}_{\mathsf{Global}}(\mathcal{G})$ $\triangleright$ Global Step: Sense Graph Clustering
$C \leftarrow \{\{u \in V : \hat{u} \in \mathcal{C}^i\} \subseteq V : \mathcal{C}^i \in \mathcal{C}\}$ $\triangleright$ Remove the sense labels
return C

Watset: Example

- We consider an example from Ustalov et al. (2018a)

Watset: Discussion

Pros:

- Conceptually, very simple
- Scales very well
- Shows very good results on very different tasks (Ustalov et al., 2017; Ustalov et al., 2018b)

Cons:

- Slow; computational complexity of disambiguation is $O(\Delta^4)$
- As good as the underlying clustering algorithms are good

Implementations:

- `https://github.com/dustalov/watset`
- `https://github.com/nlpub/watset-java`

  The Java implementation of Watset also contains CW, MCL, and MaxMax. **Feel free to play with them!**

Evaluation

- Clustering is an *unsupervised* task, so evaluation is not easy
  - For evaluating *hard* clustering algorithms, it is possible to use the evaluation techniques for flat clustering, see Manning et al. (2008, Chapter 16)
  - Evaluation of *soft* clustering is an even more challenging task, we will focus on *paired F-score* and *normalized modified purity*
- There are a lot of others, such as generalized conventional mutual information (Viamontes Esquivel et al., 2012), etc.
- Also, apparently, NLP researchers do not pay enough attention to statistical significance of their results (Dror et al., 2018)

Paired Precision, Recall, and F_1-score

- Every cluster C^i can be represented as a complete graph of $\frac{|C^i|(|C^i|-1)}{2}$ undirected edges (pairs) P^i
- A clustering C can be then compared to a gold clustering C_G using *paired F-score* between pair unions P and P_G (Manandhar et al., 2010):

$$\mathrm{TP} = |P \cup P_G|, \quad \mathrm{FP} = |P \setminus P_G|, \quad \mathrm{FN} = |P_G \setminus P|$$

$$\mathrm{Pr} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \quad \mathrm{Re} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \quad \mathrm{F}_1 = 2\frac{\mathrm{Pr} \times \mathrm{Re}}{\mathrm{Pr} + \mathrm{Re}}$$

- This is a very straightforward and interpretable approach, but it does not explicitly assess the quality of overlapping clusters

Normalized Modified Purity

- **Purity** is a measure of the extent to which clusters contain a single class (Manning et al., 2008), which is useful for evaluating *hard* clusterings:

$$\mathrm{PU} = \frac{1}{|C|} \sum_{i}^{|C|} \max_{j} |C^i \cap C_G^j|$$

- Kawahara et al. (2014) proposed *normalized modified purity* for *soft* clustering that considers weighted overlaps $\delta_{C^i}(C^i \cap C_G^j)$:

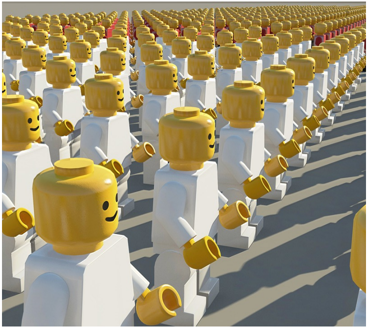
$$\mathrm{nmPU} = \frac{1}{|C|} \sum_{i \text{ s.t. } |C^i|>1}^{|C|} \max_{1 \leq j \leq |C_G|} \delta_{C^i}(C^i \cap C_G^j)$$

$$\mathrm{niPU} = \frac{1}{|C_G|} \sum_{j=1}^{|G|} \max_{1 \leq i \leq |C|} \delta_{C_G^j}(C^i \cap C_G^j)$$

$$\mathrm{F}_1 = 2\frac{\mathrm{nmPU} \times \mathrm{niPU}}{\mathrm{nmPU} + \mathrm{niPU}}$$

Statistical Significance

- It is not enough just to measure the clustering quality, it is necessary to evaluate the statistical significance!
- However, the use of statistical tests is not yet widespread in NLP experiments (Dror et al., 2018)
- Use computationally-intensive **randomization tests** for precision, recall and F-score (Yeh, 2000)
  - "No difference in means after *shuffling*"
- Consider the `sigf` toolkit (Padó, 2006) that implements these tests in Java



Source: Pixabay (2016)

Randomization Test for Average Values

Input: vectors $\vec{A}$ and $\vec{B}$, number of trials $N \in \mathbb{N}$
Output: two-tailed p-value

$\text{uncommon} \leftarrow \{1 \leq i \leq |\vec{A}| : A_i \neq B_i\}$
$s \leftarrow 0$
for all $1 \leq n \leq N$ **do**
    $\vec{A}' \leftarrow \vec{A}$ ▷ Copy $\vec{A}$
    $\vec{B}' \leftarrow \vec{B}$ ▷ Copy $\vec{B}$
    for all $i \in \text{uncommon}$ **do**
        if $\texttt{rand}(1) = 0$ **then** ▷ Flip a coin
            $A'_i, B'_i \leftarrow B_i, A_i$ ▷ Shuffle by swapping the values if tails
    if $|\text{mean}(\vec{A}') - \text{mean}(\vec{B}')| \geq |\text{mean}(\vec{A}) - \text{mean}(\vec{B})|$ **then**
        $\text{s} \leftarrow \text{s} + 1$ ▷ The test is two-tailed
return $\frac{s}{N}$ ▷ This value can be compared to a significance level

Randomization Test for Average Values: Example

Example from Padó (2006):

- $\vec{A} = (1, 2, 1, 2, 2, \mathbf{2}, 0), \quad \mathrm{mean}(\vec{A}) \approx 1.4286$
- $\vec{B} = (4, 5, 5, 4, 3, \mathbf{2}, 1), \quad \mathrm{mean}(\vec{B}) \approx 3.4286$
- $\mathrm{uncommon} = \{1, 2, 3, 4, 5, 7\}$
- $|\mathrm{mean}(\vec{A}) - \mathrm{mean}(\vec{B})| = 2$
- $N = 10^6$
- $p \approx 0.0313$
- Given the significance level of 0.05, the difference is significant

This technique can be generalized to F-score and others (Yeh, 2000).

Case Studies

We describe two case studies in our paper draft for COLI (Ustalov et al., 2018a):

- **Synset Induction** from Synonymy Dictionaries, from our ACL 2017 paper (Ustalov et al., 2017)
- Unsupervised Semantic **Frame Induction**, from our ACL 2018 paper (Ustalov et al., 2018b)

Source: Pixabay (2017)

Synset Induction

- Ontologies and thesauri are crucial to many NLP applications that require common sense reasoning
- The building blocks of WordNet (Fellbaum, 1998) are **synsets**, sets of mutual synonyms

  {*broadcast*, *program*, *programme*}
- Can we build synsets from scratch using just *synonymy dictionaries* like Wiktionary?

Source: Pixabay (2016)

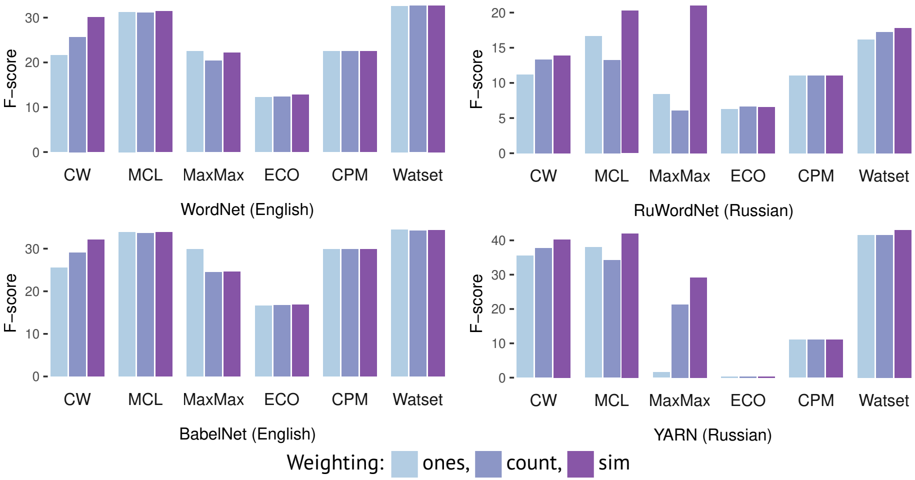
Synset Induction: Approach

1. Construct a weighted undirected graph using synonymy pairs from Wiktionary as edges
2. Weight them using cosine similarity between the corresponding word embeddings
3. Cluster this graph and treat the clusters as the synsets

Code and Data: `https://github.com/dustalov/watset`

Synset Induction: Results

- Watset showed the best results as according to paired F_1-score



Synset Induction: Example

Size	Synset
2	{*decimal point*, *dot*}
2	{*wall socket*, *power point*}
3	{*gullet*, *throat*, *food pipe*}
3	{*CAT*, *computed axial tomography*, *CT*}
4	{*microwave meal*, *ready meal*, *TV dinner*, *frozen dinner*}
4	{*mock strawberry*, *false strawberry*, *gurbir*, *Indian strawberry*}
5	{*objective case*, *accusative case*, *oblique case*, *object case*, *accusative*}
5	{*discipline*, *sphere*, *area*, *domain*, *sector*}
6	{*radio theater*, *dramatized audiobook*, *audio theater*, *radio play*, *radio drama*, *audio play*}
6	{*integrator*, *reconciler*, *consolidator*, *mediator*, *harmonizer*, *uniter*}
7	{*invite*, *motivate*, *entreat*, *ask for*, *incentify*, *ask out*, *encourage*}
7	{*curtail*, *craw*, *yield*, *riding crop*, *harvest*, *crop*, *hunting crop*}

Frame Induction

- A **semantic frame** is a collection of facts that specify features, attributes, and functions (Fillmore, 1982)

FrameNet	**Role**	**Lexical Units (LU)**
Perpetrator	Subject	kidnapper, alien, militant
FEE	Verb	snatch, kidnap, abduct
Victim	Object	son, people, soldier, child

- Used in question answering, textual entailment, event-based predictions of stock markets, etc.
- Can we build frames from scratch using just *subject-verb-object* (SVO) triples like DepCC (Panchenko et al., 2018)?

Source: Pixabay (2017)

Kidnapping

Definition:

The words in this frame describe situations in which a Perpetrator carries off and holds the Victim against his or her will by force.

Two men KIDNAPPED a Millwall soccer club employee, police said last night.

Not even the ABDUCTION of his children by Captain Hook and his scurvy sidekick, Smee, can shake Peter's scepticism.

FEs:

Core:

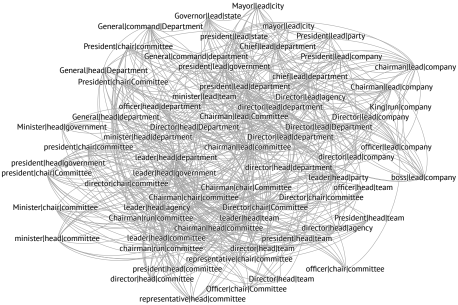
Perpetrator [Perp] **Semantic Type:** Sentient	The Perpetrator is the person (or other agent) who carries off and holds the Victim against his or her will.
Victim [Vict] **Semantic Type:** Sentient	The Victim is the person who is carried off and held against his/her will.

Lexical Units:

abduct.v, abducted.a, abduction.n, abductor.n, kidnap.v, kidnapped.a, kidnapper.n, kidnapping.n, nab.v, shanghai.v, snatch.v, snatcher.n

Source: https://framenet.icsi.berkeley.edu/fndrupal/luIndex

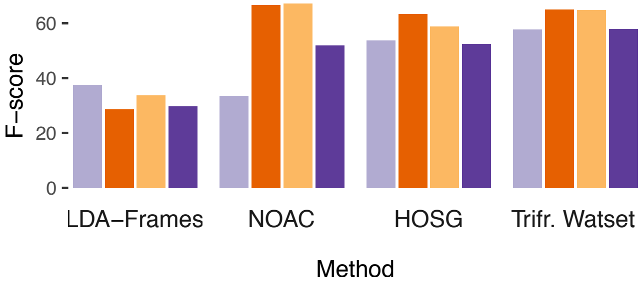
1. Use word embeddings to embed each triple $t = (s, v, o)$ in a low-dimensional *vector space* as $\vec{t} = \vec{s} \oplus \vec{v} \oplus \vec{o}$
2. Construct a weighted undirected graph using $k \in \mathbb{N}$ nearest neighbors of each triple vector
3. Cluster this graph and extract *triframes* by aggregating the corresponing roles



Code and Data: `https://github.com/uhh-lt/triframes`

Frame Induction: Results

- *Triframes* outperformed state-of-the-art frame induction approaches, including Higher-Order Skip-Gram (HOSG) and LDA-Frames, on the FrameNet corpus (Baker et al., 1998) as according to F_1 (nmPU)



Element: verb, subject, object, frame

Frame Induction: Examples I

Subjects: Company, firm, company
Verbs: buy, supply, discharge, purchase, expect
Objects: book, supply, house, land, share, company, grain, which, item, product, ticket, work, this, equipment, House, it, film, water, something, she, what, service, plant, time

Subjects: student, scientist, we, pupil, member, company, man, nobody, you, they, US, group, it, people, Man, user, he
Verbs: do, test, perform, execute, conduct
Objects: experiment, test

Subjects: people, we, they, you
Verbs: feel, seek, look, search
Objects: housing, inspiration, gold, witness, partner, accommodation, Partner

Frame Induction: Examples II

Subjects: you, she, he, return, they, we, themselves, road, help, who
Verbs: govern, discourage, resemble, encumber, urge, pummel, *...912 more verbs...*, swarm, anticipate, spew, derail, emit, snap
Objects: you, pass, she, he, it, product, change, solution, total, any, wall, they, something, people, classic, this, interest, itself, flat, place, part, controversy

Subjects: Word, glue, pill, speed, drug, pot, they, those, mine, item, resource, this, its, it, something, most, horse, material, chemical, plant, information, word
Verbs: use, attach, apply, follow
Objects: we, they, you, it, report, he

Subjects: he
Verbs: phone, book
Objects: you

Which Algorithm to Choose?

? Is your graph relatively small and you need *hard* clustering?

! Markov Clustering

? Is your graph big and you still need *hard* clustering?

! Chinese Whispers

? Do you need *soft* clustering?

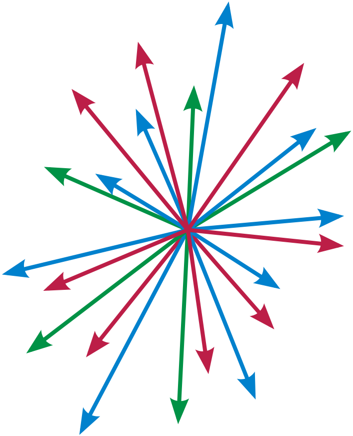
! Watset

…but My Objects are Just Vectors!

It is possible to represent the objects in a vector space as a graph (von Luxburg, 2007):

- use the k nearest neighbors,
- use all the neighbors within the ε-radius,
- use a fully-connected *weighted* graph

Think of a graph as a *discretized* vector space.



Source: Wikipedia (2007)

Resources

Events:

- **TextGraphs**, a Workshop on Graph-Based Algorithms for NLP, `http://www.textgraphs.org/`

Books:

- Graph-Based NLP & IR (Mihalcea et al., 2011)
- Structure Discovery in Natural Language (Biemann, 2012)

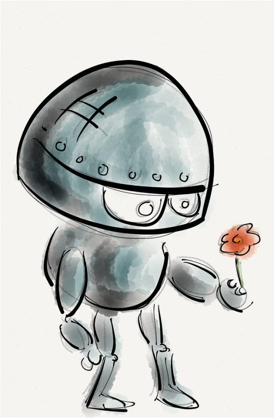
Datasets:

- Stanford Network Analysis Project, `https://snap.stanford.edu/data/`
- Leipzig Corpora Collection (Goldhahn et al., 2012)
- Wiktionary (Zesch et al., 2008; Krizhanovsky et al., 2013)

NLPub, `https://nlpub.ru/` (in Russian)

Conclusion

- A graph is a meaningful representation; clustering captures its implicit structure as exhibited by data
- The algorithms are well-developed and ready to use as soon as a graph is constructed
- Not covered here:
  - spectral graph theory, see a great tutorial by von Luxburg (2007)
  - community detection algorithms from network science, see Fortunato (2010)
- A few promising research directions:
  - graph convolutional networks (Marcheggiani et al., 2017),
  - graph embeddings (Goyal et al., 2018)



Source: Pixabay (2016)

Questions?

Contacts

Dr. **Dmitry Ustalov**,
Data and Web Science Group,
University of Mannheim

- `https://dws.informatik.uni-mannheim.de/en/people/researchers/dr-dmitry-ustalov/`
- dmitry@informatik.uni-mannheim.de

Revision: `17533ee`

References I

Azadani, M. N., Ghadiri, N., and Davoodijam, E. (2018). Graph-based biomedical text summarization: An itemset mining and sentence clustering approach. In: *Journal of Biomedical Informatics* 84, pp. 42–58. doi: 10.1016/j.jbi.2018.06.005.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet Project. In: *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*. ACL '98. Montreal, QC, Canada: Association for Computational Linguistics, pp. 86–90. doi: 10.3115/980845.980860.

Biemann, C. (2006). Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems. In: *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*. TextGraphs-1. New York, NY, USA: Association for Computational Linguistics, pp. 73–80. url: http://dl.acm.org/citation.cfm?id=1654774.

Biemann, C. (2012). Structure Discovery in Natural Language. Theory and Applications of Natural Language Processing. Springer Berlin Heidelberg. doi: 10.1007/978-3-642-25923-4.

Buluç, A. et al. (2009). Parallel Sparse Matrix-vector and Matrix-transpose-vector Multiplication Using Compressed Sparse Blocks. In: *Proceedings of the Twenty-first Annual Symposium on Parallelism in Algorithms and Architectures*. SPAA '09. Calgary, AB, Canada: ACM, pp. 233–244. doi: 10.1145/1583991.1584053.

van Dongen, S. (2000). Graph Clustering by Flow Simulation. PhD thesis. Utrecht, The Netherlands: University of Utrecht.

Dorogovtsev, S. N. and Mendes, J. F. F. (2001). Language as an evolving word web. In: *Proceedings of the Royal Society of London B: Biological Sciences* 268.1485, pp. 2603–2606. doi: 10.1098/rspb.2001.1824.

Dorow, B. and Widdows, D. (2003). Discovering Corpus-Specific Word Senses. In: *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 2*. EACL '03. Budapest, Hungary: Association for Computational Linguistics, pp. 79–82. doi: 10.3115/1067737.1067753.

Dror, R. et al. (2018). The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2018. Melbourne, VIC, Australia: Association for Computational Linguistics, pp. 1383–1392.

Duff, I. S., Grimes, R. G., and Lewis, J. G. (1989). Sparse Matrix Test Problems. In: *ACM Transactions on Mathematical Software* 15.1, pp. 1–14. doi: 10.1145/62038.62043.

Faralli, S. et al. (2016). Linked Disambiguated Distributional Semantic Networks. In: *The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part II*. Cham, Switzerland: Springer International Publishing, pp. 56–64. doi: 10.1007/978-3-319-46547-0_7.

Fellbaum, C. (1998). WordNet: An Electronic Database. MIT Press.

Fillmore, C. J. (1982). Frame Semantics. In: *Linguistics in the Morning Calm*. Seoul, South Korea: Hanshin Publishing Co., pp. 111–137.

Fortunato, S. (2010). Community detection in graphs. In: *Physics Reports* 486.3, pp. 75–174. doi: 10.1016/j.physrep.2009.11.002.

References II

Frey, B. J. and Dueck, D. (2007). Clustering by Passing Messages Between Data Points. In: *Science* 315.5814, pp. 972–976. doi: 10.1126/science.1136800.

Goldhahn, D., Eckart, T., and Quasthoff, U. (2012). Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages. In: *Proceedings of the Eight International Conference on Language Resources and Evaluation*. LREC 2012. Istanbul, Turkey: European Language Resources Association (ELRA), pp. 759–765.

Goyal, P. and Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. In: *Knowledge-Based Systems* 151, pp. 78–94. doi: 10.1016/j.knosys.2018.03.022.

Hope, D. and Keller, B. (2013a). MaxMax: A Graph-Based Soft Clustering Algorithm Applied to Word Sense Induction. In: *Computational Linguistics and Intelligent Text Processing: 14th International Conference, CICLing 2013, Samos, Greece, March 24-30, 2013, Proceedings, Part I*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 368–381. doi: 10.1007/978-3-642-37247-6_30.

Hope, D. and Keller, B. (2013b). UoS: A Graph-Based System for Graded Word Sense Induction. In: *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. Atlanta, GA, USA: Association for Computational Linguistics, pp. 689–694. url: https://aclweb.org/anthology/S13-2113.

Kawahara, D., Peterson, D. W., and Palmer, M. (2014). A Step-wise Usage-based Method for Inducing Polysemy-aware Verb Classes. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics Volume 1: Long Papers*. ACL 2014. Baltimore, MD, USA: Association for Computational Linguistics, pp. 1030–1040. url: https://aclweb.org/anthology/P14-1097.

Krizhanovsky, A. A. and Smirnov, A. V. (2013). An approach to automated construction of a general-purpose lexical ontology based on Wiktionary. In: *Journal of Computer and Systems Sciences International* 52.2, pp. 215–225. doi: 10.1134/S1064230713020068.

Lewis, M. and Steedman, M. (2013). Unsupervised Induction of Cross-Lingual Semantic Relations. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2013. Seattle, WA, USA: Association for Computational Linguistics, pp. 681–692.

von Luxburg, U. (2007). A tutorial on spectral clustering. In: *Statistics and Computing* 17.4, pp. 395–416. doi: 10.1007/s11222-007-9033-z.

Lyzinski, V., Sell, G., and Jansen, A. (2015). An Evaluation of Graph Clustering Methods for Unsupervised Term Discovery. In: *INTERSPEECH-2015*. Dresden, Germany: International Speech Communication Association, pp. 3209–3213. url: https://www.isca-speech.org/archive/interspeech_2015/papers/i15_3209.pdf.

Manandhar, S. et al. (2010). SemEval-2010 Task 14: Word Sense Induction & Disambiguation. In: *Proceedings of the 5th International Workshop on Semantic Evaluation*. SemEval 2010. Uppsala, Sweden: Association for Computational Linguistics, pp. 63–68. url: https://aclweb.org/anthology/S10-1011.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.

References III

Marcheggiani, D. and Titov, I. (2017). Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2017. Copenhagen, Denmark: Association for Computational Linguistics, pp. 1506–1515. url: https://aclweb.org/anthology/D17-1159.

Mihalcea, R. and Radev, D. (2011). Graph-Based Natural Language Processing and Information Retrieval. Cambridge University Press.

Navigli, R. and Ponzetto, S. P. (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. In: *Artificial Intelligence* 193, pp. 217–250. doi: 10.1016/j.artint.2012.07.001.

Padó, S. (2006). User's guide to sigf: Significance testing by approximate randomisation. url: https://nlpado.de/~sebastian/software/sigf.shtml.

Page, L. et al. (1999). The PageRank Citation Ranking: Bringing Order to the Web. Tech. rep. 1999-66. Stanford InfoLab. url: http://ilpubs.stanford.edu:8090/422/.

Panchenko, A. et al. (2018). Building a Web-Scale Dependency-Parsed Corpus from Common Crawl. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*. LREC 2018. Miyazaki, Japan: European Language Resources Association (ELRA), pp. 1816–1823.

Pelevina, M. et al. (2016). Making Sense of Word Embeddings. In: *Proceedings of the 1st Workshop on Representation Learning for NLP*. RepL4NLP. Berlin, Germany: Association for Computational Linguistics, pp. 174–183. url: https://aclweb.org/anthology/W16-1620.

Steyvers, M. and Tenenbaum, J. B. (2005). The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth. In: *Cognitive Science* 29.1, pp. 41–78. doi: 10.1207/s15516709cog2901_3.

Tauer, G. et al. (2019). An incremental graph-partitioning algorithm for entity resolution. In: *Information Fusion* 46, pp. 171–183. doi: 10.1016/j.inffus.2018.06.001.

Ustalov, D., Panchenko, A., and Biemann, C. (2017). Watset: Automatic Induction of Synsets from a Graph of Synonyms. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2017. Vancouver, BC, Canada: Association for Computational Linguistics, pp. 1579–1590. doi: 10.18653/v1/P17-1145.

Ustalov, D. et al. (2018a). Local-Global Graph Clustering with Applications in Sense and Frame Induction. Submitted to Computational Linguistics. arXiv: 1808.06696 [cs.CL].

Ustalov, D. et al. (2018b). Unsupervised Semantic Frame Induction using Triclustering. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. ACL 2018. Melbourne, VIC, Australia: Association for Computational Linguistics, pp. 55–62. url: https://aclweb.org/anthology/P18-2010.

Viamontes Esquivel, A. and Rosvall, M. (2012). Comparing network covers using mutual information. arXiv: 1202.0425 [math-ph].

References IV

Vlasblom, J. and Wodak, S. J. (2009). Markov clustering versus affinity propagation for the partitioning of protein interaction graphs. In: *BMC Bioinformatics* 10.1, p. 99. doi: 10.1186/1471-2105-10-99.

Yeh, A. (2000). More accurate tests for the statistical significance of result differences. In: *Proceedings of the 18th Conference on Computational Linguistics - Volume 2*. COLING '00. Saarbrücken, Germany: Association for Computational Linguistics, pp. 947–953. doi: 10.3115/992730.992783.

Zesch, T., Müller, C., and Gurevych, I. (2008). Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In: *Proceedings of the 6th International Conference on Language Resources and Evaluation*. LREC 2008. Marrakech, Morocco: European Language Resources Association (ELRA), pp. 1646–1652. url: http://www.lrec-conf.org/proceedings/lrec2008/pdf/420_paper.pdf.