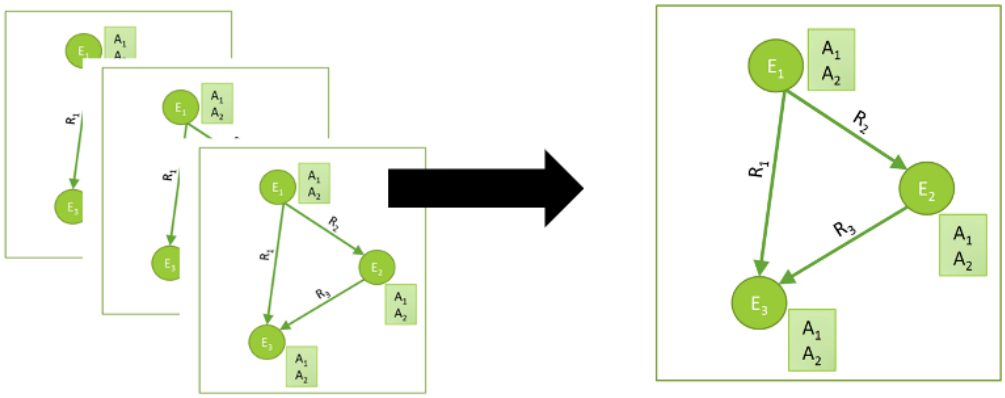
3.3 KNOWLEDGE INFERENCE

Knowledge Inference

From available knowledge to more complete and precise knowledge



Once knowledge graphs have been extracted from text, they can be further processed. This enables the inference of new knowledge from the existing knowledge, but as well the correction, completion and integration of existing knowledge bases.

Basic Questions in Knowledge Inference

Who are the entities?

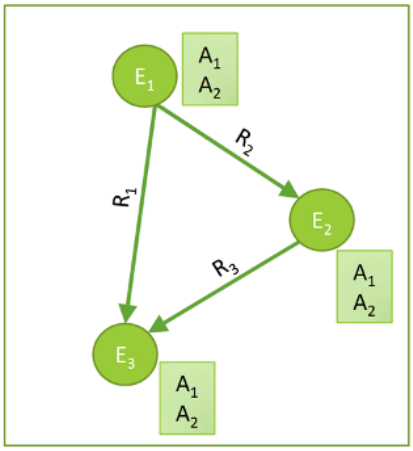
- ➢ Entity Linking / Disambiguation
- ➢ Data integration

What are their attributes?

- ➢ Collective Classification

How are they related?

- ➢ Link prediction



Knowledge inference concerns a wide number of problems that have been studied in different contexts. Some of the basic examples are:

- Entity linking and disambiguation, which concerns the problem of identifying which entity names represent the same real-world entity, respective which entity is referred to in case of ambiguous entity names.
- Schema integration, which concerns the problem which classes, attributes and relationships in one knowledge base correspond to which in another one.
- Collective classification, which concerns the problem of learning unknown attribute values from the available knowledge in a knowledge base.
- Link prediction, which concerns the problem of learning unknown relationships from the available knowledge in a knowledge base.

3.3.1 Entity Disambiguation

Task: Link a text mention in a document to an entry in a knowledge base (e.g., WikiPedia or WikiData)

- Also called entity resolution and linking

Example: "Schindler is a Swiss industrial company. One of its main competitors is the American producer, Otis."

Once entities have been recognized in a text, one can link them to their corresponding counter-parts in a knowledge base. So-called ntitiy disambiguation is a step that usually follows named entity recognition.

Challenge

Two problems

- Homonyms: entities with the same name
- Synonyms: different names for the same entity

Schindler's List

From Wikipedia, the free encyclopedia

This article is about the film. For the book that inspired this film (published in the U.S. as Schindler's List), see Schindler's Ark.

Schindler's List is a 1993 American epic historical period drama film directed and co-produced by Steven Spielberg and written by Steven Zaillian. It is based on the novel *Schindler's Ark* by Australian novelist Thomas Keneally. The film follows Oskar Schindler, a Sudeten German businessman, who saved

Schindler's List

Otis, Colorado

From Wikipedia, the free encyclopedia

Coordinates: 40°9′2″N 102°57′45″W

Otis is a Statutory Town in Washington County, Colorado, United States. The population was 534 at the 2000 census.

Contents [hide]

1 History
2 Geography
3 Demographics
4 Climate
5 See also
6 References

Town of Otis, Colorado

Town

Entering Otis from the east.

Entity disambiguation can however be quite challenging due to homonymy and synonymy. Handling these problems is essential for every text analytics tasks. Not being able to handle homonymy usually results in the introduction of noise into the results (poor precision), whereas not properly handling synonymy risks to miss relevant documents (poor recall).

Sources of Information

Local information: textual similarity of a mention of the entity and the entry in the knowledge base

Example: "Schindler" ≈ "Schindler's list"

"Schindler" ≈ "Schindler Group"

Global information: coherence of different text mentions of potential entities within a document with respect to a knowledge base

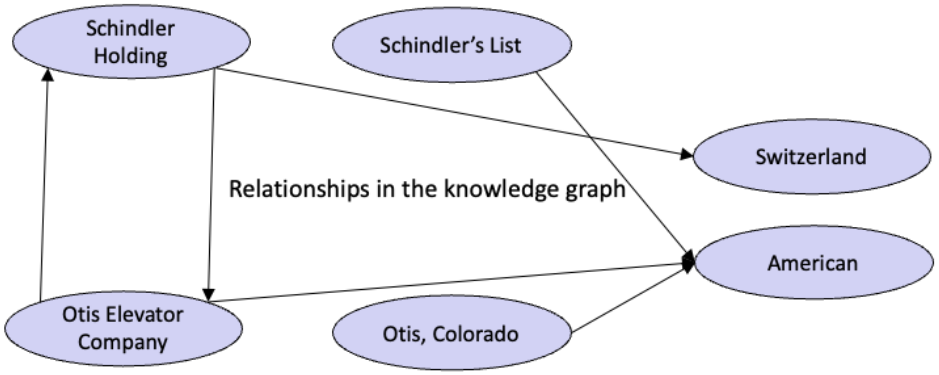
→ entity graph

For performing entity disambiguation one can exploit two different sources of information.

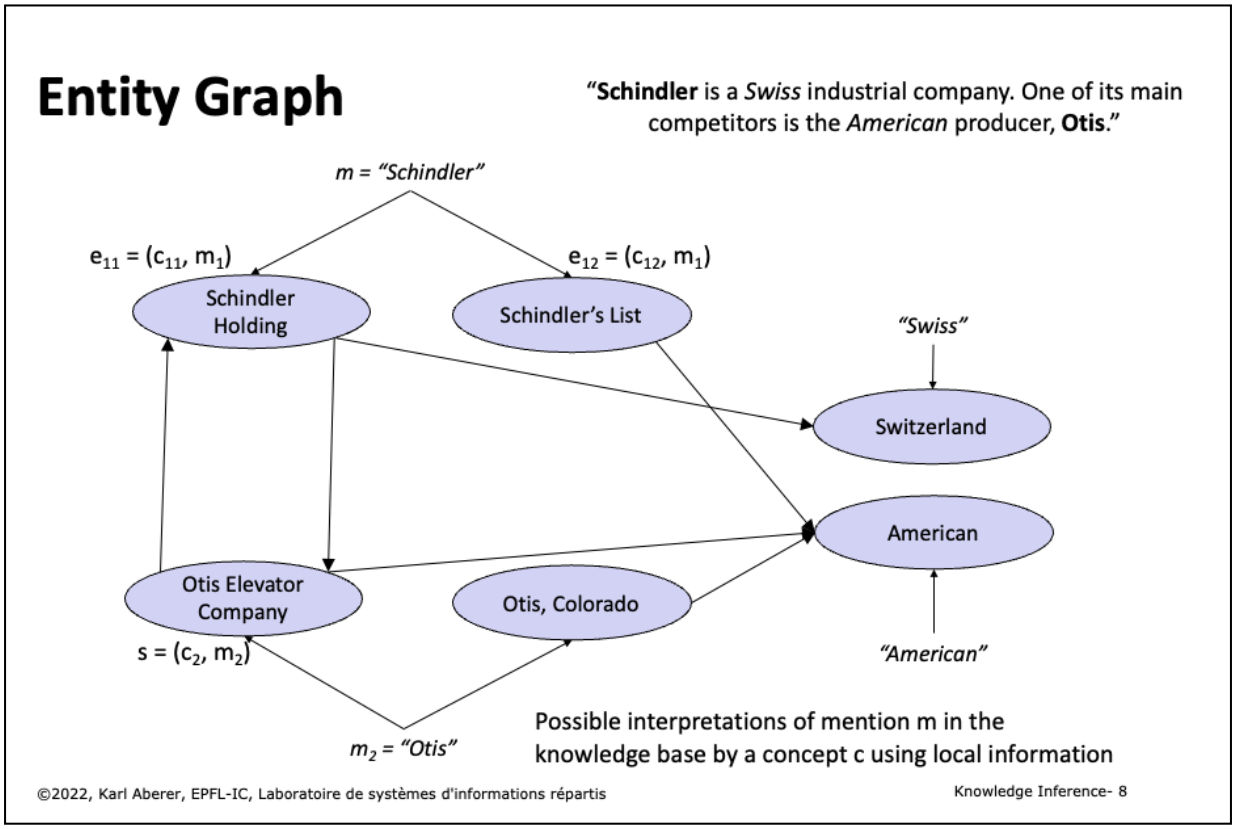
1. Local information extracted from the text mention, or its vicinity. This can be used to compare the text mention and its features with the text entry in the knowledge base, in order to obtain evidence which entities in the knowledge base are potential matches.

2. Coherence of different text mentions and knowledge base entries. When multiple entities are extracted from text, they will have relationships among each other. By analyzing whether the relationships among entries in the text and in the knowledge base are consistent with which each other one can disambiguate mentions of entities in the text.

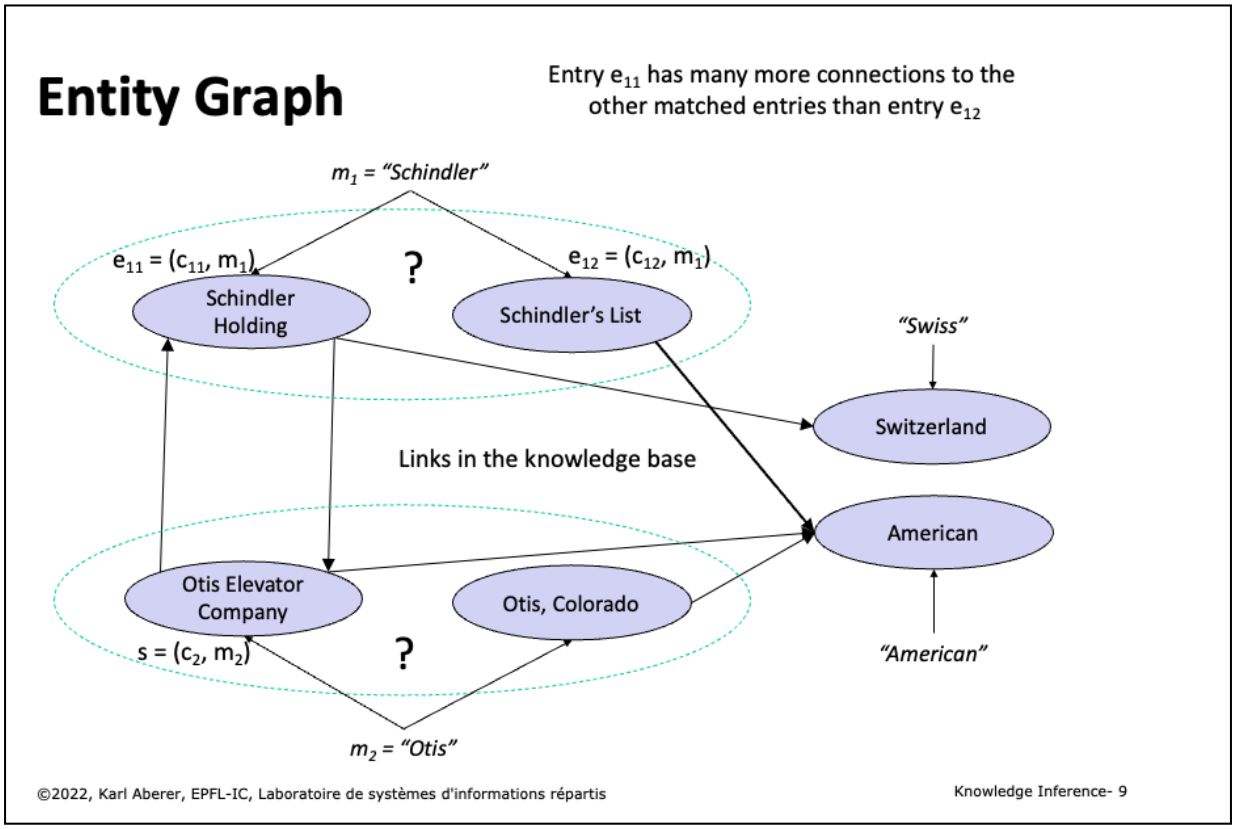
Example Knowledge Graph



We will give now an example to illustrate the process of entity disambiguation using a knowledge graph. Assume the knowledge base contains the subgraph shown in the figure, with entities that apparently are relevant to our text example.



In a first step the textual mentions of the entities are linked to entries in the knowledge graph. After performing NER, this can be done using local information, textual similarity between the text mention and the name of the concept in the knowledge graph. By linking the text mentions we create entity matches of the form e = (c, m), where e is a node in the entity graph, c is the concept from the knowledge graph, and m is the textual mention of an entity in the text. The entity graph consists of all matched nodes in the knowledge graph, and the relationships among them. We may also associate a similarity measure to each node in the entity graph, capturing how well the text mention matches the concept in the knowledge graph.

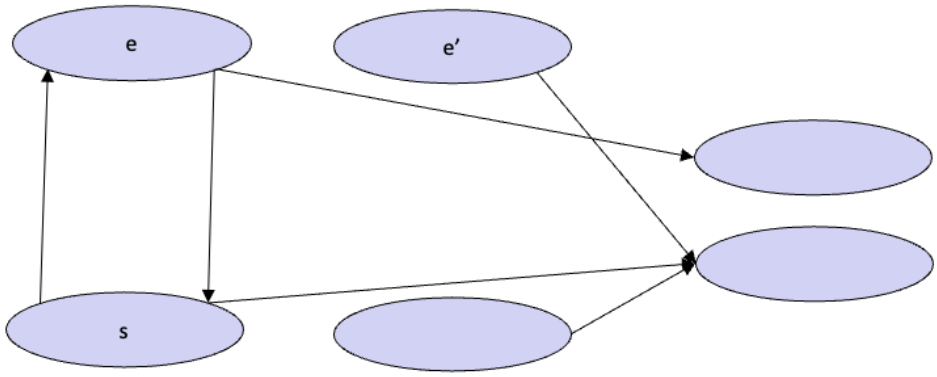


The example shows that different interpretations are possible for both the mention “Schindler” and “Otis”. Therefore, the problem is to decide which if each these two alternative matches is the better one. A possibility indication for the quality of a match is the connectivity of the nodes with the other nodes in the entity graph.

Coherence

How well does the node s
support the choice of node e?

Other formulation: How relevant is node e for node s?
(as compared to e', an alternative node)



Abstracting from the details of the example before, we can highlight the problem we need to solve. Given two alternative interpretations of a text mention, e and e', and another node s in the entity graph, how well does the node s support the two alternatives. In the example graph we see that intuitively node s is much better connected with node e which indicates that e might indeed be the better choice.

The problem described bears similarity with another problem that has been addressed in the context of personalized Web search. Imagine the nodes are Web pages, s is a page that has been bookmarked by a user, and the question is which other pages, like e and e', are also of interest to the user. From the example, it appears evident that it is page e. For making this intuition on connectedness operational, a variant of the PageRank algorithm has been proposed, called Personalized PageRank. The same algorithm has also been used to solve entity disambiguation.

Personalized PageRank

Same as PageRank, except that random jumps are always back to the same node (or same set of nodes)

- Original motivation: use personal bookmark list as source of rank
- Entity disambiguation: node s is considered as source of rank

$$\overrightarrow{p_s} = c(qR \cdot \overrightarrow{p_s} + (1-q)\overrightarrow{e_s})$$

$\overrightarrow{e_s} = (0,0,\dots,1,\dots,0)$, 1 at entry s

Personalized PageRank works almost the same as the original PageRank algorithm. The difference is that random jumps are not performed uniformly at random to nodes, but to a selected subset of nodes. In the context of personalized Web search this subset would be the personal bookmark list. By jumping back to the nodes from that list, it will have a large influence on the ranking of a page, such that nodes that are well connected to the bookmarks will receive a larger ranking value. In the context of entity disambiguation, the source is a selected node in the entity graph, and personalized page rank is used to compute how well other nodes in the entity graph are supported by that node.

Computing PPR

Standard iterative computation

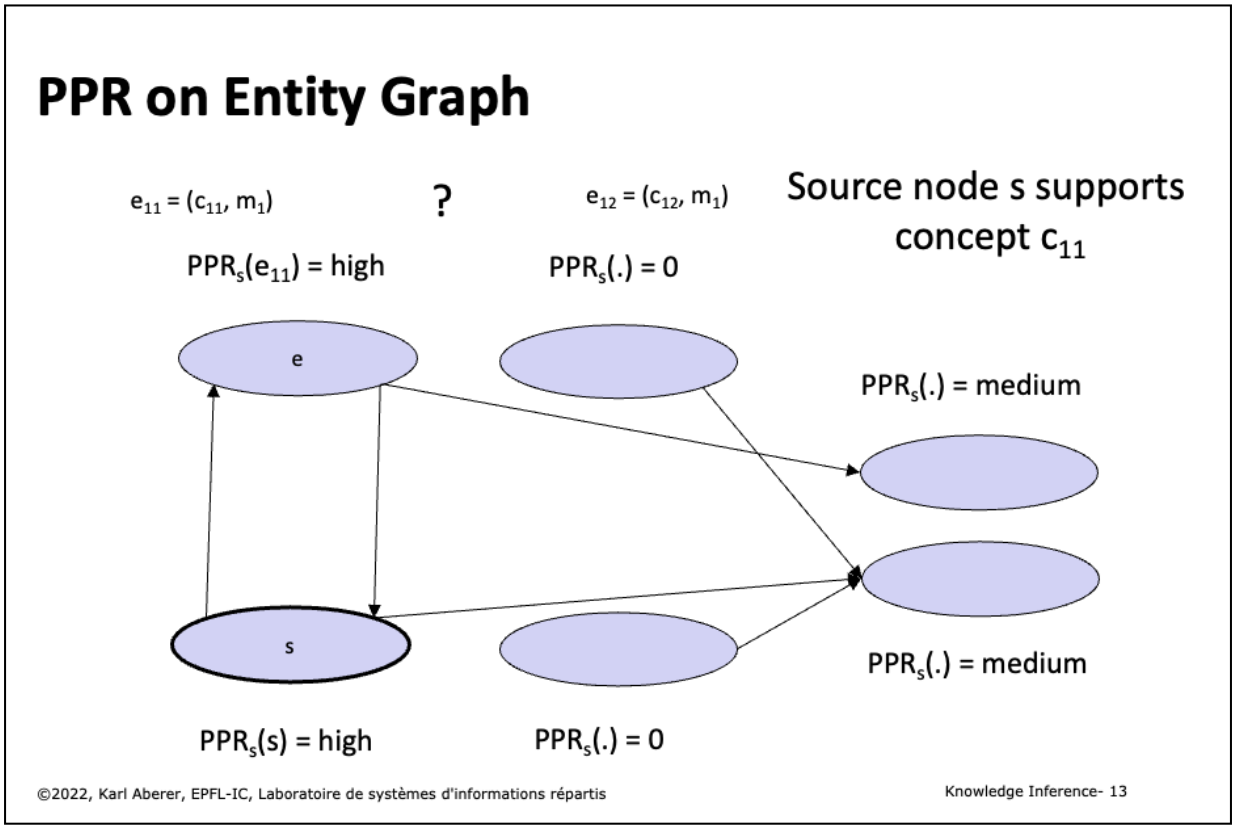
$$\vec{p}_{s,0} := \overrightarrow{e_s}$$

$$\vec{p}_{s,i+1} := c\left(qR \cdot \overrightarrow{p_{s,i}} + (1-q)\overrightarrow{e_s}\right)$$

Monte Carlo method

- Perform multiple independent random walks starting at *s*
- Compute distribution of end points of random walks

PPR can be computed either iteratively, like standard PageRank, or using a Monte Carlo method, by starting random walks independently at s and aggregating the distribution of the end points of those walks.



Applying PPR to the entity group, by considering a **source node s** as the source of rank, will generate a distribution of ranking values for all other nodes. Nodes that are well connected to s will receive higher ranking values. Intuitively it is clear, that in our example node e will be receive higher ranking when starting from s, and thus is the preferred interpretation for the entity matching mention m_1.

Contributing Nodes

Only one interpretation c for a mention m is valid

- Competing nodes $e' = (c', m)$ that have the same entity mention as $e = (c, m)$ cannot support e
- For multiple nodes s that have the same entity mention m', only the one with highest contribution is considered

Thus

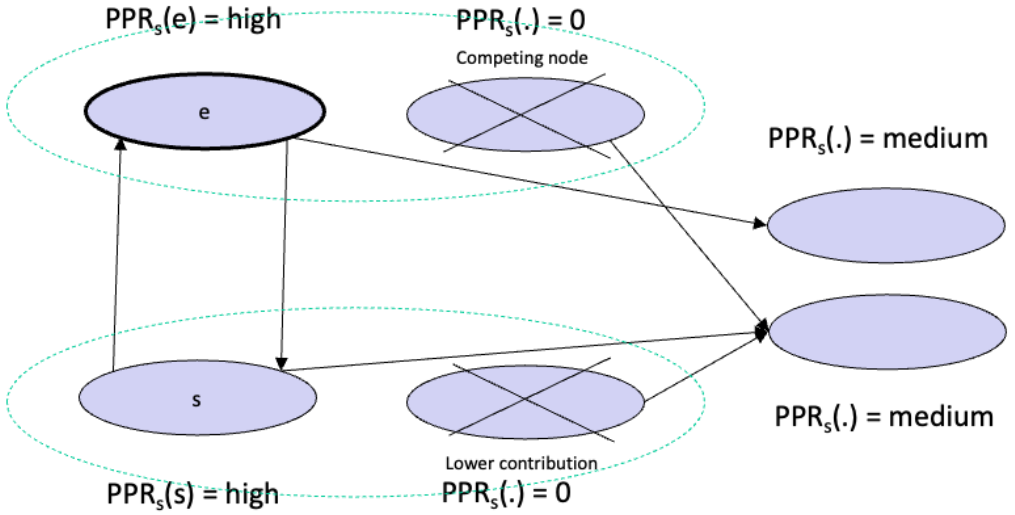
$$Contributors_e = \{(m', \underset{c}{\text{argmax}}\, PPR_{(c, m')}(e), m' \neq m)\}$$

The question is which nodes s should contribute to the disambiguation of a mention m. The considerations for choosing those nodes take into account the following two issues:

- When we are computing the support for an interpretation $e = (c, m)$ of text mention m, with competing interpretations $e' = (c', m)$, the competing node should not be used to produce support for e. So, these nodes are excluded.
- When there exist multiple nodes for a text mention m' , only the one that is producing the largest contribution to the interpretation $e = (c, m)$ of text mention m is considered. This makes sense, since the other nodes related to m' might favor an alternative interpretation for m, and therefore should not be considered.

This results in a set of contributing nodes $Contributors_e$ for each node $e = (c, m)$ in the entity graph.

Example: Contributors$_e$



This figure shows that for the computation of the score of node e using node s as source, two nodes from the entity graph will be excluded. First, the node that is linked to the same text mention as e and is a competing node, and second the node that is linked to the same text mention and s and is producing a lower score.

Scoring

Finding the concept candidate linked to a mention m that is most likely to be valid

1. For a concept candidates c compute total support received from contributing nodes s

$$e = (c, m), s = (c', m')$$

$$score(e) = \sum_{s \in Contributors_e} PPR_s(e)$$

2. Select the candidate with highest score

Using the contributing nodes, the personalized PageRank scores that they contribute are added and the candidate with the best score is selected.

Considering Popularity

The method can furthermore consider popularity measures for nodes, e.g., it's degree

- If information is insufficient, favor popular nodes

$$score(e) = \sum_{s \in Contributors_e} PPR_s(e)pop(s) + PPR_{avg}pop(e)$$

Promotes contributions from popular nodes

Promotes popular nodes

To further improve the method, it is possible to add a general popularity measure as a weight the contributions of source nodes. This will favor the contribution of popular nodes, which is beneficial if little information is available for disambiguation. In such a case, it is better to choose a popular candidate since chances that an interpretation supported by a popular node are higher. One of possible choice of a popularity score could be the number of links a node has in the knowledge base. Similarly, also the popularity of the candidates e can be used as a contribution to the score.

Some Results

Without popularity

Other methods

Uses pageRank

With popularity

Models	Cucerzan	Kulkarni	Hoffart	Shirakawa	Alhelbawy	iSim	PPR	PPRSim
Micro	51.03	72.87	81.82	82.29	87.59	62.61	85.56	91.77

Experimental results show that the method works relatively well, with around 90% of entities that are correctly disambiguated. One can observe that the use of popularity helps to slightly improve the results.

Which is false?

A. Entity disambiguation addresses the problem of synonyms
B. Named entity recognition addresses the problem of synonyms
C. Entity disambiguation addresses the problem of entity classification
D. Named entity recognition addresses the problem of entity classification

Which nodes cannot contribute to the score of a mention linked to a concept?

A. Other concepts linked to the same mention
B. Concepts that have in the knowledge graph no outgoing links
C. Concepts that have in the knowledge graph no incoming links
D. Concepts with low popularity

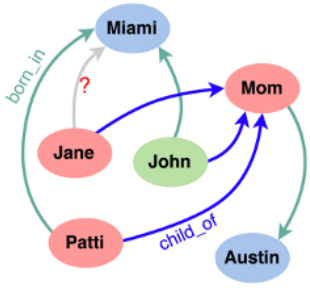
3.3.3 Link Prediction

Large knowledge bases are usually incomplete

- DBPedia: 60% of persons miss place of birth
- FreeBase: 71% of persons miss place of birth etc.

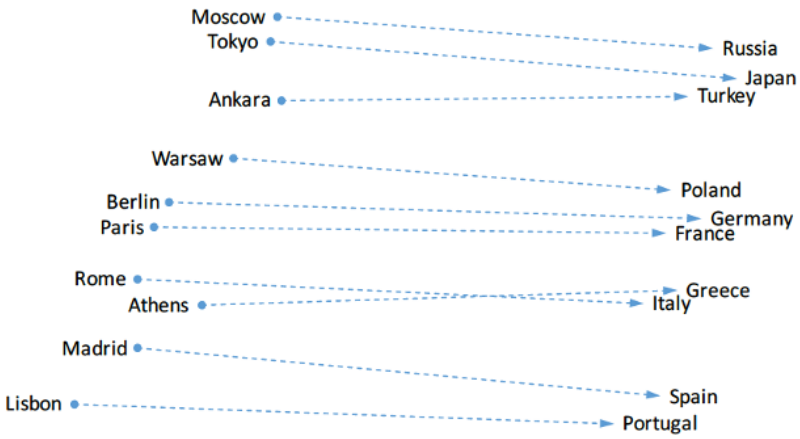
Try to predict missing links from existing data

Jane born in Miami?



In general, knowledge bases are incomplete. Thus, there is a significant interest in completing the relationships among entities. To do so one might exploit “patterns” that entities and relationships follow and generalize them.

Observation on Word Embeddings



In order to tackle the problem of link prediction, we come back to an observation that we had made earlier for word embeddings. We have seen that relationships seem to be represented as linear transformations.

Relations in Word Embeddings

$$
\begin{aligned}
\boldsymbol{v}_{Japan} - \boldsymbol{v}_{Tokyo} &\approx \boldsymbol{v}_{Germany} - \boldsymbol{v}_{Berlin} \\
\boldsymbol{v}_{Germany} - \boldsymbol{v}_{Berlin} &\approx \boldsymbol{v}_{Italy} - \boldsymbol{v}_{Rome} \\
\boldsymbol{v}_{Italy} - \boldsymbol{v}_{Rome} &\approx \boldsymbol{v}_{Portugal} - \boldsymbol{v}_{Lisbon}
\end{aligned}
$$

Idea: Find a vector $v_{is\_capital\_of}$ such that

$$
\begin{aligned}
\boldsymbol{v}_{Tokyo} + \boldsymbol{v}_{is\_capital\_of} - \boldsymbol{v}_{Japan} &\approx \mathbf{0} \\
\boldsymbol{v}_{Berlin} + \boldsymbol{v}_{is\_capital\_of} - \boldsymbol{v}_{Germany} &\approx \mathbf{0} \\
\boldsymbol{v}_{Rome} + \boldsymbol{v}_{is\_capital\_of} - \boldsymbol{v}_{Italy} &\approx \mathbf{0} \\
\boldsymbol{v}_{Lisbon} + \boldsymbol{v}_{is\_capital\_of} - \boldsymbol{v}_{Portugal} &\approx \mathbf{0}
\end{aligned}
$$

Relations are also represented as embedding vectors!

We can express the linear relationship among the embeddings of two types of entities, e.g., countries and their capitals, by stating that the differences of the embedding vectors are similar. Since the differences of the vectors are similar, we can also introduce a vector for this difference, which we call $v_{is\_capital\_of}$. This vector can be considered as a representation of the relationship. This formulation of the problem is the starting point for methods for identifying new relationships in knowledge graphs using embedding techniques.

Model

Knowledge graph G consists of (correct) triples (h, r, t) where $h, t \in E$ and $r \in R$

Each entity and relationship is mapped to a low-dimensional vector, resulting in $\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t$

To formulate the model, assume that we have a knowledge graph consisting of triples (h, r, t). h indicates the term "head" and t the term "tail", and both are entities. The objective is to find low-dimensional vectors representing both the head and tail entities, and the relationships.

Plausibility Score

Define a (im)plausibility score $\mathrm{f}(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t)$ such that

$$\mathrm{f}(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t) < \mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_{r'}, \boldsymbol{v}_{t'})$$

if (h, r, t) is a plausible triple and (h', r, t') is an implausible triple for relation r

$(h', r, t') \in G'(h, r, t)$ is a set of incorrect triples, generated by corrupting the correct triple (h, r, t)

For learning the model, we need to introduce a loss function. To define this loss function, we first introduce a plausibility score $\mathrm{f}(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t)$ for triples of embedding vectors. This score can be thought of as the inverse of the probability a triple triple to represent a correct fact. That means that more plausible the fact is the lower the plausibility score, approaching zero.

The property that we require the score to satisfy is that correct triples have a higher score than incorrect triples. This requires examples of both correct and incorrect triples. For correct triples we can use facts from a given knowledge graph. For obtaining incorrect triples we can take a correct fact and corrupt it by modifying parts of the fact with random replacements.

Learning the Model

Minimize the margin loss function

$$J(\theta) = \sum_{\substack{(h,r,t)\epsilon G \\ (h',r,t')\epsilon G'(h,r,t)}} \max(0, \gamma + \mathrm{f}(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t) - \mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'}))$$

$\gamma > 0$ is a hyperparameter

- The loss function attempts to separate scores from correct and incorrect triples by the margin γ

Example: if $\gamma = 2$ and $\mathrm{f}(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t) = 0$, then
if $\mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'}) = 2$ or 3, then max = 0
if $\mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'}) = 1$, then max = 1 ☞ minimize $J(\theta)$

Using the plausibility function, we can formulate a loss function that should be minimized. The form of the function is a margin loss function. It tries to separate the positive from the negative samples, using the plausibility score. The hyperparameter γ determines by which margin the optimization will try to separate the plausibility scores of correct vs. incorrect triples.

The optimization is performed as usual with SGD.

TransE Model

One of the first embedding-based models for knowledge base completion

- Based on the intuition from text WE

$$f(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t) = \|\boldsymbol{v}_h + \boldsymbol{v}_r - \boldsymbol{v}_t\|_2$$

Each entity and relationship is mapped to a low-dimensional vector, resulting in $\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t$

The generic model described so far can be instantiated using different choices for the plausibility score f. In an early version the plausibility score was computed following the inspiration from the initial observation made for relationships in word embeddings. It introduces a vector $\boldsymbol{v}_r$ to represent the relationship and compares it to the difference of head and tail entity embedding vectors.

Performing SGD

Initialize vectors with random values

- From interval $[-\frac{c}{\sqrt{k}}, \frac{c}{\sqrt{k}}]$ where k is the embedding dimension
- In each iteration
  - Sample a correct triple or batch
  - Derive a corrupt triple from the correct one: replace h or t by a random entity
  - Update embeddings by minimizing loss function
  - Normalize all entity vectors to 1 (not relationship vectors!)

The SGD algorithm proceeds as follows. First the vectors are initialized with random values (the choice is motivated by empirical findings from neural network training. In the published paper c = 6). Then in every iteration a triple (or several triples are randomly chosen). Negative samples are generated by randomly replacing head or tail (not both). The update of the embeddings is performed as usual by computing the differential of the loss function. Entity vectors are normalized to 1 in every iteration (this avoids the model to find a trivial solution).

Qualitative Results

INPUT (HEAD AND LABEL)	PREDICTED TAILS
J. K. Rowling `influenced by`	*G. K. Chesterton*, J. R. R. Tolkien, *C. S. Lewis*, **Lloyd Alexander**, Terry Pratchett, Roald Dahl, Jorge Luis Borges, *Stephen King*, Ian Fleming
Anthony LaPaglia `performed in`	*Lantana*, *Summer of Sam*, *Happy Feet*, *The House of Mirth*, Unfaithful, **Legend of the Guardians**, Naked Lunch, X-Men, The Namesake
Camden County `adjoins`	**Burlington County**, *Atlantic County*, *Gloucester County*, Union County, Essex County, New Jersey, Passaic County, Ocean County, Bucks County
The 40-Year-Old Virgin `nominated for`	*MTV Movie Award for Best Comedic Performance*, *BFCA Critics' Choice Award for Best Comedy*, *MTV Movie Award for Best On-Screen Duo*, MTV Movie Award for Best Breakthrough Performance, **MTV Movie Award for Best Movie**, MTV Movie Award for Best Kiss, D. F. Zanuck Producer of the Year Award in Theatrical Motion Pictures, Screen Actors Guild Award for Best Actor - Motion Picture
Costa Rica football team `has position`	*Forward*, *Defender*, *Midfielder*, **Goalkeepers**, Pitchers, Infielder, Outfielder, Center, Defenseman
Lil Wayne `born in`	**New Orleans**, Atlanta, Austin, St. Louis, Toronto, New York City, Wellington, Dallas, Puerto Rico
WALL-E `has the genre`	Animations, Computer Animation, *Comedy film*, *Adventure film*, *Science Fiction*, **Fantasy**, Stop motion, *Satire*, Drama

The qualitative results show that the method works reasonably well. The bold phrases are the correct predections. However, given that the knowledge base used for evaluation is incomplete, it is possible that also other predictions are meaningful, like the ones highlighted in italic.

Alternative Models

Model	Score function $f(h, r, t)$	Opt.
Unstructured	$\|\boldsymbol{v}_h - \boldsymbol{v}_t\|_{\ell_{1/2}}$	SGD
SE	$\|\mathbf{W}_{r,1}\boldsymbol{v}_h - \mathbf{W}_{r,2}\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_{r,1}, \mathbf{W}_{r,2} \in \mathbb{R}^{k\times k}$	SGD
SME	$(\mathbf{W}_{1,1}\boldsymbol{v}_h + \mathbf{W}_{1,2}\boldsymbol{v}_r + \mathbf{b}_1)^\top(\mathbf{W}_{2,1}\boldsymbol{v}_t + \mathbf{W}_{2,2}\boldsymbol{v}_r + \mathbf{b}_2)$ $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^n$; $\mathbf{W}_{1,1}, \mathbf{W}_{1,2}, \mathbf{W}_{2,1}, \mathbf{W}_{2,2} \in \mathbb{R}^{n\times k}$	SGD
TransE	$\|\boldsymbol{v}_h + \boldsymbol{v}_r - \boldsymbol{v}_t\|_{\ell_{1/2}}$; $\boldsymbol{v}_r \in \mathbb{R}^k$	SGD
TransH	$\|(\mathbf{I} - \boldsymbol{r}_p\boldsymbol{r}_p^\top)\boldsymbol{v}_h + \boldsymbol{v}_r - (\mathbf{I} - \boldsymbol{r}_p\boldsymbol{r}_p^\top)\boldsymbol{v}_t\|_{\ell_{1/2}}$ $\boldsymbol{r}_p, \boldsymbol{v}_r \in \mathbb{R}^k$; $\mathbf{I}$: Identity matrix size $k \times k$	SGD
TransR	$\|\mathbf{W}_r\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_r\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_r \in \mathbb{R}^{n\times k}$; $\boldsymbol{v}_r \in \mathbb{R}^n$	SGD
TransD	$\|(\mathbf{I} + \boldsymbol{r}_p\boldsymbol{h}_p^\top)\boldsymbol{v}_h + \boldsymbol{v}_r - (\mathbf{I} + \boldsymbol{r}_p\boldsymbol{t}_p^\top)\boldsymbol{v}_t\|_{\ell_{1/2}}$ $\boldsymbol{r}_p, \boldsymbol{v}_r \in \mathbb{R}^n$; $\boldsymbol{h}_p, \boldsymbol{t}_p \in \mathbb{R}^k$; $\mathbf{I}$: Identity matrix size $n \times k$	AdaDelta
lppTransD	$\|(\mathbf{I} + \boldsymbol{r}_{p,1}\boldsymbol{h}_p^\top)\boldsymbol{v}_h + \boldsymbol{v}_r - (\mathbf{I} + \boldsymbol{r}_{p,2}\boldsymbol{t}_p^\top)\boldsymbol{v}_t\|_{\ell_{1/2}}$ $\boldsymbol{r}_{p,1}, \boldsymbol{r}_{p,2}, \boldsymbol{v}_r \in \mathbb{R}^n$; $\boldsymbol{h}_p, \boldsymbol{t}_p \in \mathbb{R}^k$; $\mathbf{I}$: Identity matrix size $n \times k$	SGD
STransE	$\|\mathbf{W}_{r,1}\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_{r,2}\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_{r,1}, \mathbf{W}_{r,2} \in \mathbb{R}^{k\times k}$; $\boldsymbol{v}_r \in \mathbb{R}^k$	SGD
TranSparse	$\|\mathbf{W}_r^h(\theta_r^h)\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_r^t(\theta_r^t)\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_r^h, \mathbf{W}_r^t \in \mathbb{R}^{m\times k}$; $\theta_r^h, \theta_r^t \in \mathbb{R}$; $\boldsymbol{v}_r \in \mathbb{R}^n$	SGD
DISTMULT	$\boldsymbol{v}_h^\top\mathbf{W}_r\boldsymbol{v}_t$; $\mathbf{W}_r$ is a diagonal matrix $\in \mathbb{R}^{k\times k}$	AdaGrad
NTN	$\boldsymbol{v}_r^\top tanh(\boldsymbol{v}_h^\top\mathbf{M}_r\boldsymbol{v}_t + \mathbf{W}_{r,1}\boldsymbol{v}_h + \mathbf{W}_{r,2}\boldsymbol{v}_t + \mathbf{b}_r)$ $\boldsymbol{v}_r, \mathbf{b}_r \in \mathbb{R}^n$; $\mathbf{M}_r \in \mathbb{R}^{k\times k\times m}$; $\mathbf{W}_{r,1}, \mathbf{W}_{r,2} \in \mathbb{R}^{a\times k}$	L-BFGS
HolE	$sigmoid(\boldsymbol{v}_r^\top(\boldsymbol{v}_h \circ \boldsymbol{v}_t))$; $\boldsymbol{v}_r \in \mathbb{R}^k$, $\circ$ denotes *circular correlation*	AdaGrad
Bilinear-COMP	$\boldsymbol{v}_h^\top\mathbf{W}_{r_1}\mathbf{W}_{r_2}...\mathbf{W}_{r_m}\boldsymbol{v}_t$; $\mathbf{W}_{r_1}, \mathbf{W}_{r_2}, ..., \mathbf{W}_{r_m} \in \mathbb{R}^{k\times k}$	AdaGrad
TransE-COMP	$\|\boldsymbol{v}_h + \boldsymbol{v}_{r_1} + \boldsymbol{v}_{r_2} + ... + \boldsymbol{v}_{r_m} - \boldsymbol{v}_t\|_{\ell_{1/2}}$; $\boldsymbol{v}_{r_1}, \boldsymbol{v}_{r_2}, ..., \boldsymbol{v}_{r_m} \in \mathbb{R}^k$	AdaGrad
ConvE	$\boldsymbol{v}_t^\top g\left(\text{vec}\left(g\left(\text{concat}(\overline{v}_h, \overline{v}_r) * \Omega\right)\right)\boldsymbol{W}\right)$; g denotes a non-linear function	Adam
ConvKB	$\mathbf{w}^\top \text{concat}\left(g\left([\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t] * \Omega\right)\right)$; $*$ denotes a convolution operator	Adam

The TransE method is only one example of numerous methods that have been in the meanwhile proposed to tackle the link prediction problem. In the meantime, also transformer models are used for this task.

Which is true? The score function f(h,r,t) …

A. has always larger values for triples (h,r,t) that are part of the known knowledge graph than for other triples
B. maps triples to vectors in the embedding space
C. is always positive
D. is optimized by stochastic gradient descent

Question

If $f(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t) = 0.1$ and γ is increased from 2 to 3, optimizing the loss function

A. is primarily achieved by decreasing the values of $f(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t)$
B. is primarily achieved by increasing the values of $f(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t)$
C. is primarily achieved by decreasing the values of $f(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'})$
D. is primarily achieved by increasing the values of $f(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'})$

References

Course material based on

- Pershina, Maria, Yifan He, and Ralph Grishman. "Personalized page rank for named entity disambiguation." *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2015.
- Talukdar, Partha Pratim, and Koby Crammer. "New regularized algorithms for transductive learning." *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, Berlin, Heidelberg, 2009.
- Bordes, Antoine, et al. "Translating embeddings for modeling multi-relational data." Advances in neural information processing systems. 2013.
- Nguyen, Dat Quoc. "An overview of embedding models of entities and relationships for knowledge base completion." arXiv preprint arXiv:1703.08098 (2017).
- Doan, AnHai, et al. "Learning to map between ontologies on the semantic web." *Proceedings of the 11th international conference on World Wide Web*. ACM, 2002.

Appendix

The following material is kept for information only

3.3.2 Label Propagation

Inferring Attribute Values

Example: Which users on Twitter have positive or negative emotion towards a topic?

- Users are nodes in a graph (follower network)
- Emotion is an attribute of the node

Potential source of information in the case of Twitter

- Emoticons in tweets: indicate stance of user towards the topic
- Only a (small) fraction of the users is using emoticons

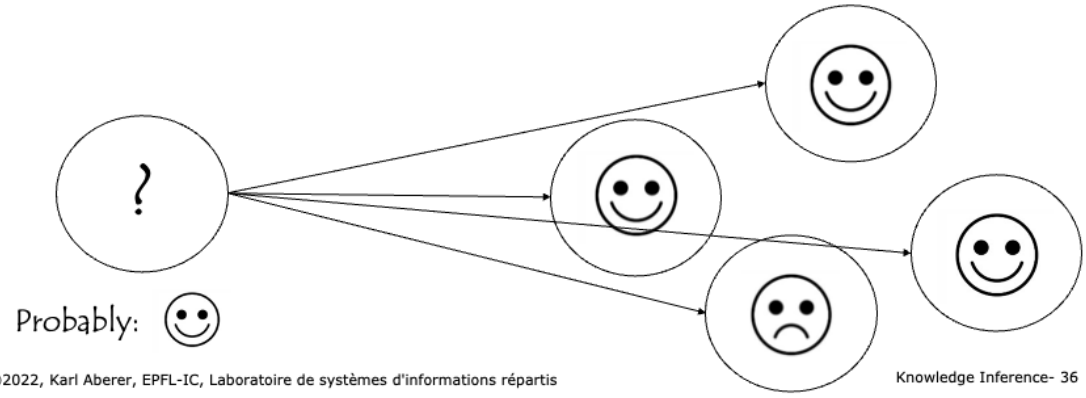
A second inference task in knowledge bases, after entities have been disambiguated, is to assign to the entities correct attribute values. For discrete attributes this problem can be understood as a classification problem. This question has, for example, been studied for classifying users in a social network with respect to their stance or emotion towards a specific topic.

In the case of emotion analysis there exists typically indications of emotions, e.g., in the form of the use of emoticons or specific hashtags. However, only few users are using those.

Propagating Attribute Values

Assumption: nodes that are connected by an edge, have a higher propensity of sharing the attribute of interest

- Twitter users following each other, are more likely to share the same emotion towards a topic

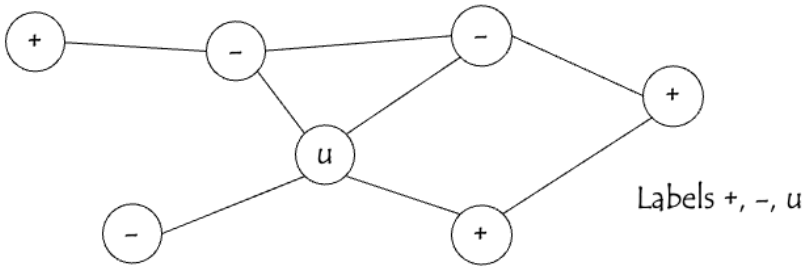


In many cases nodes that are connected by edges in a knowledge graph or as well in a social network, share properties. In social networks this is quite apparent. People that are connected through social links (e..g follower, friend, retweet, reply etc) are in general more likely to share opinions than those that are not. Is it possible to exploit this property to predict the attributes (respectively class lables) for those users that have none?

Model

Graph $G = (V, E)$ with vertices V and edges E

- Label set L of size n
- Vertices V have a label from a set $L \cup \{unknown\}$
- Edges are undirected and unweighted



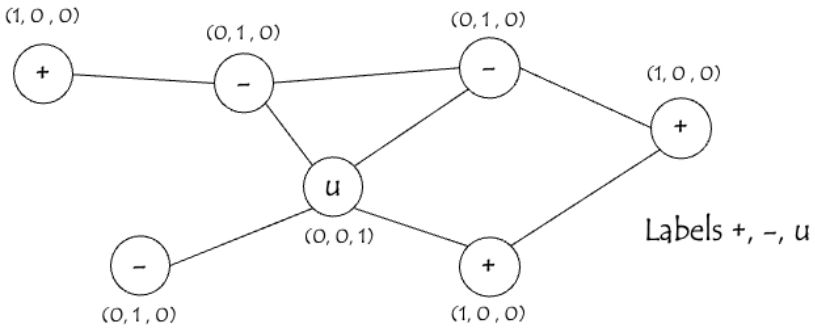
We consider the following model. A graph with vertices that can have either a label from a set L, or a label unknown. The edges are all undirected and unweighted. The model and approach can be extended for directed graphs with weighted edges.

We associate with vertices probability distributions that represent our knowledge about the assignment of a label. For all vertices we will compute an inferred probability distribution.

Optimization Objective

Objective

- Determine for a vertex v a label vector $\boldsymbol{l}_{inferred}(v)$ of size $n + 1$
- $\boldsymbol{l}_{inferred}(v)$ assigns a label probability



Label Inference

We assume that all neighbors excert the same influence on a node

Thus we would require that

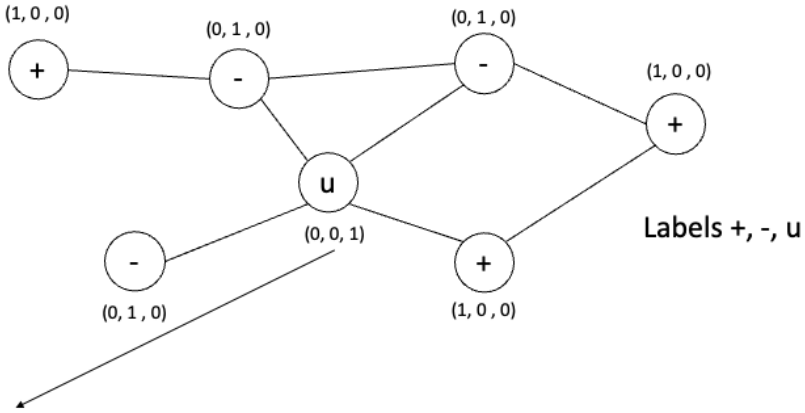
$$\boldsymbol{l}_{inferred}(v) = \frac{1}{\deg(v)} \sum_{(v,w)\in E} \boldsymbol{l}_{inferred}(w)$$

Recursive equation resp. random walk model (like PageRank)

In this model we assume that all neighbors of a node in the graph are equally influencing it. Thus, we can compute its expected label distribution by averaging the distributions of the neighbors. The resulting equation is analogous to the formulation of the PageRank model. Thus the Label Inference can be interpreted as a random walk model.

The model can be extended to weighted graphs, in which case the edge weight would be considered in the aggregation of the distributions of the neighboring vertices.

Example



$l_{inferred}$=1/4 ((0, 1 , 0) + (0, 1 , 0) + (0, 1 , 0) + (1, 0 , 0)) = (1/4, ¾, 0)

Injecting Pre-existing Knowledge

Initial knowledge on labels

Known labels

- $\boldsymbol{l}_{apriori}(v)$ is a vector of size $n+1$
- assigns weight 1 for label if known for $v \in V$

Unknown labels

- $\boldsymbol{l}_{unknown}$ is a vector of size $n+1$
- assigns weight 1 for label $unknown$

For some vertices we have an apriori assignment of labels (these are the vertices for which the label is known). For vertices that have no apriori label assigned we have a vector to represent the unknown state. Note that the apriori distribution can also be a true probability distribution, if we are initially not sure about the label, but have some partial knowledge.

Label Propagation Algorithm

$\boldsymbol{l}_{inferred}(v) := \boldsymbol{l}_{apriori}(v)$ for nodes with known labels,
otherwise $\boldsymbol{l}_{inferred}(v) := \boldsymbol{l}_{unknown}$

`while not converged`

$$\boldsymbol{l}_{inferred}(v) := \frac{1}{\deg(v)} \sum_{(v,w)\in E} \boldsymbol{l}_{inferred}(w)$$

$$\boldsymbol{l}_{inferred}(v) := \begin{array}{ll} p_v^{inj}\ \boldsymbol{l}_{apriori}(v) + & \text{// inject apriori knowledge} \\ p_v^{con}\ \boldsymbol{l}_{inferred}(v) + & \text{// infer from neighbors} \\ p_v^{aba}\ \boldsymbol{l}_{unknown} & \text{// abandon} \end{array}$$

The probabilities p_v^{inj}, p_v^{con} and p_v^{aba} can be interpreted as decisions in a random walk

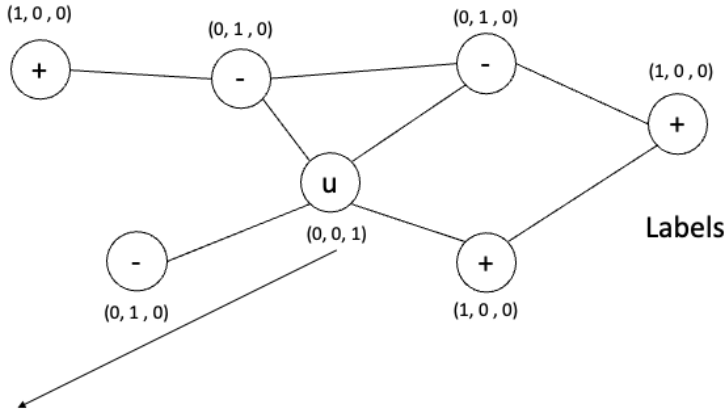
The full label propagation model adds additional aspects to the propagation of labels to neighbors. The probabilities should be understood as the decisions that a random walker can take to find a neighboring node from which to learn the label.

- For vertices with apriori labels, at every step the apriori distribution is injected with a certain probability p_v^{inj} that depends on the vertex
- For all vertices the propagation process (random walk) can also be abandoned with a certain probability p_v^{aba}. In that case the becomes unknown.
- The propagation of the label distribution to neighbors occurs then with a (remaining) probability of p_v^{con}.

As in PageRank the process is iterated till convergence occurs.

Example

Assume $p_v^{inj} = 1/2, p_v^{con} = 1/4, p_v^{aba} = 1/4$



$I_{inferred}$ = ½ (0, 0, 1) + ¼ (¼, ¾, 0) + ¼ (0, 0, 1) = (1/16, 3/16, 3/4)

Determining the Probabilities

The choice of the transition probabilities results in different variants of label propagation algorithms

Possible considerations

- pre-labelled nodes should have higher influence on neighbors than initially unlabeled nodes
- well-connected nodes should have higher influence that sparsely connected nodes

Example Model

$$c_v = \frac{\log 2}{\log(2+\deg(v))}$$

$$d_v = (1 - c_v)\sqrt{H(v)},\ H(v) = -\log\frac{1}{\deg(v)} \text{ if } v \text{ is labelled,}$$

$$d_v = 0 \text{ otherwise}$$

$$z_v = \max(c_v + d_v, 1)$$

$$p_v^{con} = \frac{c_v}{z_v}, \quad p_v^{inj} = \frac{d_v}{z_v} \text{ for labelled nodes, 0 otherwise}$$

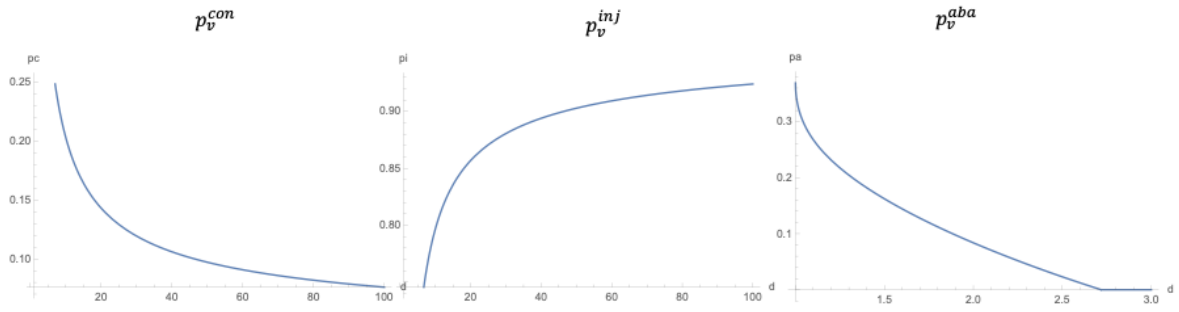
$$p_v^{aba} = 1 - p_v^{con} - p_v^{inj}$$

The transition probabilities depend on the properties of the vertices. In our model, the only relevant property is its degree.

In a more general model with edge weights the probabilities would depend on the distribution of edge weights of the edges connected to the vertice (Fan-out entropy heuristics)

Behavior of Probabilities: Labelled Nodes

- Injection probability increases with the degree of the nodes, while continuation probability decreases
- Abandoning probability positive for only very low degree nodes (d = 1,2)

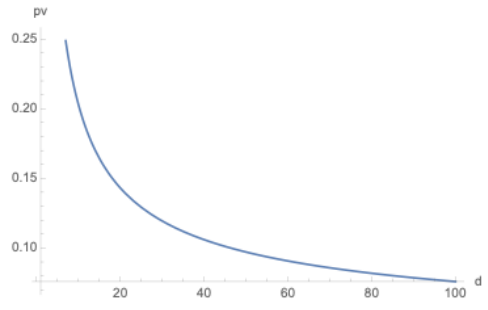


For labelled nodes the injection probability increases with the degree. Thus, well connected pre-labelled nodes have a lot of influence.

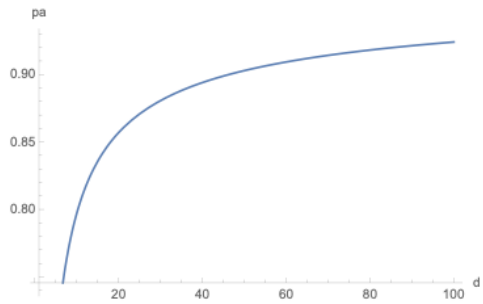
Behavior of Probabilities: Unlabelled Nodes

- Abandon probability increases with degree
- Prevents algorithm from propagating information through unlabeled, high-degree nodes

p_v^{con}



p_v^{aba}



For unlabeled nodes the behavior the abandon probability increases with the degree. Thus, high degree nodes have less influence.

Extensions

Label Propagation can be extended to

- A priori knowledge given as probability distribution
- Graphs with weighted edges
- Directed Graphs

Alternative algorithm: MAD (modified adsorption)

- Formulates an optimization problem and solves it directly
- Slightly better performance in practice

Discussion

Label propagation is an example of a **semi-supervised learning** algorithm

- Exploit partial labelling
- Useful in cases where labels are sparse or labels can produced only for special cases using heuristics or background knowledge
- Require that relationships among entities and their labels are correlated by some underlying principle

Different neighbors of a node v

A. Have a different influence depending on their degree
B. Have exactly same influence
C. Have a different influence depending on the degree of v
D. Have a different influence depending on whether they have a known label

The probabilities p_v^{inj}, p_v^{con} and p_v^{aba} depend on

A. The node degree
B. The pre-existing knowledge on labels
C. On both node degree and pre-existing knowledge
D. On further factors