2019 Final - MCQ

1. Which one of the following about the Expectation-Maximization algorithm is FALSE?
    a) In E step the labels change, in M step the weights of the workers change.
    b) The label with the highest probability is assigned as the new label.
    c) Assigning equal weights to workers initially decreases the convergence time. (CORRECT)
    d) It distinguishes experts from normal workers.

2. Which of the following is TRUE?
    a) Ontologies are used to directly map two schemas in order to overcome semantic heterogeneity.
    b) Graph representation of an RDF statement facilitates exchange and storage.
    c) RDF is a standardized model for encoding ontologies (CORRECT)
    d) XML does not facilitate introducing new terms which are domain specific.

3. Regarding features engineering, which of the following is wrong:
    a. Supervised discretization can merge any two intervals of the same variable. (CORRECT)
    b. Classifiers can be sensitive to the absolute scale of the variables.
    c. Features *filtering* consider single variables, whereas *wrapping* considers features combinations.
    d. Standardisation can produce arbitrarily large values whereas scaling does not.

4. Given the graph 1→2, 1→ 3, 2→3, 3→2, switching from Page Rank to Teleporting PageRank will have an influence on the value(s) of
    a. All the nodes (CORRECT)
    b. Node 1
    c. Node 2 and 3
    d. No nodes. The values will stay unchanged.

5. Which of the following is true:
    a. Modularity is a measure of how communities are connected together
    b. Agglomerative algorithms recursively decompose communities into sub-communities
    c. Divisive algorithms are based on modularity
    d. Girvan-Newman works by removing edges with the highest betweenness measure (CORRECT)

6. Which of the following is true:
    a. The tf-idf weight is the ratio between tf and idf
    b. The idf term decrease the impact of stop-words (CORRECT)
    c. Frequent terms obtain low tf score
    d. The tf term is computed over the whole document collection

7. Which of the following tasks would typically not be solved by clustering?
   a. Community detection in social networks.
   b. Discretization of continuous features.
   c. Spam detection in an email system (CORRECT)
   d. Detection of latent topics in a document collection

8. Which one is false about Label Propagation?

   a. The labels are inferred using the labels that are known apriori.
   b. It can be interpreted as a random walk model.
   c. Propagation of labels through high degree nodes are penalized by low abandoning probability. (CORRECT)
   d. Injection probability should be higher when labels are obtained from experts than by crowdworkers.

2021 Final - MCQ

Question 1

We want to return, from the two posting lists below, the top-2 documents matching a query using Fagin's algorithm with the aggregation function taken as the sum of the tf-idf weights. How many entries (total of both lists) are accessed in the first phase of the algorithm performing round robin starting at List 1 (i.e., before performing the random access)?

List 1

Document	tf-idf
d3	0.8
d2	0.6
d1	0.5
d4	0.4

List 2

Document	tf-idf
d1	0.8
d3	0.6
d4	0.5
d2	0.4

a. 4
b. 8
c. 6 (CORRECT)
d. 2

Question 2

Which of the following is **true** regarding inverted files?

a. Storing differences among word addresses reduces the size of the postings file (CORRECT)
b. Varying length compression is used to reduce the size of the index file
c. The space requirement for the postings file is $O(n\beta)$, where β is generally between 0.4 and 0.6

d. Inverted files prioritize efficiency on insertion over efficiency on search

Question 3

The number of non-zero entries in a column of a term-document matrix indicates:

a. none of the other responses is correct
b. how relevant a term is for a document
c. how often a term of the vocabulary occurs in a document
d. how many terms of the vocabulary a document contains (CORRECT)

Question 4

Which of the following is **true** for community detection in social graphs?

a. If n cliques of the same order are connected cyclically with n edges, then the Louvain algorithm will always detect the same communities, independently of the order of processing of the nodes. (CORRECT)
b. The Louvain algorithm always creates a hierarchy of communities with a common root.
c. The Louvain algorithm is efficient for small networks, while the Girvan-Newman algorithm is efficient for large networks.
d. The result of the Girvan-Newman algorithm can depend on the order of processing of nodes whereas for the Louvain algorithm this is not the case.

Question 5

When using matrix factorization for information extraction the entries of the matrix are obtained

a. from both text and a knowledge base (CORRECT)
b. from a knowledge base represented as text
c. from text
d. from a knowledge base

Question 6

For the number of times the apriori algorithm and the FPgrowth algorithm for association rule mining are scanning the transaction database the following is true

a. fpgrowth and apriori can have the same number of scans (CORRECT)
b. apriori cannot have fewer scans than fpgrowth
c. fpgrowth has always strictly fewer scans than apriori
d. all three above statements are false

Question 7

Why is non-discounted cumulative gain used as evaluation metrics for recommender systems

a. because it is more accurate than retrieval metrics, like precision and recall
b. because often only the top recommendations are considered by the user (CORRECT)
c. because it allows to consider the financial value of recommended items
d. because it considers the predicted ratings of all items that have not been rated by the user

Question 8

Which of the following models for generating vector representations for text require to precompute the frequency of co-occurrence of words from the vocabulary in the document collection

a. CBOW
b. Fasttext
c. Glove (CORRECT)
d. LSI

1. Assume documents d_1 and d_2 are two documents where d_1 is the string S and d_2 is the string S+" "+S. Given a query Q, which document will be ranked higher by a cosine-distance-based TF-IDF retrieval model?

a. d_1
b. d_2
c. They will have the same rank (CORRECT)
d. It depends on the query Q

Explanation: the two document will have exactly the same direction, therefore given any vector representation for Q, they will always have the same rank (as it's a cosine-distance-based retrieval)

2. Run the PageRank algorithm on the following graph. What would be the node with the highest PageRank after the 2nd iteration?
Note: In Iteration 0 we assign uniform weights to all nodes. You need to run for the next two iterations: Iteration 1 and Iteration 2.

a. A
b. B
c. C (CORRECT)
d. D

Explanation:

	Iteration 0	Iteration 1	Iteration 2	Ranking
A	¼	1/12	1.5/12	4
B	¼	2.5/12	2/12	3
C	**¼**	**4.5/12**	**4.5/12**	**1**
D	¼	4/12	4/12	2

3. You have a chain of pages where each page links to the next. Additionally, every page in the chain links back to the first page. How will the PageRank <u>probability</u> of the first page behave, using basic PageRank without random jumps, as the chain growths?

a. It will converge to 0
b. It will converge to ½ (CORRECT)
c. It will converge to 1
d. It will converge to infinity

4. What is TRUE regarding Item-based Collaborative Filtering?

a. It does leverage item description
b. It can recommend niche or new items (CORRECT)
c. It recommends items by finding similar users
d. None of the above

5. Using Matrix Factorization we have ended up with two matrices representing the user preferences (for 4 users: A, B, C, D) and item preferences (For 5 items: 1, 2, 3, 4, 5) as shown below.

$$U = \begin{bmatrix} 3 & 0 \\ 2 & 2 \\ 4 & 4 \\ 0 & 4 \end{bmatrix} \qquad I = \begin{bmatrix} 1 & 3 \\ 0 & 4 \\ 4 & 3 \\ 1 & 2 \\ 4 & 0 \end{bmatrix}$$

Which top 2 users, the item 3 should be recommended to?

a. A, B
b. B, C (CORRECT)
c. C, D
d. A, D

Explanation:
The UxI matrix will be the following:

	I1	I2	**I3**	I4	I5
UA	3	0	12	3	12
UB	8	8	**14**	6	8
UC	16	16	**28**	12	16
UD	12	16	12	8	0

And the normalized version of it:

	I1	I2	**I3**	I4	I5
UA	0	0	1.5	0	1.5
UB	1	1	**2**	1	1
UC	2	2	**3.5**	3	2
UD	1.5	2	1.5	1	0

6. You have the following sentence: “The **dollar** index dropped today around 0.5% in New York Stock Exchange” and you want to do Entity Linking for the word “**dollar**”.
You retrieve in the KG different nodes that can be related to the mention of “dollar” and end up in an entity graph with the following properties:

KG Node	**Out-degree**	**In-degree**
United States dollar	2	4
Canadian dollar	7	0
Australian dollar	0	6

Using Personalized Pagerank ranking, starting in the entity graph with a node related to the mention “New York Stock Exchange”, which one of the following is always TRUE:

a. P(“Canadian dollar”) <= P(“Australian dollar”) (CORRECT)
b. P(“United States dollar”) < P(“Australian dollar”)
c. P(“United States dollar”) <= P(“Canadian dollar”)
d. None of the above

Explanation: Since the in-degree of the Canadian dollar is zero, in PPR, we never reach this node, for the prob will be zero.

Question 1

If for the χ^2 statistics for a binary feature, we obtain $P(\chi^2 | DF = 1) < 0.05$, this means:

a. **That the class labels depends on the feature**
b. That the class label is independent of the feature
c. That the class label correlates with the feature
d. No conclusion can be drawn

Question 2

Given a document collection, if we change the ordering of the words in the documents, which of the following will not change?

a. **Singular values in LSI**
b. The entities extracted using HMM
c. The embedding vectors produced by Word2vec
d. All the previous will change

Question 3

We want to return, from the two posting lists below, the top-2 documents matching a query using Fagin's algorithm with the aggregation function taken as the sum of the tf-idf weights. How many entries (total of both lists) are accessed in the first phase of the algorithm performing round robin starting at List 1 (i.e., before performing the random access)?

List 1			List 2	
document	tf-idf		document	tf-idf
d3	0.8		d1	0.8
d2	0.6		d3	0.6
d1	0.5		d4	0.5
d4	0.4		d2	0.4

a. 3
b. 4
c. **5**
d. 6

Question 4

In the physical representation of an inverted file, the size of the index file is typically in the order of (where n is the number of documents):

a. O(log(n))
b. **O(sqrt(n))**
c. O(n)
d. O(n^2)

Question 5

Which is **true** about the use of entropy in decision tree induction?

a. The entropy of the set of class labels of the samples from the training set at the leaf

level is always 0

b. We split on the attribute that has the highest entropy
c. **The entropy of the set of class labels of the samples from the training set at the leaf level can be 1**
d. We split on the attribute that has the lowest entropy

<u>Question 6</u>

Given the *2-itemsets* {1, 2}, {1, 3}, {1, 5}, {2, 3}, {2, 5}, when generating the *3-itemset* we will:

a. Have **4** *3-itemsets* after the join and **4** *3-itemsets* after the prune
b. **Have 4 *3-itemsets* after the join and 2 *3-itemsets* after the prune**
c. Have **3** *3-itemsets* after the join and **3** *3-itemsets* after the prune
d. Have **2** *3-itemsets* after the join and **2** *3-itemsets* after the prune

Distributed Information Systems: Spring Semester 2018 - Quiz 1

Student Name: _____
Date: May 18 2018
Student ID: _____
Total number of questions: 8
Each question has a single answer!

_____

1. **Data being classified as unstructured or structured depends on the:**
   A. Degree of abstraction
   B. Level of human involvement
   C. Type of physical storage
   D. Amount of data

2. **Which of the following is an advantage of Vector Space Retrieval model?**
   A. No theoretical justification is needed why the model works
   B. Produces provably correct query results
   C. Enables ranking of query results according to cosine similarity function
   D. Allows to retrieve documents that do not contain any of the query terms

3. **Which of the following is *true*?**
   A. High precision implies low recall
   B. High precision hurts recall
   C. High recall hurts precision
   D. High recall implies low precisions

Comment: C) was the intended answer but B is true in some cases. Assume that there are 100 people, 3 of them has cancer (positive) and 97 of them has not. You can force your classification algorithm to find all 3 people with cancer, but in doing so it could also misclassify 1 person as he has cancer as well, so you get %96 precision but %100 recall. Or you can just say everyone is healthy, which awards you with %97 precision but %0 recall (congratz)

4. **Recall can be defined as:**
   A. P(relevant documents | retrieved documents)
   B. P(retrieved documents | relevant documents)
   C. P(retrieved documents | number of documents)
   D. P(relevant documents | number of documents)

5. **Thang, Jeremie and Tugrulcan have built their own search engines. For a query Q, they got precision scores of 0.6, 0.7, 0.8 respectively. Their F1 scores (calculated by same parameters) are same. Whose search engine has a higher recall on Q?**

A. **Thang**
B. Jeremie
C. **Tugrulcan**
D. We need more information

Comment: The intended answer was A) Thang. However, the question intended to say that their F1 scores as same as each other, but some students thought their F1 scores are same as their precision scores, which rendered the answer C.

6. **The number of non-zero entries in a column of a term-document matrix indicates:**
   A. how many terms of the vocabulary a document contains
   B. how often a term of the vocabulary occurs in a document
   C. how relevant a term is for a document
   D. **none of the other responses is correct**

7. **Which one of the following is *wrong*. Schema mapping is used to:**
   A. Overcome semantic heterogeneity
   B. Reconcile different logical representations of the same domain
   C. **Optimize the processing of queries**
   D. Support schema evolution of databases

8. **In a Ranked Retrieval result, the result at position k is non-relevant and at k+1 is relevant. Which of the following is *always* true (P@k and R@k are the precision and recall of the result set consisting of the k top ranked documents)?**
   A. P@k-1 > P@k+1
   B. P@k-1 = P@k+1
   C. **R@k-1 < R@k+**
   D. R@k-1 = R@k+1

Distributed Information Systems: Spring Semester 2018 - Quiz 2

Student Name: _____
Date: March 22 2018
Student ID: _____
Total number of questions: XXX
Each question has a single answer!

1. Which of the following is TRUE when comparing Vector Space Model (VSM) and Probabilistic Language Model (PLM)? (Slide 73 Week 2)
   - ☐ a. Both VSM and PLM require parameter tuning
   - ☐ b. Both VSM and PLM use collection frequency in the model
   - ☐ **c. Both VSM and PLM take into account multiple term occurrences**
   - ☐ d. Both VSM and PLM are based on a generative language model

2. Which of the following is WRONG about inverted files? (Slide 24,28 Week 3)
   - ☐ a. The space requirement for the postings file is O(n)
   - ☐ **b. Variable length compression is used to reduce the size of the index file**
   - ☐ c. The index file has space requirement of O(n^beta), where beta is about ½
   - ☐ d. Storing differences among word addresses reduces the size of the postings file

3. The SMART algorithm for query relevance feedback modifies? (Slide 11 Week 3)
   - ☐ a. The original document weight vectors
   - ☐ **b. The original query weight vectors**
   - ☐ c. The result document weight vectors
   - ☐ d. The keywords of the original user query

4. What is the benefit of LDA over LSI?
   - ☐ a. LSI is sensitive to the ordering of the words in a document, whereas LDA is not
   - ☐ **b. LDA has better theoretical explanation, and its empirical results are in general better than LSI's**
   - ☐ c. LSI is based on a model of how documents are generated, whereas LDA is not
   - ☐ d. LDA represents semantic dimensions (topics, concepts) as weighted combinations of terms, whereas LSI does not

5. How does LSI querying work?
   - ☐ a. The query vector is treated as an additional term; then cosine similarity is computed

- ☐ b. The query vector is transformed by Matrix S; then cosine similarity is computed
- ☐ **c. The query vector is treated as an additional document; then cosine similarity is computed**
- ☐ d. The query vector is multiplied with an orthonormal matrix; then cosine similarity is computed

6. In general, what is true regarding Fagin's algorithm?
   - ☐ a. It performs a complete scan over the posting files
   - ☐ **b. It provably returns the k documents with the largest aggregate scores**
   - ☐ c. Posting files need to be indexed by the TF-IDF weights
   - ☐ d. It never reads more than $(k\ n)^{½}$ entries from a posting list

7. Tugrulcan wanted to plan his next summer vacation so he wrote “best beaches” to his favourite search engine. Little did he know, his favourite search engine was using pseudo-relevance feedback and the top-k documents that are considered relevant were about the beaches only in Turkey. What is this phenomenon called?
   - ☐ a. Query Bias
   - ☐ b. Query Confounding
   - ☐ **c. Query Drift**
   - ☐ d. Query Malfunction

8. Which of the following statements about index merging (when constructing inverted files) is correct?
   - ☐ **a. While merging two partial indices on disk, the inverted lists of a term are concatenated without sorting**
   - ☐ b. Index merging is used when the vocabulary does no longer fit into the main memory
   - ☐ c. The size of the final merged index file is O (n log2 (n) M)), where M is the size of the available memory
   - ☐ d. While merging two partial indices on disk, the vocabularies are concatenated without sorting

Distributed Information Systems: Spring Semester 2018 - Quiz 3

Student Name: _____
Date: April 12 2018
Student ID: _____
Total number of questions: XXX
Each question has a single answer (!)

1. When representing the adjacency list of a Web page in a connectivity server by using a reference list from another Web page, the reference list is searched only in a neighbouring window of the Web page's URL, because:
   - ☐ a. subsequent URLs in an adjacency list have typically small differences
   - ☐ b. typically many URLs in a web page are similar to each other
   - ☐ **c. often many URLs among two pages with similar URL are similar**
   - ☐ d. most extra nodes are found in the neighbouring window.

2. When constructing a word embedding, negative samples are
   - ☐ **a. word - context word combinations that are not occurring in the document collection**
   - ☐ b. context words that are not part of the vocabulary of the document collection
   - ☐ c. all less frequent words that do not occur in the context of a given word
   - ☐ d. only words that never appear as context word

3. Which of the following statements on Latent Semantic Indexing (LSI) and Word Embeddings (WE) is correct
   - ☐ **a. LSI is deterministic (given the dimension), whereas WE is not**
   - ☐ **b. LSI does not take into account the order of words in the document, whereas WE does**
   - ☐ **c. The dimensions of LSI can be interpreted as concepts, whereas those of WE cannot**
   - ☐ d. LSI does take into account the frequency of words in the documents, whereas WE does not.

Comment: The question intended to ask which one is incorrect :)

4. Given the following list of transactions: {apple,milk}, {milk, bread}, {apple, bread, milk}, {bread}
   - ☐ a. milk -> apple has support 1/2 and confidence 1
   - ☐ b. milk -> bread has support 1/2 and confidence 1
   - ☐ c. bread -> milk has support 1/2 and confidence 1
   - ☐ **d. apple -> milk has support 1/2 and confidence 1**

5. Given the 2-itemsets {1,2}, {1,5}, {2,5}, {1,4}, {1,3}, when generating the 3-itemsets we will…
   - ☐ a. Generate 5 3-itemsets after the join and 2 3-itemsets after the prune
   - ☐ **b. Generate 6 3-itemsets after the join and 1 3-itemsets after the prune**
   - ☐ c. Generate 4 3-itemsets after the join and 1 3-itemsets after the prune
   - ☐ d. Generate 4 3-itemsets after the join and 2 3-itemsets after the prune

6. Given the following teleporting matrix (E) for nodes A, B and C:

$$\begin{matrix} [0 & ½ & 0] \\ [0 & 0 & 0] \\ [0 & ½ & 1] \end{matrix}$$

   and making no assumptions about the link matrix (R), which of the following is **correct**:

   - ☐ a. A random walker can never reach node A
   - ☐ b. A random walker can never leave node A
   - ☐ c. A random walker can always leave node C
   - ☐ **d. A random walker can always leave node B**

   Reminder: columns are the probabilities to leave the respective node.

7. When computing PageRank iteratively, the computation ends when:
   - ☐ **a. The norm of the difference of rank vectors of two subsequent iterations falls below a predefined threshold**
   - ☐ b. The difference among the eigenvalues of two subsequent iterations falls below a predefined threshold
   - ☐ c. All nodes of the graph have been visited at least once
   - ☐ d. The probability of visiting an unseen node falls below a predefined threshold

8. For his awesome research, Tugrulcan is going to use the Pagerank with teleportation and HITS algorithm, not on a network of webpages but on the retweet network of Twitter! The retweet network is a directed graph, where nodes are users and an edge going out from a user A and to a user B means that "User A retweeted User B". Which one is FALSE about a Twitter bot that retweeted other users frequently but got never retweeted by other users or by itself?
   - ☐ a. It will have a non-zero hub value.
   - ☐ b. It will have an authority value of zero.
   - ☐ **c. It will have a pagerank of zero.**
   - ☐ **d. Its authority value will be equal to the hub value of a user who never retweets other users.**

Comment: The intended answer was C. There is no general rule regarding the self-links as the paper proposing the HITS algorithm did not specify it, so we also accepted d.

Distributed Information Systems: Spring Semester 2018 - Quiz 4

Student Name: _____
Date: April 26 2018
Student ID: _____
Total number of questions: 8
Each question may have a single answer!

1. **In a FP tree, the leaf nodes are the ones with:**
   a. Lowest confidence
   b. Lowest support
   c. Least in the alphabetical order
   d. None of the above

Comment: The intended (and to me, the only correct) answer was b. However it is also possible to say that the leaf nodes are the ones with lowest frequency, that's why we also accepted d.

2. **Suppose that an item in a leaf node N exists in every path. Which one is correct?**
   a. N co-occurs with its prefix in every transaction.
   b. For every node p that is a parent of N in the fp tree, confidence(p->n) = 1
   c. N's minimum possible support is equal to the number of paths.
   d. The item N exists in every candidate set.

3. **Fundamentally, why clustering is considered an unsupervised machine learning technique?**
   a. Number of clusters are not known.
   b. The class labels are not known.
   c. The features are not known.
   d. The clusters can be different with different initial parameters.

4. **Which of the following is true for a density based cluster C:**
   a. Any two points in C must be density reachable. Each point belongs to one, and only one cluster
   b. Any two points in C must be density reachable. Border points may belong to more than one cluster
   c. Any two points in C must be density connected. Border points may belong to more than one cluster
   d. Any two points in C must be density connected. Each point belongs to one, and only one cluster

5. **Suppose that q is density reachable <u>from</u> p. The chain of points that ensure this relationship are {t,u,g,r} Which one is <u>FALSE</u>?**
   a. {t,u,g,r} have to be all core points.
   b. p and q will also be density-connected
   c. p has to be a core point
   d. q has to be a border point

6. **What is a correct pruning strategy for decision tree induction?**
    a. Apply Maximum Description Length principle
    b. Stop partitioning a node when either positive or negative samples dominate the samples of the other class
    c. Choose the model that maximizes L(M) + L(M|D)
    d. Remove attributes with lowest information gain

7. **Given the distribution of positive and negative samples for attributes A1 and A2, which is the best attribute for splitting?**

A1	P	N
a	7	0
b	1	4
A2	P	N
x	5	1
y	3	3

    a. A1
    b. A2
    c. They are the same
    d. There is not enough information to answer the question

You can use this website to compute the binary entropy:
http://www.wolframalpha.com/widgets/view.jsp?id=4e095a8fa96257fbf9da2529e930ccd3

8. **When using bootstrapping in Random Forests, the number of different data items used to construct a single tree is:**
    a. smaller than the size of the training data set, with high probability
    b. of order square root of the size of the training set, with high probability
    c. the same as the size of the training data set
    d. subject to the outcome of the sampling process, and can be both smaller or larger than the training set

Distributed Information Systems: Spring Semester 2018 - Quiz 5

Student Name: _____
Date: May 17 2018
Student ID: _____
Total number of questions: 8
Each question hopefully has a single answer!

1. Which of the following is **correct** regarding *Louvain* algorithm?
   - ☐ a. It creates a hierarchy of communities with a common root
   - ☐ b. *Clique* is the only topology of nodes where the algorithm detects the same communities, independently of the starting point
   - ☐ **c. If *n* cliques of the same order are connected cyclically with *n-1* edges, then the algorithm will always detect the same communities, independently of the starting point**
   - ☐ d. Modularity is always maximal for the communities found at the top level of the community hierarchy

2. In *User-Based Collaborative Filtering*, which of the following is **correct,** assuming that all the ratings are positive?
   - ☐ a. *Pearson Correlation Coefficient* and *Cosine Similarity* have different value range, but return the same similarity ranking for the users
   - ☐ **b. If the ratings of two users have both variance equal to 0, then their *Cosine Similarity* is maximized**
   - ☐ c. *Pearson Correlation Coefficient* and *Cosine Similarity* have the same value range, but can return different similarity ranking for the users
   - ☐ d. If the variance of the ratings of one of the users is 0, then their *Cosine Similarity* is not computable

3. Which of the following is **correct** regarding *Crowdsourcing*?
   - ☐ a. *Random Spammers* give always the same answer for every question
   - ☐ b. It is applicable only for binary classification problems
   - ☐ c. *Honey Pot* discovers all the types of spammers but not the sloppy workers
   - ☐ d. **The output of *Majority Decision* can be equal to the one of *Expectation-Maximization***

4. Which of the following is **correct** regarding prediction models?
   - ☐ a. Training error being less than test error means overfitting
   - ☐ b. Training error being less than test error means underfitting
   - ☐ **c. Complex models tend to overfit, unless we feed them with more data**
   - ☐ d. Simple models have lower bias than complex models

5. In the χ^2 statistics for a binary feature, we obtain $P(\chi^2 \mid DF = 1) > 0.05$. This means in this case, it is assumed:
   - ☐ a. That the class labels depends on the feature
   - ☐ **b. That the class label is independent of the feature**
   - ☐ c. That the class label correlates with the feature
   - ☐ d. None of the above

6. Which is an appropriate method for fighting skewed distributions of class labels in classification?
   - ☐ a. Include an over-proportional number of samples from the larger class
   - ☐ b. Use leave-one-out cross validation
   - ☐ **c. Construct the validation set such that the class label distribution approximately matches the global distribution of the class labels**
   - ☐ d. Generate artificial data points for the most frequent classes

		Class	
		Fraud	¬Fraud
Classified	Fraud	20	20
	¬Fraud	10	60

7. Considering the results of this fraud classifier, which of the following is **correct**?
   - ☐ **a. The classifier has a precision of 50% and a recall of 66.6%**
   - ☐ b. The classifier has a precision of 75% and a recall of 50%
   - ☐ c. The classifier has a precision of 50% and a recall of 75%
   - ☐ d. The classifier has a precision of 66.6% and a recall of 75%

The following question is cancelled:

8. Which of the following is **correct** regarding community detection?
   - ☐ a. High betweenness of an edge indicates that the communities are well connected by that edge
   - ☐ b. The *Louvain* algorithm attempts to minimize the overall modularity measure of a community graph
   - ☐ c. High modularity of a community indicates a large difference between the number of edges of the community and the number of edges of a null model
   - ☐ d. The *Girvan-Newman* algorithm attempts to maximize the overall betweenness measure of a community graph

Comment: a) Is not correct, yes an edge with high betweenness connects two communities and these two communities will be distinct and won't be connected by many edges. But these communities are not "well-connected" and also answer does not specificy which communities are that edge connects. (Though I must admit this sentence is not well-formed.) b) and d) are not correct.

c) was the intended answer. Suppose that we take one of the community detected by Louvain algorithm. Then randomize the edges of the whole graph, which we call null model, and take the nodes which belong to the community we found. The community extracted from the original graph would have more edges between the nodes that belong to that community, then the community extracted from the null model. However in c, we accidentally meant the whole graph by the null model, which has the same number of edges as the original graph, which would have more edges than a community.

Distributed Information Systems: Spring Semester 2018 - Quiz 6

Student Name: _____
Date: May 18 2018
Student ID: _____
Total number of questions: 8
Each question really has a single answer!

1. Which of the following is **wrong** regarding *Ontologies*?
   - ☐ a. We can create more than one ontology that conceptualize the same real-world entities
   - ☐ b. Ontologies help in the integration of data expressed in different models
   - ☐ c. Ontologies support domain-specific vocabularies
   - ☐ d. **Ontologies dictate how semi-structured data are serialized**

2. Which of the following is **correct** regarding the use of *Hidden Markov Models (HMMs)* for entity recognition in text documents?
   - ☐ a. HMMs cannot predict the label of a word that appears only in the test set
   - ☐ b. If the smoothing parameter *λ* is equal to *1*, the emission probabilities for all the words in the test set will be equal
   - ☐ **c. When computing the emission probabilities, a word can be replaced by a morphological feature (e.g., the number of uppercase first characters)**
   - ☐ d. The label of one word is predicted based on all the previous labels

3. Which of the following statements is **correct** concerning the use of *Pearson's Correlation* for user-based collaborative filtering?
   - ☐ **a. It measures whether different users have similar preferences for the same items**
   - ☐ b. It measures how much a user's ratings deviate from the average ratings
   - ☐ c. It measures how well the recommendations match the user's preferences
   - ☐ d. It measures whether a user has similar preferences for different items

4. Which of the following statements is **wrong** regarding RDF?
   - ☐ a. An RDF statement would be expressed in SQL as a tuple in a table
   - ☐ **b. Blank nodes in RDF graphs correspond to the special value NULL in SQL**
   - ☐ c. The object value of a type statement corresponds to a table name in SQL
   - ☐ d. RDF graphs can be encoded as SQL databases

5. Which of the following statements is **correct** in the context of information extraction?
   - ☐ **a. A confidence measure that prunes too permissive patterns discovered with bootstrapping can help reducing semantic drift**
   - ☐ b. The bootstrapping technique requires a dataset where statements are labelled
   - ☐ c. Distant supervision typically uses low-complexity features only, due to the lack of training data
   - ☐ d. For supervised learning, sentences in which NER has detected no entities are used as negative samples

6. Which of the following is **correct** regarding schemas and ontologies?

- ☐ a. An ontology is created from constructing mappings between schemas
- ☐ **b. Ontologies can be used for reasoning about different schemas**
- ☐ c. Ontologies always require a schema
- ☐ d. Semi-structured data cannot have a schema

7. *Dude said "I like bowling"*. With how many statements can we express this sentence using *RDF Reification*?
   - ☐ a. We cannot
   - ☐ b. 1
   - ☐ c. 3
   - ☐ **d. 5**

8. Modularity of a social network always:
   - ☐ a. Increases with the number of communities
   - ☐ b. **Increases when an edge is added between two members of the same community**
   - ☐ c. Decreases when new nodes are added to the social network that form their own communities
   - ☐ d. Decreases if an edge is removed

Quiz - 2021

1. Information extraction:

- [ ] Necessarily requires training data.
- [ ] Is used to identify characteristic entities in a document.
- [ ] Is always bootstrapped by using ontologies.
- [x] ~~Can be used to populate ontologies.~~

2. What is **TRUE** regarding Fagin's algorithm?

- [ ] Posting files need to be indexed by TF-IDF weights
- [ ] It performs a complete scan over the posting files
- [ ] It never reads more than (kn)½ entries from a posting list
- [x] ~~It provably returns the k documents with the largest aggregate scores~~

3. Which of the following statements on Latent Semantic Indexing (LSI) and Word Embeddings (WE) is false?

- [ ] The dimensions of LSI can be interpreted as concepts, whereas those of WE cannot
- [ ] LSI does not depend on the order of words in the document, whereas WE does
- [ ] LSI is deterministic (given the dimension), whereas WE is not
- [x] ~~LSI does take into account the frequency of words in the documents, whereas WE with negative sampling does not~~

4. When constructing a word embedding, what is **TRUE** regarding negative samples?

- [x] ~~They are oversampled if less frequent~~
- [ ] Their frequency is decreased down to its logarithm
- [ ] They are words that do not appear as context words
- [ ] They are selected among words that are not stop-words

5. A page that points to all other pages but is not pointed by any other page would have:

- [ ] Nonzero authority
- [ ] Zero hub
- [x] ~~Nonzero PageRank~~
- [ ] None of the above

6. When computing PageRank iteratively, the computation ends when:

- [ ] The difference among the eigenvalues of two subsequent iterations falls below a predefined threshold

- [x] ~~The norm of the difference of rank vectors of two subsequent iterations falls below a predefined threshold~~
- [ ] The probability of visiting an unseen node falls below a predefined threshold
- [ ] All nodes of the graph have been visited at least once

7. In Ranked Retrieval, the result at position k is non-relevant and at k+1 is relevant. Which of the following is always true?
Hint: P@k and R@k are the precision and recall of the result set consisting of the k top-ranked documents.

- [ ] P@k-1>P@k+1
- [ ] R@k-1=R@k+1
- [x] ~~R@k-1<R@k+1~~
- [ ] P@k-1=P@k+1

8. Which of the following is **TRUE** regarding community detection?

- [ ] The high betweenness of an edge indicates that the communities are well connected by that edge
- [ ] The Girvan-Newman algorithm attempts to maximize the overall betweenness measure of a community graph
- [x] ~~The high modularity of a community indicates a large difference between the number of edges of the community and the number of edges of a null model~~
- [ ] The Louvain algorithm attempts to minimize the overall modularity measure of a community graph

9. What is **WRONG** regarding the Transformer model?

- [x] ~~Its computation cannot be parallelized compared to LSTMs and other sequential models.~~
- [ ] It uses a self-attention mechanism to compute representations of the input and output.
- [ ] Its complexity is quadratic to the input size.
- [ ] It captures the semantic context of the input.

10. In User-Based Collaborative Filtering, which of the following is **TRUE**?

- [ ] Pearson Correlation Coefficient and Cosine Similarity have the same value range and return the same similarity ranking for the users.
- [x] ~~Pearson Correlation Coefficient and Cosine Similarity have different value ranges and can return different similarity rankings for the users~~
- [ ] Pearson Correlation Coefficient and Cosine Similarity have different value ranges, but return the same similarity ranking for the users
- [ ] Pearson Correlation Coefficient and Cosine Similarity have the same value range but can return different similarity rankings for the users

11. Which of the following is **TRUE** for Recommender Systems (RS)?

- [ ] The complexity of the Content-based RS depends on the number of users
- [ ] Item-based RS need not only the ratings but also the item features
- [ ] Matrix Factorization is typically robust to the cold-start problem.
- [x] ~~Matrix Factorization can predict a score for any user-item combination in the dataset.~~

12. Considering the transaction below, which one is **WRONG**?

Transaction ID	Items Bought
1	Tea
2	Tea, Yoghurt
3	Tea, Yoghurt, Kebap
4	Kebap
5	Tea, Kebap

- [ ] {Yoghurt} -> {Kebab} has 50% confidence
- [ ] {Yoghurt, Kebap} has 20% support
- [ ] {Tea} has the highest support
- [x] ~~{Yoghurt} has the lowest support among all itemsets~~

13. Suppose that in a given FP Tree, an item in a leaf node N exists in every path. Which of the following is **TRUE**?

- [ ] N co-occurs with its prefixes in every transaction
- [ ] For every node P that is a parent of N in the FP tree, confidence (P->N) = 1
- [x] ~~{N}'s minimum possible support is equal to the number of paths~~
- [ ] The item N exists in every candidate set

14. Which of the following properties is part of the RDF Schema Language?

- [ ] Description
- [ ] Type
- [ ] Predicate
- [x] ~~Domain~~

15. Which of the following is wrong regarding Ontologies?

- [ ] We can create more than one ontology that conceptualizes the same real-world entities
- [ ] Ontologies help in the integration of data expressed in different models
- [x] ~~Ontologies dictate how semi-structured data are serialized~~
- [ ] Ontologies support domain-specific vocabularies

Answers to quizzes

1.1 INTRODUCTION TO INFORMATION RETRIEVAL

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A retrieval model attempts to capture …

1. the interface by which a user is accessing information
2. the importance a user gives to a piece of information for a query
3. the formal correctness of a query formulation by user
4. the structure by which a document is organised

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

The role of a retrieval model is to capture the notion of relevance, which means what documents a user would consider as relevant for a given query. The user interface is an orthogonal question. If the retrieval model involves a notion of formal correctness of queries, it is also verified in a user-interfacing application. The structure of a document can be considered by a retrieval model, but is not modelled by it.

Full-text retrieval refers to the fact that …

1. the document text is grammatically fully analyzed for indexing
2. queries can be formulated as texts
3. all words of a text are considered as potential index terms
4. grammatical variations of a word are considered as the same index terms

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

The term full-text retrieval has been introduced to the fact that the text is considered as a bag of words. This means that any grammatical structure us ignored. Considerung grammatical variations of a word as the same index term is achieved by using stemming.

The entries of a term-document matrix indicate …

1. how many relevant terms a document contains
2. how frequent a term is in a given document
3. how relevant a term is for a given document
4. which terms occur in a document collection

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

As per definition the entries determine the relevance of a term to the document. The frequency can be such an indicator, but is not necessarily so. An entry in the matrix does not indicate which terms occur an a document collection, but whether a term occurs (otherwise the entry would be a set). It does also not indicate how many relevant terms a document contains, sicne the entry is linked to a single term.

Let the Boolean query be represented by {(1, 0, -1), (0, -1, 1)} and the document by (1, 0, 1). The document …

1. matches the query because it matches the first query vector
2. matches the query because it matches the second query vector
3. does not match the query because it does not match the first query vector
4. does not match the query because it does not match the second query vector

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

The document vector (1,0,1) matches the query vector (0,-1,1), since the first word is not relevant for the query, the second word has to be absent from the document, which is the case since the value is 0, and the third word has to be present in the document, which is the case since the value is 1.

The term frequency of a term is normalized …

1. by the maximal frequency of all terms in the document
2. by the maximal frequency of the term in the document collection
3. by the maximal frequency of any term in the vocabulary
4. by the maximal term frequency of any document in the collection

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

The standard normalization of term frequency is by the maximal frequency of all terms occuring in the document. Note the denominator in the definition, which expresses that you take for all words k in the vocabulary the frequency of the word in the document j. This implies that the most frequent word will have normalized term frequency 1. It implictly also takes into account the document length, since in a longer document the most frequent words will be more frequent and this the influence of a single word will be smaller.

The inverse document frequency of a term can increase …

1. by adding the term to a document that contains the term
2. by removing a document from the document collection that does not contain the term
3. by adding a document to the document collection that contains the term
4. by adding a document to the document collection that does not contain the term

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 4

The inverse document frequency of a term increases if the proportion of documents containing the term decreases. By adding a document to the collection that does not contain the term this is the case. In the first case the idf remains unchanged, in the second and thrid case it decreases.

Which is the “best” classifier?

Classifier 1		Class A	Class B
Classified	A	45	20
	B	5	30

Classifier 2		Class A	Class B
Classified	A	40	10
	B	10	40

A. Classifier 1

B. Classifier 2

C. Both are equally good

©2021, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

Without any further context, classifier B acheives 80% accuracy, which is higher than 75%.

Which is the “best” classifier?

Classifier 1

		Class: Cancer	Class: ¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

Classifier 2

		Class: Cancer	Class: ¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

A. Classifier 1

B. Classifier 2

C. Both are equally good

©2021, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

Being aware of the context, we understand that the objective is to miss as few cancers as possible. Classifier 2 misses 1 cancer out of 5, whereas cassifier 1 misses 1 out of 9, which is more suitable under this objective.

If the top 100 documents contain 50 relevant documents …

1. the precision of the system at 50 is 0.25
2. the precision of the system at 100 is 0.5
3. the recall of the system is 0.5
4. All of the above

©2021, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

Since among the retrieved 100 documents 50 are relevant, the precision at 100 is clearly 0.5. We cannot say anything about the precision at 50, since we do not know how many of the relevant docuemnts are found in the first 50 documents. We also cannot say anything about the recall, since we do not now how many relevant documents do exist.

If retrieval system A has a higher precision at k than system B …

1. the top k documents of A will have higher similarity values than the top k documents of B
2. the top k documents of A will contain more relevant documents than the top k documents of B
3. A will recall more documents above a given similarity threshold than B
4. the top k relevant documents in A will have higher similarity values than in B

©2021, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

If sysetm A has higher precision at k than system B, then b defition it will contain more relevant documents in the first k retrieved documents than B. The absolute similarity values that different retrieval systems produces does not give any information on how they will rank the results.

Let the first four documents retrieved be R N N R. Then the MAP is

1. 1/2
2. 3/4
3. 2/3
4. 5/6

©2021, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Anwer 2

P@1 = 1 and P@4 = ½. The average of the two values is this ¾.

Consider the document:
"Information retrieval is the task of finding the documents satisfying the information needs of the user"

Using MLE to estimate the unigram probability model, what is $P(\text{the}|M_d)$ and $P(\text{information}|M_d)$?

1. 1/16 and 1/16
2. 1/12 and 1/12
3. 1/4 and 1/8
4. 1/3 and 1/6

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

The total number of terms in the document is 16.

The term "the" appears 4 times, the term information 2 times.

Therefore the probabilities are 4/16 and 2/16.

Consider the following document

d = “information retrieval and search”

1. P(information search | M_d) > P(information | M_d)
2. P(information search | M_d) = P(information | M_d)
3. P(information search | M_d) < P(information | M_d)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

The probability P(t1 t2 | M_d) is the product of the probabilities P(t1 | M) and P(t2 | M).

Since P(t2 | M) < 1, necessarily P(t1 t2 | M_d) < P(t1| M_d)

Note that this is not a problem in the sense that we are never comparing probabilities of different queries in retrieval, but probabilities of different documents for the same query.

Can documents which do not contain any keywords of the original query receive a positive similarity coefficient after relevance feedback?

1. No
2. Yes, independent of the values β and γ
3. Yes, but only if $\beta > 0$
4. Yes, but only if $\gamma > 0$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

It is possible that a document that does not contain any terms of the original query can contain terms of another document that has been selected for the set of relevant documents D_r. If $\beta > 0$. This will result in a positive similarity value. Note that all weights in document and query vectors are positive, so the scalar product of two such vectors will be positive.

Which year Rocchio published his work on relevance feedback?

A. 1965

B. 1975

C. 1985

D. 1995

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

Rocchio, J. J., and Gerard Salton. "Information search optimization and interactive retrieval techniques." *Proceedings of the November 30--December 1, 1965, fall joint computer conference, part I*. 1965.

In vector space retrieval each row of the matrix M corresponds to

A. A document
B. A concept
C. A query
D. A term

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

M is an mxn matrix, with m rows corresponding to the m terms in the vocabulary.

Applying SVD to a term-document matrix M. Each concept is represented in K

A. as a singular value
B. as a linear combination of terms of the vocabulary
C. as a linear combination of documents in the document collection
D. as a least squares approximation of the matrix M

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

K is mxr matrix, where the columns correspond to vectors. These vectors correspond to a linear combination of the m terms of the vocabulary.

The number of term vectors in the matrix K_s used for LSI

A. Is smaller than the number of rows in the matrix M
B. Is the same as the number of rows in the matrix M
C. Is larger than the number of rows in the matrix M

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

K_s is a mxs matrix, where each row corresponds to a term in the vocabulary, as for M. The number of columns s is smaller than the number of columns in the original matrix K.

A query transformed into the concept space for LSI has …

A. s components (number of singular values)
B. m components (size of vocabulary)
C. n components (number of documents)

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

The transformed query is a vector over the number of selected concepts s.

Question

A row of matrix $W^{(c)}$ represents

1. How relevant each word is for a dimension
2. How often a context word c co-occurs with all words
3. A representation of word c in concept space

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

The rows of the matrix correspond to the dimensions of the embedding. Each entry in the row corresponds to a word from the vocabulary. Therefore the row represents the importance of each word for a given dimension.

Question

Which of the following functions is not equal to the three others?

1. $f(w, c)$
2. $f_\theta(w, c)$
3. $\boldsymbol{f(w, c)}$
4. $\sigma(\boldsymbol{c} \cdot \boldsymbol{w})$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

$f(w, c)$ is the function to be approximated. The three other functions are the same, with a more detailed specification for answers 2, 3 and 4.

Question

From which data samples the embeddings are learnt?

1. Known embeddings for (w,c) pairs
2. Frequency of occurrences of (w,c) pairs in the document collection
3. Approximate probabilities of occurrences of (w,c) pairs
4. Presence or absence of (w,c) pairs in the document collection

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 4

In the skipgram model the sample data consists of word-context pairs that are present or absent in the document collection.

Question

With negative sampling a set of negative samples is created for

1. For each word of the vocabulary
2. For each word-context pair
3. For each occurrence of a word in the text
4. For each occurrence of a word-context pair in the text

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 4

For each occurrence of a word-context pair, a set of negative samples is produced. Note that this is also different from creating a set of negative samples for each word-context pair, since the same word-context pair can occur multiple times in the document collection.

Question

The loss function is minimized

1. By modifying the word embedding vectors
2. By changing the sampling strategy for negative samples
3. By carefully choosing the positive samples
4. By sampling non-frequent word-context pairs more frequently

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

The loss function is minimized by incrementally modifying the word embedding vectors. Answer 4 refers to an approach to improve the quality of the word embeddings achieved, but not to minimize the loss function.

Question

A word embedding for given corpus …

1. depends only on the dimension d
2. depends on the dimension d and number of iterations in gradient descent
3. depends on the dimension d, number of iterations and chosen negative samples
4. there are further factors on which it depends

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 4

Other factors that can influence the outcome of the optimization are

- The order in which the documents are processed
- The initialization of the word embedding vectors

Question

Fasttext speeds up learning by

1. Considering subwords of words
2. By selecting the most frequent phrases in the text as tokens
3. By selecting the most frequent subwords in the text as tokens
4. By pre-computing frequencies of n-grams

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

Answer 1 improves the quality of embeddings, but may slow down learning.
Answer 4 speeds up the selection of frequent phrases, but not learning.

Question

The most important difference between Glove and skipgram is

1. That Glove considers the complete context of a word
2. That Glove computes a global frequency for word-context pair occurrences
3. That Glove uses a squared error loss function
4. That Glove does not differentiate words and context words

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

The main difference between Glove and earlier methods, including skipgram and CBOW, is the computation of global co-occurrence counts. This is an additional processing step, but provides additional information on the global statistics. Answer 3 refers to a difference that is rather a consequence of using a global statistics. As for answer 4, in a sense also skipgram does not really distinguish between words and context words.

Given 3 users with ratings…

u1:	1	3
u2:	2	4
u3:	1	4

A. $Sim_{corr}(u1, u2) > Sim_{corr}(u1, u3)$
B. $Sim_{corr}(u1, u2) = Sim_{corr}(u1, u3)$
C. $Sim_{corr}(u1, u2) < Sim_{corr}(u1, u3)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

With Pearson correlation, similarity is computed using vectors that are centered around the mean and the normalized. Therefore, after normalization all three vectors are the same, namely 1/sqrt(2) * (-1,1).

Item-based collaborative filtering addresses better the cold-start problem because …

A. usually there are fewer items than users
B. it uses ratings from items with similar content
C. item similarities can be pre-computed
D. none of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

If a user has no ratings, also item-based collaborative filtering cannot make a prediction on the user's rating of an unknown item.

If the item in question has not ratings, not similar items can be determined and no predication can be made as well.

For a user that has not done any ratings, which method can make a prediction?

A. User-based collaborative RS
B. Item-based collaborative RS
C. Content-based RS
D. None of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

User-base and item-based collaborative filtering cannot make predictions without previous ratings of the users.

Since the user has not rated any item, also content-based filtering is not applicable.

For an item that has not received any ratings, which method can make a prediction?

A. User-based collaborative RS
B. Item-based collaborative RS
C. Content-based RS
D. None of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

Based on the content, similat items can be found, even if the item has not yet received a rating.

Question

How does matrix factorization address the issue of missing ratings?

A. It uses regularization of the rating matrix
B. It performs gradient descent only for existing ratings
C. It sets missing ratings to zero
D. It maps ratings into a lower-dimensional space

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

In gradient descent only data on existing ratings (user-item pairs) are exploited.

Question

With matrix factorization one estimates ratings of unrated items

A. By retrieving the corresponding item from a user vector
B. By retrieving the corresponding item from an item vector
C. By computing the product of a user and an item vector
D. By looking up the rating in an approximation of the original rating matrix

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

For estimating a new rating the latent representations of the corresponding user and item are used to compute an estimation. In principle, one could also consider Answer D as correct, but this would imply that all ratings are precomputed, which would be a waste of resources, if only few ratings need to be known.

Question

Which of the following matrices is not sparse?

A. The matrix W

B. The matrix $\hat{R}$

C. The matrix RW

D. More than one of the above are sparse

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

Only the matrix W is sparse. The two other matrices are the same, and contain rating estimations for all items. Therefore, they are not sparse.

Question

If users rate about 1% of items, and W has sparsity 1%, how many multiplications are needed to recommend the top-k elements?

A. $|I| * |I| * 0.0001$

B. $|I| * |I| * k * 0.01$

C. $|U| * |I| * k * 0.0001$

D. $|U| * |I| * 0.0001$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

For finding the top k ratings, for each non-rated item of the user (about |I|), the rating vector of the user has to be multiplied with the corresponding item vector in W. The user rating vector has about |I| * 0.01 non-zero entries, as well as the item vector. Computing the scalar product of a dense vector takes |I| multiplictaions. Therefore, in total |I|^2 * 0.00001 multiplications will be performed.

Question

Assume in top-1 retrieval recommendation 1 is (2, 3, 1) and recommendation 2 is (2, 1, 3)

A. RMSE(rec 1) < RMSE(rec 2) and DCG(rec 1) > DCG(rec 2)
B. RMSE(rec 1) = RMSE(rec 2) and DCG(rec 1) > DCG(rec 2)
C. RMSE(rec 1) < RMSE(rec 2) and DCG(rec 1) = DCG(rec 2)
D. RMSE(rec 1) = RMSE(rec 2) and DCG(rec 1) = DCG(rec 2)

Remark: the optimal method would rank (3,2,1)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

Since DCG considers only the top-k items, in this case the first item, the measure is independent of the ordering of the non top-k items. RMSE will give a better score to rec 1, since the higher rated item 3 is ranked higher in the result.

For which document classifier the training cost is low and inference is expensive?

A. for none
B. for kNN
C. for NB
D. for fasttext

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

For kNN the only cost incurred for training is the computation of document weights, including the idf values. In Naïve Bayes for all words the likelihood of the word to occur in a class has to be computed. Fasttext requires SGD.

How many among the listed classifiers can be used to derive probability estimate of the class label?

1NN, kNN, Rocchio, NB, fasttext

A. 1

B. 2

C. 3

D. 4

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

1NN cannot be used to compute a probability estimate, since it provides a binary decision of which is the closest class label. The same is true for the Rocchio method.

kNN an be used to estimate the probabilities for class labels if k is sufficiiently large. NB and fasttext by defintion provide probability estimates.

Question

The number of parameters of the fasttext classifier and the simple self-attention classifier

1. Are the same
2. Fasttext has more
3. Self-attention has more

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

The simple self-attention based classifier applied before the averaging of the word vectors the self-attention mechanism. This mechanism uses only the word vectors as weights and therefore does nto introduce new parameters into the model.

The relevance determined using the random walker model corresponds to

1. The number of steps a random walker needs to reach a page
2. The probability that the random walker visits the page in the long term
3. The number of incoming links a random walker can use to visit the page
4. The probability that the random walker will visit once the page

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

The relevance measure derived from the random walker model corresponds to the stationary (long-term) probability of the random walker to visit a page. The number of incoming links does have an influence on this probability but is not determining it.

Consider a random jump matrix with entries 1/3 in the first column and 0 otherwise. It means

1. A random walker can always leave node 1 even without outgoing edges
2. A random walker can always reach node 1, even without incoming edges
3. A random walker can always leave node 2, even without outgoing edges
4. none of the above

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

The first column of the matrix corresponds to the probabilities to jump to from node 1 to every other node. Therefore, the random walker when arriving at node 1 can always leave that node by making random jump. On the other hand, node 1 can only be reached when an incoming edge exists. If node 2 has no outgoing edge, the random walker cannot leave it.

When computing HITS, the initial values

1. Are set all to 1
2. Are set all to $\frac{1}{n}$
3. Are set all to $\frac{1}{\sqrt{n}}$
4. Are chosen randomly

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

In the standard formulation of the algorithm the normalization factor is $\frac{1}{\sqrt{n}}$ so that the L2 norm of the vector equals 1.

If the first column of matrix L is (0,1,1,1) and all other entries are 0 then the authority values

1. $(0,1,1,1)$
2. $\left(0, 1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}\right)$
3. $\left(1, 1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}\right)$
4. $(1,0,0,0)$

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

If the first column of matrix L is (0,1,1,1), then node 1 has one link to each other graph. Correspondingly it will be a hub, and the only hub and have hub weight 1. The other nodes will receive the same authority from that node. Thus, their weights will be equally distributed and be therefore $1/\sqrt{3}$ so that the L2 norm of the authority vector equals 1. Note that as consequence node 1 will receive equal weights from the three authority nodes.

Modularity clustering will end up always with a single community at the top level?

A. true
B. Only for dense graphs
C. Only for connected graphs
D. never

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

If the graph is disconnected in distinct subgraphs, modularity will end up with a node representing each of those subgraphs that are not connected to each other and will therefore not be grouped to a community.

Modularity clustering will end up always with the same community structure?

A. true
B. Only for connected graphs
C. Only for cliques
D. false

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

They detailed outcome of modularity clustering depends on the order of processing of the nodes. By changing the processing order different communities may result. For cliques always one community will be formed in one pass (to be shown), but other regular graph structures, e.g. a ring, may also have this property.

$\sigma_{xy}(v)$ of edge 3-4 is …

A. 16
B. 12
C. 9
D. 4

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

We have 4 shortest paths from {1,2} to {5, 6} passing through edge 3-4

In addition, there are 2 shortest paths from 3 to {5, 6} and from 4 to {1, 2} and the path from 3 to 4.

This gives total 9 paths.

When computing path counts for node 1 with BFS, the count at 6 is …

A. 1

B. 2

C. 3

D. 4

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

BFS computes the number of shortest paths from the starting node 1 to every other node of the graph.

Since there exists only one shortest path from 1 to 6, this count is 1.

An HMM model would not be an appropriate approach to identify

A. Named Entities
B. Part-of-Speech tags
C. Concepts
D. Word n-grams

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

Word n-grams can be determined by simple algortihmic search. All other types of information are suitable to be modelled using HMMs.

Which statement is correct?

A. The Viterbi algorithm works because words are independent in a sentence
B. The Viterbi algorithm works because it is applied to an HMM model that makes an independence assumption on the word dependencies in sentences
C. The Viterbi algorithm works because it makes an independence assumption on the word dependencies in sentences
D. The Viterbi algorithm works because it is applied to an HMM model that captures independence of words in a sentence

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

All other answers have some logical problem in the formulation. Answer A would imply that works are independent in a sentence, which is clearly wrong. Answer C states that the algorithm makes an independence assumption. The Viterbi algorithm does not make any assumptions to independence, it is applied to an HMM. Therefore Answer C is inaccurate. Answer D is also not exact, since the HMM models does not capture an existing independence, but simply makes the (wrong) assumption that the independence exists.

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

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

Names entity recognition is a local method that extracts different entities from text independently. In standard NER the fact that different entities might share the names is not taken into account. However, NER performs classification into different entity types.

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

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

Concepts that have been linked to the same mention are competing for the interpretation. Therefore it is not meaningful to take them into account when trying to support an alternative interpretation.

One might wonder why Answer B and C are correct. A node with no outgoing link can still be the end of a path when computing PPR. Therefore it will contribute to the ranking. A node with no incoming link can on the other hand be the start of a path when comuting PPR and therefore contributes to the scoring.

Semi-structured data …

A. is always schema-less
B. always embeds schema information into the data
C. must always be hierarchically structured
D. can never be indexed

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

Answer A: there exist frameworks for providing schemas for semi-structured data, such as XML Schema, or RDF schema we will introduce.

Answer C: hierarchical structure is very common for semi-structured data, but not necessary, as for example for email formats.

Answer D: Indexing of semi-structured data is possible, but cannot rely on a fixed data structure. Therefore the approaches are different from the ones used typical in databases with structure schemas.

Why is XML a document model and not just a general data model?

A. It supports application-specific markup
B. It supports domain-specific schemas
C. It has a serialized representation
D. It uses HTML tags

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C: the notion of “document” is related to the fact that the data is represented as a string. Statements A and B are also correct for XML, but do not justify the notion of document. More accurately, we are referring to text documents here.

An ontology …

A. helps to separate layout issues from the structural representation of data
B. defines a common syntactic framework to represent standardized domain models
C. can be used as a mediation framework for integrating semantically heterogeneous databases

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

Answer A refers to the role of structured document models, such as XML. An ontology is a priori also independent of its specific syntactic representation. It can be used to integrate data from semantically heterogeneous resources.

A basic statement in RDF would be expressed in the relational data model by a table …

A. with one attribute
B. with two attributes
C. with three attributes
D. cannot be expressed in the relational data model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

The canonical representation of an RDF statement in a relational table would be by a table that represents the predictate of the statement and two attributes that are used to represent the subject and object. For example, the statement X isauthor Y would be represented as tuple is-author(X, Y). One might argue that also the predicate might be represented as attribute. This would be resulting in a generic representation with single table used to represent all RDF statements. This is not the standard approach of data modeling used in relational data models.

The type statement in RDF would be expressed in the relational data model by a table …

A. with one attribute
B. with two attributes
C. with three attributes
D. cannot be expressed in the relational data model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

The type relation expresses membership to a sepcific category of elements in a domain. The canonical way to represent this in the relational model is to use a table that lists the elements of the category. For example, if we have elements of type country, and statements of the form X is-a country, we would represent this by tuples of the form country(X). One might argue that one could list the type also as an attribute in a binary table. However, then this corresponds rather to a specific property than a type information. For example, if we have elements of type female and male, we would have a table gender(X, F/M), but we would consider the gender not as type information.

Which is true?

A. Reification is used to produce a more compact representation of complex RDF statements
B. Reification requires to make a statement the subject of another statement
C. Reified statements are always used to make a statement about another statement

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

Answer A is obviously the opposite what happens. As for Answer B, a statement about which we make another statement can also be the object of the statement. For example, author X produced the statement S.

Which of the following are part of the RDF schema language?

A. The « type » statement for RDF resources?
B. The « domain » statement for RDF properties?
C. The « subject » statement for RDF statements?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

You have noticed that the namespace for the type and subject statements is RDF. This makes sense since these two types of statements involve instances that are not part of an RDF schema, wheras the domain statement involves exclusively schema instances.

Which of the following is NOT an (instance-level) ontology?

A. Wordnet
B. WikiData
C. Schema.org
D. Google Knowledge Graph

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

Schema.org provides a schema without any instances of facts, whereas the three other resources provide sets of facts.

Which is true?

A. Hand-written patterns are in general more precise than classifiers
B. Hand-written patterns cannot exploit syntactic features
C. Supervised classifiers do not require any human input
D. Supervised classifiers can only detect typed statements

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

Answer B is exactly the opposite of what is the case, that patterns exploit syntactice features. Classifiers requires human input for labelling. They also can be used to detet untyped statements.

Which is true?

A. Distant supervision requires rules for bootstrapping
B. Classifiers produced with distant supervision are more precise than rules
C. Distant supervision can help to detect rules

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

Distant supervision does not require and rules, as opposed to bootstrapping. Classifiers from distant supervision tend to be less precise than hand-written rules, though in some cases they achieve comparable performance. Using complex features, distant supervision effectively detects new rules.

Question

When searching for an entity e_{new} that has a given relationship r with a given entity e

A. We search for e_{new} that have a similar embedding vector to e
B. We search for e_{new} that have a similar embedding vector to e_{old} which has relationship r with e
C. We search for pairs (e_{new}, e) that have similar embedding to (e_{old}, e)
D. We search for pairs (e_{new}, e) that have similar embedding to (e_{old}, e) for e_{old} which has relationship r with e

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

In the model for relation embedding individual entities have no embedding, therefore Answers A and B are not applicable. As for Answer D, it is not required that the existing embedding pairs (e_{old}, e) have a confirmed relationship. However, we could also search for embeddings of relations r that are similar to the embedding (e_{new}, e).

If t has no Hypernym ..

A. It is a root concept
B. It cannot match c such as t and X
C. It is identical to the initial root concept
D. It is a basic concept

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer A

Hypernym means that the concept has no more general concept, therefore it is a root concept. It cannot be part of the pattern from Answer B, since c would be a hypernym of t. In the search for hypernyms different root concepts can be found that have no relation among each other, which excludes Answer C.

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

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

The score function is a dissimilarity function that goes to zero when the facts are correct.

Answer A is exactly the opposite of the behavior of the score functions. As for Answer B, the score function maps tripes into real numbers, not vectors. And finally SGD optimizes the loss function not the score function.

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 $f(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'})$

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

Since the score function for $f(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t)$ us already almost zero, the increase in margin can only be achieved by increasing the scores for the implausible triples, $f(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'})$.

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C is incorrect, since the degree of a node v has no influence of how it is updated by its neighbors.

For a similar reason also answer D is incorrect. The fact that a node has a known label does not influence how it is updated by the neighbors.

Answer B is not correct since neighbors have different influences, e.g. depending on their node degrees.

Answer A is correct, as the degree of neighbors determines their influence.

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

The probabilities take clearly into account the node degrees. For each node they also depend on whether a pre-existing label exist or not, thus depend on the pre-existing knowledge.

Question

In the proposed schema matching approach, classifiers are constructed

1. for one schema using as training data the universe of the database
2. for each class of one schema using as training data the features of the instances of the class
3. for each class of both schemas using as training data the universe of the database
4. for each class of both schemas using as training data the features of all instances of the universe of the database

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 4

The classifiers are trained using the features of instances of the universe (this excludes answers 1 and 3 already).

The classifiers are trained per class, for both schemas, using as training data the features of the whole universe, which serve as positive and negative examples of the members of the class.

Answers to quizzes

5.1 INDEXING FOR INFORMATION RETRIEVAL

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A posting indicates…

1. The frequency of a term in the vocabulary
2. The frequency of a term in a document
3. The occurrence of a term in a document
4. The list of terms occurring in a document

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 3

A posting corresponds to the occurrence (or location) of a term in a document. It may contain in its payload the frequency of occurrence of the term, so Answer 2 might be true in special cases, but not in general.

When indexing a document collection using an inverted file, the main space requirement is implied by ...

1. The access structure
2. The vocabulary
3. The index file
4. The postings file

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 4

The sizes of the access structure, the vocabulary and the index file all are proportional to the size of the vocabulary which follows for reald world datasets Heap's law and are therefore of order O(n^0.5). The postings file always has a size of order O(n).

Using a trie in index construction …

1. Helps to quickly find words that have been seen before
2. Helps to quickly decide whether a word has not seen before
3. Helps to maintain the lexicographic order of words seen in the documents
4. All of the above

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 4

When receiving the next word during the indexing process first the word is searched in the trie constructed so far. So the trie helps to decide whether the word is already known, and if it is known find the associated data. The true can also be used to derive the list of words in. the vocabulary in lexicographical order by traversing the trie. So no sorting of the vocabulary is required.

Maintaining the order of document identifiers for vocabulary construction when partitioning the document collection is important ...

1. in the index merging approach for single node machines
2. in the map-reduce approach for parallel clusters
3. in both
4. in neither of the two

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

In the index merging approach documents are traversed in order and the order is maintained when generating the postings for individual words, which is important for avoiding expensive sorting. In the map-reduce indexing approach postings are sent over the network to the reducer nodes in an arbitrary order, since the infrastructure is controlling the communication of the messages. Therefore, the reducer nodes have to reestablish the order of document identifiers.

When compressing the adjacency list of a given URL, a reference list

1. Is chosen from neighboring URLs that can be reached in a small number of hops
2. May contain URLs not occurring in the adjacency list of the given URL
3. Lists all URLs not contained in the adjacency list of given URL
4. All of the above

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

In fact, the reference list can contain URLs that are not contained in the page of the given URL. Answer 1 is wrong since the reference list is chosen from a window in the lexicographical ordering of URLs. Answer 3 does not make sense, as it says that the reference list contains all URLs, except those in the given page.

Which is true?

1. Exploiting locality with gap encoding may increase the size of (the representation of) an adjacency list
2. Exploiting similarity with reference lists may increase the size of (the representation of) an adjacency list
3. Both of the above is true
4. None of the above is true

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 1

It is possible that with a very unfortuntate distribution of integers the size of the representation might increase with gap encoding (though it is not straightforward to find such a counter-example).

The encoding using a reference list might naturally increase the size of the representation of the adjacency list, if the reference list is poorly chosen. However, in the practical implementation only reference lists will be chosen that lead to an improvement in storage size.

When applying Fagin's algorithm for a query with three different terms for finding the k top documents, the algorithm will scan ...

1. 2 different lists
2. 3 different lists
3. k different lists
4. it depends how many rounds are taken

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

For answering a query with three terms, the posting lists of the three terms need to be inspected, therefore 3 different lists have to be scanned.

With Fagin's algorithm, once k documents have been identified that occur in all of the lists ...

1. These are the top-k documents
2. The top-k documents are among the documents seen so far
3. The search has to continue in round-robin till the top-k documents are identified
4. Other documents have to be searched to complete the top-k list

©2022, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer 2

In Fagin's algorithm round-robin traversal of posting lists stops once k documents have been seen in all lists. At this point also other documents may have been seen. The top-k documents are among all the documents that have been seen, but not necessarily those that have been seen in all lists.

10 itemsets out of 100 contain item A, of which 5 also contain B. The rule A ➔ B has:

A. 5% support and 10% confidence
B. 10% support and 50% confidence
C. 5% support and 50% confidence
D. 10% support and 10% confidence

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

The number of itemsets containing both A and B is 5, therefore the support is 5%.

In 5 out of 10 cases the rule is correct, therefore the confidence is 50%.

10 itemsets out of 100 contain item A, of which 5 also contain B. The rule B ➔ A has:

A. unknown support and 50% confidence
B. unknown support and unknown confidence
C. 5% support and 50% confidence
D. 5% support and unknown confidence

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

As in the previous question the support is 5%.

Since we do not know the number of itemsets containing the item B, we cannot determing the confidence of the rule.

Given the frequent 2-itemsets {1,2}, {1,4}, {2,3} and {3,4}, how many 3-itemsets are generated and how many are pruned?

A. 2, 2

B. 1, 0

C. 1, 1

D. 2, 1

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

Only one itemset will be generated, {1,2,4} for the first shared prefix {1}. Since the itemset {2,4} is not frequent, it will be pruned.

After the join step, the number of k+1-itemsets ...

A. is equal to the number of frequent k-itemsets
B. can be equal, lower or higher than the number of frequent k-itemsets
C. is always higher than the number of frequent k-itemsets
D. is always lower than the number of frequent k-itemsets

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

We cannot make any statement on the cardinality of the candidate set generated in the join step.

Examples:

{1,2} {1,3} will generate 1 3-itemset, so the cardinality is lower.

{1,2} {1,3} {1,4} will generate 3 3-itemsets, so the cardinality is equal.

{1,2} {1,3} {1,4} {1, 5} will generate 6 3-itemsets, so the cardinality is higher.

If rule {A,B} ➔ {C} has confidence c_1 and rule {A} ➔ {C} has confidence c_2, then ...

A. $c_2 >= c_1$
B. $c_1 > c_2$ and $c_2 > c_1$ are both possible
C. $c_1 >= c_2$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer B

See example on next slide.

Example

$$c_1 = \frac{s(A,B,C)}{s(A,B)} = \frac{1}{3}$$

$$c_2 = \frac{s(A,C)}{s(A)} = \frac{1}{6} = 0 \Rightarrow c_1 > c_2$$

$$c_1 = \frac{s(A,B,C)}{s(A,B)} = \frac{1}{3}$$

$$c_2 = \frac{s(A,C)}{s(A)} = \frac{5}{10} = \frac{1}{2} \Rightarrow c_1 < c_2$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A false negative in sampling can only occur for itemsets with support smaller than …

A. the threshold s

B. p*s

C. p*m

D. None of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D (under reasonable assumptions)

A false negative occurs if an itemset has a support above the threshold s, but in the sample less then p*s transactions are selected that contain the itemset (respectively less than the lowered threshold).

For each of the options in answers A, B, C it is easy to construct an adversarial example in which this is the case. Actually the more difficult case is when the itemset has the larger support count, as fewer remaining transactions remain to select from. Therefore, we conly consider the maximal possible size.

For answer A consider an itemset with support count larger than s. If m > (s – p*s+1) + p*m, then we can select the sample such that it contains at most p*s – 1 of the occurrences of the itemset. This is equivalent to saying that m > s, i.e. the database should have larger size than the support threshold. We never explicitly stated that this should be the case but is an obvious condition to satisfy.

For answer B the itemset has a support strictly smaller than s, and therefore for the same reasoning as in answer A, we can produce the counter-example.

For answer C we can select the occurrences without including the itemset from the (1-p)*m remaining occurrences if m > (m – p*m + 1) + p*m, which is equivalent to m > m-1. Here lowering the threshold could in the extreme case avoid having a false negative.

In the first pass over the database of the FP Growth algorithm

A. Frequent itemsets are extracted
B. A tree structure is constructed
C. The frequency of items is computed
D. Prefixes among itemsets are determined

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer C

The algorithms proceeds in two steps, where the first step has two passes over the database. The first pass is to compute the frequency of items, and the second to construct the tree structure. The frequent itemsets are extracted in the second step.

The FP tree below is …

A. not valid, b is missing
B. not valid, since count at leaf level larger than 1
C. possible, with 2 transactions {a}
D. possible, with 2 transactions {a,c}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Answer D

Answer A can be excluded, since it is possible to have paths that are not including some elements. It simply means that the corresponding itemsets do not occur.

Answer B can be excluded since leaf counts can be larger than 1, of multiple itemsets corresponding to the path exist.

Answer C is not possible, since in the presence of two transactions with itemset {a} and the tree structure implying that there exist two transactions with itemset {a,c}, the minimal count of node a has to be 4.

Answer D is possible with three transactions {a}, {a,c}, {a,c}

Distributed Information Systems
An Overview

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Overview

1. What is an Information System?
2. Data Modelling
   - 2.1 Models
   - 2.2 Data
   - 2.3 Representation
3. Managing data and information
   - 3.1 Data Management
   - 3.2 Information Management
   - 3.3 Distributed Information Management
4. About the lecture

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1. WHAT IS AN INFORMATION SYSTEM?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information Systems – are everywhere

A day in a student's life

- IS academia: course registration, grades, …
- Moodle: course information, slides, …
- Bank account: payments, savings, ….
- Library system: literature
- Search engine: where to find food, …
- Facebook, TikTok: connecting to friends, news, …
- Google maps: finding your way
- Campus map: finding lecture hall
- Email: exchanging messages

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In this part of the lecture we would like to understand the basic concept of information system. We are surrounded today by information systems and everybody has an intuitive understanding what an information systems is and does (a system that is treating information).

A large organization such as EPFL is running dozens if not hundreds of information systems. With the advent of the Web and mobile computing and the resulting democratization of information technology and integration of information technology in every day's life a plethora of new information systems have been emerging. Increasingly, information systems are not only interacting with humans, but also are among each other, with sensors gathering data from the environment, algorithms making decisions and different systems exchanging their data. The recent trend of generating and analysing increasing amounts of data, the so-called Big Data, is even more demonstrating the growing importance of information systems. The following are some types of information systems that are common:

The classical information systems: Organizational databases, Business process management systems, Geographic information systems, Text retrieval systems, Business intelligence (data warehousing and mining systems)

More recent types of information systems: Social Networks, Question-Answering System, Recommender Engines, Bioinformatics systems (e.g., genome or protein sequence retrieval), Environmental monitoring systems (disaster warning, meteo website)

However, not every computing system is considered as an information system. Systems that are used for communication through different media (e.g., IP telephony), games or simulations are not widely considered as information systems. What is the distinctive feature of an information system? In the following we will attempt to provide a more precise characterization of the concept of information system as we understand the concept in the context of this course.

The Business Perspective

Jobs related to information systems

- Project Managers
- Chief Information Officers (CIO)
- Technical Writers
- System Analysts
- Requirements Analyst

Jobs related to computer science

- Computer Programmer
- Java Developer
- Database Administrator
- Software Engineer
- Network Engineer

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The notion of information systems is overloaded with many different meanings, depending on the context. From a business perspective, it relates typically to jobs that are concerned with the design of computing systems in an enterprise context. The notion can be distinguished in this context from activities more directly related to software and systems implementation and management.

The Data Perspective

Information systems:

Computer systems handling and interpreting **large amounts of data**

- Transaction data
- Documents
- Maps
- Social networks
- Sensor data

Not information systems:

Computer systems performing **lots of computation**

- Computational science and simulation
- Computer games
- Computer algebra (Matlab etc)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

From a more technical perspective we can interpret the notion of information systems as computing systems that process large amounts of data, i.e., the focus is on processing data, instead of performing large numbers of computations.

Ask Wikipedia

Information system

From Wikipedia, the free encyclopedia

This article has multiple issues. Please help improve it or discuss these issues on the talk page. *(Learn how and when to remove these template messages)* [hide]

- The **lead section of this article may need to be rewritten.** *(April 2017)*
- This article **contains weasel words: vague phrasing that often accompanies biased or unverifiable information.** *(April 2017)*

Information systems (IS) are formal, sociotechnical, organizational systems designed to collect, process, store, and distribute information.[1] In a sociotechnical perspective, information systems are composed by four components: task, people, structure (or roles), and technology.[2]

A **computer information system** is a system composed of people and computers that processes or interprets information.[3][4][5][6] The term is also sometimes used in more restricted senses to refer to only the software used to run a computerized database or to refer to only a computer system.

Information Systems is an academic study of systems with a specific reference to information and the complementary networks of hardware and software that people and organizations use to collect, filter, process, create and also distribute data. An emphasis is placed on an information system having a definitive boundary, users, processors, storage, inputs, outputs and the aforementioned communication networks.[7]

Any specific information system aims to support operations, management and decision-making.[8][9] An information system is the information and communication technology (ICT) that an organization uses, and also the way in which people interact with this technology in support of business processes.[10]

Some authors make a clear distinction between information systems, computer systems, and business processes. Information systems typically include an ICT component but are not purely concerned with ICT, focusing instead on the end use of information technology. Information systems are also different from business processes. Information systems help to control the performance of business processes.[11]

Alter[12][13] argues for advantages of viewing an information system as a special type of work system. A work system is a system in which humans or machines perform processes and activities using resources to produce specific products or services for customers. An information system is a work system whose activities are devoted to capturing, transmitting, storing, retrieving, manipulating and displaying information.[14]

As such, information systems inter-relate with data systems on the one hand and activity systems on the other. An information system is a form of communication system in which data represent and are processed as a form of social memory. An information system can also be considered a semi-formal language which supports human decision making and action.

Information systems are the primary focus of study for organizational informatics.[15]

2018

Information system

From Wikipedia, the free encyclopedia

An **information system (IS)** is a formal, sociotechnical, organizational system designed to collect, process, store, and distribute information.[1] In a sociotechnical perspective, information systems are composed by four components: task, people, structure (or roles), and technology.[2] Information systems can be defined as an integration of components for collection, storage and processing of data of which the data is used to provide information, contribute to knowledge as well as digital products that facilitate decision making.[3]

A **computer information system** is a system composed of people and computers that processes or interprets information.[4][5][6][7] The term is also sometimes used to simply refer to a computer system with software installed.

Information Systems is an academic study of systems with a specific reference to information and the complementary networks of hardware and software that people and organizations use to collect, filter, process, create and also distribute data. An emphasis is placed on an information system having a definitive boundary, users, processors, storage, inputs, outputs and the aforementioned communication networks.[8]

In many organizations, the department or unit responsible for information systems and data processing is known as "**Information Services**".[9][10][11][12]

Any specific information system aims to support operations, management and decision-making.[13][14] An information system is the information and communication technology (ICT) that an organization uses, and also the way in which people interact with this technology in support of business processes.[15]

Some authors make a clear distinction between information systems, computer systems, and business processes. Information systems typically include an ICT component but are not purely concerned with ICT, focusing instead on the end-use of information technology. Information systems are also different from business processes. Information systems help to control the performance of business processes.[16]

Alter[17][18] argues for advantages of viewing an information system as a special type of work system. A work system is a system in which humans or machines perform processes and activities using resources to produce specific products or services for customers. An information system is a work system whose activities are devoted to capturing, transmitting, storing, retrieving, manipulating and displaying information.[19]

As such, information systems inter-relate with data systems on the one hand and activity systems on the other.[20] An information system is a form of communication system in which data represent and are processed as a form of social memory. An information system can also be considered a semi-formal language which supports human decision making and action.

Information systems are the primary focus of study for organizational informatics.[21]

2021

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When inspecting the article in Wikipedia we can notice that the “business perspective” seems to be predominant.. However, it is interesting to note that recently the processing of data has been explicitly added in the characterization of the concept.

Ask ChatGPT

An information system (IS) is a collection of hardware, software, data, people, and procedures that work together to facilitate the gathering, processing, storing, and dissemination of information within an organization or a specific context. The primary purpose of an information system is to support decision-making, coordination, control, analysis, and visualization of information to help individuals, groups, and organizations achieve their goals effectively and efficiently.

Information systems can vary widely in scope and purpose, ranging from simple systems used for personal tasks like managing contacts or calendars to complex enterprise-wide systems used by large organizations for functions like customer relationship management, supply chain management, and financial reporting.

Key components of an information system include:

1. **Hardware:** The physical devices and equipment that process and store data, such as computers, servers, storage devices, networking equipment, and peripherals.
2. **Software:** The programs and applications that enable users to interact with and manipulate data. This includes operating systems, databases, communication software, and various applications tailored to specific tasks.
3. **Data:** Raw facts, figures, and symbols that are collected, processed, and stored by the system. Data can come in various forms, such as text, numbers, images, and multimedia.
4. **People:** The individuals who use, manage, develop, and maintain the information system. This includes end-users, IT professionals, managers, and other stakeholders who contribute to the system's functionality and success.
5. **Procedures:** The formalized methods, rules, and guidelines that dictate how the system is used, maintained, and controlled. Procedures ensure consistency, security, and proper functioning of the system.

Information systems can be categorized based on their scope and focus:

- **Personal Information Systems:** Used by individuals for personal tasks such as email, document management, and time tracking.
- **Enterprise Information Systems:** Used by organizations to support various business functions like sales, marketing, finance, and human resources. Examples include Customer Relationship Management (CRM) and Enterprise Resource Planning (ERP) systems.
- **Decision Support Systems:** Provide tools and information to assist in decision-making by analyzing data and generating insights.
- **Knowledge Management Systems:** Focus on capturing, storing, and disseminating an organization's collective knowledge and expertise.
- **Geographic Information Systems (GIS):** Specialized systems for capturing, managing, analyzing, and presenting geographic or spatial data.
- **Transaction Processing Systems (TPS):** Handle routine, day-to-day transactions within an organization, such as order processing and inventory management.

Overall, information systems play a crucial role in modern businesses and society by enabling efficient data management, improved communication, streamlined processes, and better decision-making.

(pretty similar to the WikiPedia article)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

ChatGPT relies in its answer largely on the article found in WikiPedia.

A Systems Perspective

Information Processing

This model applies to all types of systems: organisms, species, humans, organizations, societies, etc.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A systems perspective would rather focus on the question of how an information processing system is interacting with its environment. This is the perspective we will adopt in the following.

Information Processing: example

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We can describe this process by a simple example. Let us assume that a ship is in distress and sends the (famous) SOS signal to the coast guard.

Looking in more detail on what is happening, we can identify different aspects of this process.

First, we observe that the ship is sending a signal. This corresponds to the transmission of data, irrespective of how the data is transmitted over communication channel, e.g., by radio, Internet, Morse, fume signals etc.

The receiver of the data (the Morse code) has then the task of interpreting the data (or signal). Only this establishes what we would consider information. So, information is related to the understanding of data or giving meaning to the data. Without being able to perform this interpretation of the data at the receiver, sending the signal would be useless.

Finally, understanding the message does not mean that the receiver also knows what to do. The receiver also has some knowledge, i.e., that capability of reacting in a meaningful way with the environment based on the information received.

The process described in the example, applies to any entity that is performing information processing in the context of an environment, e.g., an organism, an organization, or a computing system. Performing information processing in the context of computational systems is what we will be studying in this lecture.

What is an Information System?

An information system is a **computing system** interacting with the environment with the capability of

1. managing **data,**
2. inferring **information** by understanding this data *by using a representation of a **model** of its **real-world environment***
3. for performing a given **task**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Based on the previous considerations, we can come up with a first attempt to define what an information system is. Note that the notion of environment in the definition can be interpreted widely. It can consist of humans interacting with the information systems, as well as the physical environment or other information systems.

Compare to the definition provided by WikiPedia, the definition looks quite similar:

In WikiPedia it is written that tnformation systems can be defined as an

- integration of components for collection, storage and processing of data (-> managing data)
- of which the data is used to provide information (-> inferring information),
- contribute to knowledge that facilitate decision making (-> performing a task).

2. DATA MODELING

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

2.1 What is a model?

Mathematical representation of a real-world phenomenon

Example: travel relationship

Relation : $travel \subseteq Person\ x\ Location\ x\ Location$

Axiom: $travel(p, l1, l2) \wedge travel(p, l2, l3) \rightarrow travel(p, l1, l3)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information systems use models to represent some aspect of their real-world environment. But what is a model? The answer is simple: a model is a (mathematical) structure. A mathematical structure consists of elements from a domain (also called constants in a mathematical language), functions and axioms. The elements provide identifiers for things (indeed everyTHING can receive a name – and a central task of information systems is to handle names, respectively identifiers, of real-world entities), the functions provide properties of those entities, and the axioms provide rules or constraints that state which properties are possible and not.

Some information systems support only very limited or specific kinds of models, others very generic models. Very often first-order logic is used as generic modelling approach for information systems. Models based on first order logic allow to represent the important entities, their relationships and properties in a generic approach. However, with the increasing need to process information for specific needs (e.g., processing text, images, sensor data), also more specific mathematical models are increasingly used, such as graphs, vector spaces, probabilistic models, differential equations, process models etc. These are some examples of formal models used in information systems:

- Entity-Relationship models have been among the earliest conceptual models used for information systems. They have been derived from knowledge representation mechanisms developed in AI.
- OWL: is a generalization of the entity relationship model enabling logical inference (for concept classes). It has become the basic model for the Semantic Web.
- Graph models: used for social network data, biological network data, communications network data
- Vector space models: used to represent feature spaces of text and media content
- Probabilistic models: used to represent uncertainty in content and sensor data
- Differential equations and simulation programs: used to represent behaviors of complex systems
- Process models: capture the structure and dynamics of business processes, also called workflows.

Data

Example: a Mathematical Model for trajectories:

- Domains: time T, space $S = Long \times Lat$
- trajectories Tr is a set of functions
- one trajectory $tr \in Tr$ is a partial function $tr: T \rightarrow S$

a sequence of GPS point (x,y,t)

Representing a single trajectory tr as data, a set of samples: $tr(t_0) = s_0, tr(t_1) = s_1, \ldots$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to illustrate the use of models in an information system, we will use a simple, yet practical, example, the management of trajectories. Here we describe a basic trajectory model in a 2-dimensional space using a function, as it is frequently used to describe, for example, vehicle trajectories recorded using a GPS.

Functions are a key constituent of information systems. They relate objects with their properties and different objects among each other. An interesting and central question is of how functions are represented. There exists two fundamentally different ways to do this: either by explicitly enumerating function values for given function arguments, or by providing a specification or algorithm to compute the function value from a given function argument. Both ways are in fact used in information systems.

In our trajectory example, we can use both methods. We could describe trajectory by (measured) samples, or by a function generating a trajectory. In the context of science, functions are often called quantitative or qualitative variables. Since many facts about the real-world cannot by specified by an algorithm (e.g. computing the birthday from the name of a person), the explicit representation of functions plays a particularly central role in information systems. Explicitly enumerated functions are commonly called **data**.

2.2 Data

How is a model represented in a computer system?

A mathematical model M is represented using a **data structure D**.

A data structure D is a discrete mathematical structure and their operations that can be **processed by a computer**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

So far we have considered models as abstract mathematical structures, consisting of domains, functions and axioms. For being able to handle a model in a computer system, we need to represent the model within the computer system. For that purpose we need a representation mechanism that can be “understood” and processed by a computer. In other words we need a mechanism that can be used to represent a model. Such a mechanism is called a **data model**. A data model consists of discrete mathematical structures and operations that a computer can represent and execute.

Relationship among Model and Data Model

The same function can be represented using different data structures

$tr: T \rightarrow S$

What you think

Dictionary

```
dict_tr = {}
dict_tr[0] = [0,0]
dict_tr[1] = [1,0]
dict_tr
```

```
{0: [0, 0], 1: [1, 0]}
```

List

```
list_tr = []
list_tr.append([0,0])
list_tr.append([1,0])
list_tr
```

```
[[0, 0], [1, 0]]
```

Relation

```
import pandas as pd
d = {0: [0, 0], 1: [1, 0]}
df = pd.DataFrame(data = d)
df
```

	0	1
0	0	1
1	0	0

What you program

Different physical implementations of the data types

What happens

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Data structures allow to represent the functions that are part of a model within the computer systems. The same function can be represented in different ways using different abstract data types and data structures. Some representations are more appropriate than others. E.g., using a hashtable enforces the constraint that a function can have only one function value, whereas using a list of tuples requires to enforce this constraint and typical operations in the code.

Three levels of modeling

Conceptual modelling

- mathematical model that user thinks in

Logical modelling

- mathematical model that computer can process

Physical modelling

- binary representation of logical model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We can identify three levels of modeling, when designing an information system: the conceptual, the logical and the physical level.

These three levels have been identified as a formal standard in the ANSI-SPARC architecture for abstract design standard for database systems in 1975.

Is this mapping trivial?

Not exactly. If the meaning is not precisely understood bad things may happen.

When NASA Lost a Spacecraft Due to a Metric Math Mistake

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Not considering properly units, has led already to some major catastrophies.

Three Notions of Data

Conceptual level (data as in **science**)

- Values a variable (function) can take

Logical level (structured data as in **programming**)

- Value (instance) of an abstract data type

Physical level (data as in **information theory**)

- Binary representation

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We can also relate the concept of “data” to these three modeling levels. This reveals the interesting fact that we are continuously using the term data with varying meaning (without necessarily being aware of that).

The first meaning corresponds to the notion of data as used in a scientific experiment, where data are values a function can take. These are also often called samples. This is also the meaning that is associated with data in the corresponding Wikipedia article.

A second meaning corresponds to the notion of data as being an instance of an abstract datatype processed in a computing system. This corresponds to the notion of data as it is being used, for example, in the context of database management systems.

Finally, when considering the physical representation of data as binary strings, data corresponds to the concept of data as being used in the context of information and coding theory (e.g, when investigating data compression)

Factoring in the design principles of data management systems we can provide now a refined view of a typical information system architecture. The semantic layer is concerned with the modeling of the real world, the conceptual modeloing. In order to implement the model the syntactic layer provides a data model in which the semantic model can be implement and that is supported by a generic software (typically a database management system) that supports a wide range of information systems. The physical layer deals with the problem of representing the constructs of the data model efficiently in the available computing infrastructure, using the typical mechanisms as provided by its operating system, such as access to storage and network resources.

2.3 Representation

A **representation** is a very general relationship that expresses similarities or equivalences between mathematical structures.

- Maps the domain of one structure into the other
- Maps functions and relations of one structure into the other
- Preserves some properties

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As we have seen, the term representation is widely used in the context of information systems, and more generally in mathematics. Though we have an intuitive understanding of the concept, it is worthwhile to investigate in some more detail, what exactly is meant by this concept. We give here a generally accepted definition of the concept representation in mathematics. It is a structure-preserving mapping, which means that it maps the domains as well as the structural elements, i.e., functions and properties. What the concept of preservation of properties means, requires some further explanation.

Example: Homomorphism

Let X and Y be two mathematical structures with the same functions

A **homomorphism** $H: X \rightarrow Y$ has to satisfy for every function f:

$$H\big(f(e_1, e_2, \dots)\big) = f(H(e_1), H(e_2), \dots)$$

This is one possible preserved property!

If H is in addition bijective, then H is an **isomorphism**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One type of representation that preserves properties in a mapping between two mathematical structures, are homomorphisms, a widely used concept in mathematics. Informally, homomorphisms preserve structural properties fully. If the homomorphisms are in addition bijective, we have isomorphism. We can think of isomorphism as a mapping between two mathematical structures that are equivalent.

Binary Representation

Data structure: `list(int)`

Example: `1,8,3, …`

Binary representation:

$R(n) = b_2 b_1 b_0$ if $n = b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0$

$R(n_1, n_2, \ldots) = R(n_1) R(n_2) \ldots$

Example: `001 100 010 …`

R is bijective, isomorphic!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Also, representations of data structures in binary form, are examples of isomorphic mappings.

Data Exchange Format

Data structure: `list(int)`

Example: `1,8,3, …`

JSON representation:

'[1, 8, 3,…]'

R is bijective, isomorphic!!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Similarly, also data exchange formats are examples of exact representations (fortunately so, otherwise we would lose data when exchanging it among computer systems).

Example: Representing a Continuous Model

Mathematical model: domains $T, S = Long \times Lat$

Trajectory: $tr: T \rightarrow S$

Representation of time domain in Python:

$R: T \rightarrow Float$

Almost a homomorphism!

```python
def timesteps(s, u, n):
    print(s)
    if n == 0:
        return s
    else:
        return timesteps(s + u, u, n-1)
```

```python
timesteps(0, 0.2, 10)
```

```
0
0.2
0.4
0.6000000000000001
0.8
1.0
1.2
1.4
1.5999999999999999
1.7999999999999998
1.9999999999999998
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

However, for non-discrete mathematical structures the situation is different, since we cannot map a continuous domain into a discrete domain without loss of information.

One typical illustration for the consequences of this, are anomalies that we encounter in numerical computation.

Representations are not always accurate

The representations need not always be homomorphisms

- They preserve some relations approximately
- Measures for the quality of approximation are needed

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Therefore, representations do not necessarily always perfectly preserve the original structure. This might be due to the impossibility of doing so (as in the case of mapping continuous domains into discrete domains), or on purpose. Often representations are used to simplify or approximate the original structures, to make them more amenable for processing.

One important case of such mappings used to approximate the original structure is found in so-called “representation learning”. There, complex discrete structures, e.g., text, are mapped into vector spaces in order to capture essential properties of the original data. We will study this approach in detail in this lecture.

Example

Representing an empirical mathematical model, by another mathematical model

- Assume we have only partial data for a trajectory tr (samples)
- We interpolate the missing values, e.g., linear interpolation

$$tr_{int}: T \rightarrow S$$

- If new data arrives, we can estimate the error
- The error measure characterizes the quality of the new representation tr_{int}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One possible reason for an approximative representation is that from the original structure data is missing. In this case the representation can use, for example, interpolation to replace the missing data values.

This examples shows a simple interpolation mechanism for trajectory data.

Representing the Real World

A model represents a reality, thus there exists a (hypothetical) **representation relationship**

Evaluating the model to characterize the quality of representation

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As we stated earlier, mathematical are used to represent some aspect of the real world. Therefore, we can also consider this as a form of representation. Obviously, this relationship can in general not be formally described.

3. MANAGING DATA AND INFORMATION

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

3.1 Data Management

The collection of data represented in a data model D is called a **database DB.**

A computer system that is designed to (generically) manage databases is called a **database management system DBMS**.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We already mentioned that the explicit representation of functions by enumeration, resulting in data, plays an important role in information systems. In the context of data management such a set of data is called a **database**. A computer system that is designed to manage databases is called a **database management system DBMS**. Thus, database management systems can be used to manage databases, and thus to implement information systems. Many information systems are implemented without relying on the use of a (generic) database management systems, but us a proprietary implementation of the the corresponding data management capabilities.

Data Management

Efficient management of large amounts of data

- efficient storage and indexing
- efficient search and aggregation

Ensuring **persistence** and **consistency** of data under updates and failures

- Persistence = data stored independent of lifetime of programs
- Consistency = data correct independent of type of failures

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When handling large amounts of data, we face two key challenges: first, the management should be performed efficiently. This concerns questions of how the data is mapped to the different available storage media, and what auxiliary data structures, so-called indices, are created to support the efficient access and questions of how operations on the data, e.g., for search and aggregation, are performed algorithmically in an efficient way. Second, the data needs to be kept safely. Different to data that is managed within the lifetime of a program, so-called transient data, data in information systems is maintained independently of the lifetime of any program. This property is called **persistence** of the data. Ensuring persistence and consistency of data, under all possible failures and when the data is stored in a variety of storage media us a non-trivial problem. The main purpose of data management systems is to provide performant implementations to support these tasks.

Examples of DBMS

Programming environment

- e.g. Python

Relational data management

- e.g. SQL, Pandas

Distributed data processing

- e.g. Map-Reduce, Spark

Text processing

- e.g. Elasticsearch, Lucene

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Whereas traditionally DBMS meant to designate a quite narrow class of systems, like relational database management systems, with the multiplication of information systems and associated infrastructures we may observe a multiplication of platforms that can be considered as database management systems.

Example: DataFrames

	tid	date	x	y
0	1	2008-02-02 15:36:08	116.51172	39.92123
1	1	2008-02-02 15:46:08	116.51135	39.93883
2	1	2008-02-02 15:46:08	116.51135	39.93883
3	1	2008-02-02 15:56:08	116.51627	39.91034
4	1	2008-02-02 16:06:08	116.47186	39.91248
...	...	...	...	...
583	1	2008-02-08 15:11:31	116.48347	39.91954
584	1	2008-02-08 15:21:31	116.50789	39.93128
585	1	2008-02-08 15:31:31	116.53174	39.91536
586	1	2008-02-08 15:41:31	116.57156	39.90263
587	1	2008-02-08 15:51:31	116.54723	39.90841

```
dt[dt['x'] < 116.5]
```

	tid	date	x	y
4	1	2008-02-02 16:06:08	116.47186	39.91248
5	1	2008-02-02 16:16:08	116.47217	39.92498
6	1	2008-02-02 16:26:08	116.47179	39.90718
7	1	2008-02-02 16:36:08	116.45617	39.90531
8	1	2008-02-02 17:00:24	116.47191	39.90577
...	...	...	...	...
579	1	2008-02-08 14:31:31	116.48503	39.91422
580	1	2008-02-08 14:41:32	116.44460	39.92156
581	1	2008-02-08 14:51:32	116.40047	39.92594
582	1	2008-02-08 15:01:31	116.44152	39.93236
583	1	2008-02-08 15:11:31	116.48347	39.91954

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the context of Python with the Pandas framework a programming environment is available, that provides some of the features found in relational DBMS, for example the capability to perform declarative queries on data tables.

There exist conventions to distinguish different levels of abstractions for data. One can think of these levels of abstraction as levels of providing "higher level meaning" to "raw data" that has been recorded or captured. These interpretations are generated either by automated or human analysis of the data, or a combination of the two.

This convention for classifying different types of data refers a priori to data in the sense of conceptual data. For each of the three levels, however, specific logical data models (e.g. graphs, relations) are preferably used.

Unstructured data is usually referred to as data that originates directly from some interaction with the real-world environment, be it through measurements (sensors, images, videos), human inputs (natural language, text, query logs), or machine input (system logs). The data is in fact structured (e.g., an image is an array of pixels, or a text is a sequence

of characters), but the structure is considered to carry little “semantic” meaning.

Structured data is data that is modelled based on a schema the represents different categories and relationships in the data. A standard example being the relational data model, where applications or users define tables with specific meaning. Structured data could be extracted from unstructured data, e.g., a table could contain the list of all objects recognized in an image.

Knowledge is also structured data and considered as a basis for decision making. Other properties associated with the notion of knowledge are the ability to reason with it, its ability to evolve dynamically including changing its schema and the derivation of new knowledge from existing knowledge through inference. Since the representation of knowledge is deemed to be more flexible, usually graph-based data models are used, that allow to represent in an unconstrained way new relationships.

Note: the notion of “knowledge management” is used in business typically with a different, though related, meaning. It refers to information systems that manage the knowledge of an organization, consisting, e.g., of its documents and information on the skills of its collaborators.

Example

Unstructured data: a GPS trace

a sequence of GPS point (x,y,t)

Bob's and Alice's GPS trace

Structured data: Road segments, Places

a sequence of episodes

Places that Bob and Alice visit

Knowledge: Concepts and Inferences

episodes with semantic annotations

Bob and Alice are frequently together in Ouchy, thus: *Bob loves Alice*

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This examples illustrates how such abstractions can look like in practice. We can consider GPS traces as raw, unstructured data. Unstructured expresses here the fact that apart from the physical location we have no further understanding of the meaning of the data. By using automated analysis (e.g. segmentation methods) and background knowledge (e.g. maps) one can extract information about places (e.g. where GPS coordinates did not change for a period) and roads / paths where movement is detecting. Aligning such structured data with maps can give an interpretation of where the person was and by what means it moved, which then constitutes knowledge. Inferences on such knowledge might then reveal relationships among persons and produce high level knowledge such as “Bob loves Alice”.

Model Building

Creating "higher level abstractions" from "lower level data"

- Using Statistical, Machine Learning methods, rule-based approaches, …
- Typically on large datasets
- Also called: data mining, data science, data analytics etc.

Create new models that represent the real world in a different way

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Deriving higher level abstractions or new interpretations of existing data is a central problem of information management. We will call this activity Model Building. Model building relies on numerous methods, both automated and with human intervention. We will introduce in this lecture many examples of such methods from the areas of data mining, information extraction and knowledge inference.

Model building is a central task in data science.

New Models generate New Data

Model

Data

Obtain data from measurement →

x	y	t
12	13	5:00
12	14	5:01
13	15	5:02

Data corresponding to the model →

place	x	y	start	end
p111	12	13	2:00	2:10
p112	13	15	2:25	2:30

Data corresponding to the model →

Subject	Relation	Object
home	ISA	Place
bus	ISA	Vehicle
Bob	ISAT	home

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Every new model, generates new data, by transforming the original data into data corresponding to the new model. Then for performing specific tasks functions can be evaluated on this news data, generating additional new data.

Information Management

Tasks
Search
Filtering
Classification
Prediction
Recommendation
Summarization
Integration
Question answering

model M

Model Usage (Inference): given a model, perform the task

Example:
Count the number of stops of a trip.

(conceptual) data

Model Building (Learning): given data, derive a model that supports the task

Example:
Given GPS traces, find places and road segments

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Since the central role of an information system is to create a model of the real-world environment based on data, the key information management tasks are related to the generation of models that derive from available data, new data that is suitable to support a task at hand.

Using then the model to perform the task is what we call Model Usage. Using a model consist of providing input to the model and to infer data that is then used for decision making.

Since there is no way to formally define or validate the relationship between a mathematical structure and the real world, methods are required that help to make plausible, that a model represents the real world correctly in some sense. With respect to physical phenomena it is indeed Science that endeavours to create models of reality (e.g. physical laws) using the scientific method. With respect to capturing human thought, expressed for example as written text, the field of information retrieval has established methods to verify whether computer models appropriately represent or emulate, the human reasoning processes. For traditional business information systems, it is the profession of business analysts that translate real world requirements and organizational structures into models for information systems.

These approaches always implement some form of feedback mechanism, in which the models that are developed are verified with respect to reality. E.g., business analysts present the models to potential users and refine them based on the feedback. Scientists verify their models with respect to experimental data and assume they are valid as long they are not falsified (as we know since Popper we can never proof that a model is correct!).

Quality vs. Cost Tradeoff

Computational cost of deriving models from data can be high

- Generally, better models have higher costs
- A trade-off

Example:

searching documents with keywords vs.
searching using a language model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Whereas information management is concerned with the issue of creating and using the best possible models of real world phenomena, in practice this cannot be done without ignoring the capabilities and costs of the underlying computing system. As a general rule, better models, i.e. models that more accurately and comprehensivly represent the world come at a higher cost. Therefore the gain in quality of presentation has to be outweighed with the computational, and hence energy cost incurred.

Utility

Users need information system to take **decisions**

Utility of information linked to the value achieved

Value depends on
- Importance of decision
- Quality of decision

Quality of decision depends on quality and understandability of information!

Using information systems for decision making is associated with the notion of **knowledge management**.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Finally we are coming back to the issue of the purpose of an information system. Information systems are needed to make informed decisions. Thus their reason to exist is to provide information helpful for decision making. The **utility of information** depends on the nature of the decision on the one hand, and on the quality of the decision that is possible based on the information on the other hand. Information systems can make such utility measures explicitly available in different forms. Systems for rating and recommendation, e.g. in social networks, are an example where quality of information is handled explicitly.

Considering the information management tasks mentioned before we can refine our view on information systems by several aspects. We can add an additional layer that corresponds to the users of the system, and that we call the pragmatic layer as users introduce the dimension of information utility. Information utility is addressed in areas such as data quality management, agent systems, social networks.

3.3 Distributed Information Management

Distribution can occur at different levels

- Network
- Data
- Models
- Control

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Centralized Information System

Centralized Information System on Computer Network

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Except in the very early days, information systems had always been used in computer networks. From the perspective of information management this has no significant implications, as long as the information system itself is centralized, i.e., running on one physical node under a single authority. The network just enables the access of a user to the information system from a remote location.

Physical Distribution

Use of distributed physical resources: locality of access, scalability, parallelism in the execution

Distributed Data Management

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

There exist however many reasons not to store all data of an information system in a single node of the network. Some of these reasons are related to the optimized use of available resources. We might want to move data in the network close to the node where it is accessed, we might want to take advantage of parallel processing of the data, and we might want to avoid bottlenecks in order to improve scalability of the system. All these are good reasons to distribute the data physically. However, physical distribution should be ideally fully transparent to the user. The user has still the impression of accessing a single information system that is running under a single authority. This model of distributed processing of data is the subject of **distributed data management**.

Key Issues in Distributed Data Management

Where to store data in the network?

- **Partitioning** of data
- **Replication** and **caching**
- Considering typical access patterns and data distributions

How to access data in the network?

- **Push** vs. **pull** access (query vs. filtering)
- Indexing of data in the network
- Distribution of queries and filters
- Considering the communication model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Distributed data management deals with similar questions as centralized data management, namely optimizing the storage of the data and the processing of accesses of the data. The new key element is that data can now be stored at different nodes in a network, and that the cost of data transmission over networks becomes an essential performance consideration. Since cost of data transmission is generally considered as expensive, in distributed data management often multiple copies of the same data are kept at different locations in order to speed up access or data is partitioned to bring the data closer to the application that use different parts of the data. This in turn implies new problems of keeping distributed data consistent while executing transactions that involve different nodes in the network. The area of distributed transaction management is dealing with this problem.

Logical Distribution

Use of different data models: **semantic heterogeneity**

- Independently developed information systems
- Different models for related concepts

Data integration

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Having a homogeneous view of a distributed information systems might not always be possible. If we want to access information in systems that have been developed by different entities, or if we want to integrate information from different information systems that have been independently developed, we can no longer assure that the information can be homogeneously accessed. The same information might be represented using different models and data structures, and the access methods might be different. In that case we are talking about **heterogeneous information systems**. The heterogeneity results from a distribution of the decision authority when designing the system. In order to overcome heterogeneity methods for integrating data and making information systems interoperable are needed.

Key Issues with Data Integration

The same real world aspect can be represented differently

- Requires agreement on the meaning of shared data
- Relating different models requires often human intervention
- human attention is a scarce resource!

Syntactic heterogeneity

- The same conceptual model can be represented using different logical data models (relational, XML, …)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Relating different models used in information systems to each other is solving the problem of **semantic heterogeneity**. This problem is as hard as creating new models for information systems and requires typically human intervention, thus the scarcest resource we have available.

Finally in heterogeneous information systems we must deal with the problem that the different information systems are under the control of different **autonomous** authorities. This poses problems of coordination, mutual trust, and privacy protection. The information systems can be considered as independent agents, that may pursue common goals, while protecting their own interests.

In a setting where different autonomous users exchange information, questions related to the utility of information come into play. As users have their own private interest, they have to consider them in interactions with other users. For example, when receiving information from another user, a fundamental question is whether the information can be trusted. It might be that the other user might have an interested to provide wrong information in order to obtain an advantage. When providing information to another user a different problem needs to be considered. Can we trust the other user to use the information properly, or will the information be used in unintended and potentially harmful ways. This is the privacy problem, and it is a fundamental challenge for an information society.

The two problems of information trust and privacy are related to each other. The more quality information we reveal the more trust we may expect to receive in return, but the more we also put our privacy at risk.

4. ABOUT THE LECTURE

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Focus of the Lecture

How do we **infer** and **share** information?

Distributed information systems:

- Inferring information from different data: text, records, graphs
- Natural language as a common model
- Inferring knowledge from shared data

► Foundations of Information Retrieval

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Today's information systems are inherently distributed, since they serve the sharing of information among humans (and increasingly machines). Therefore, the focus of the lecture is on the question of inferring and sharing of information. Most of the methods that we will cover in this course are related to the field of information retrieval in a general sense. With the rise of powerful language models text retrieval is considered a problem typically solved by such models. However, earlier methods remain relevant for two reasons: (1) understanding the basic mechanisms and (2) as more efficient alternatives in large scale data processing. Since we will mostly be studying fundamental methods from the field of information retrieval the course could also be called Foundations of Information Retrieval.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Part 1: Information Retrieval

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1. Information Retrieval - Overview

1.1 Introduction to Information Retrieval
1.2 Basic Information Retrieval
- 1.2.1 Text-Based Information Retrieval
- 1.2.2 Boolean Retrieval
- 1.2.3 Vector Space Retrieval
- 1.2.4 Evaluating Information Retrieval
  - 1.2.4.1 Precision-Recall
  - 1.2.4.2 Ranked Retrieval
- 1.2.5 Probabilistic Information Retrieval
- 1.2.6 Query Expansion
  - 1.2.6.1 User Relevance Feedback
  - 1.2.6.2 Global Query Expansion

1.3 Embedding Techniques
- 1.3.1 Latent Semantic Indexing
- 1.3.2 Latent Dirichlet Allocation
- 1.3.3 Word Embeddings – skipgram, CBOW
- 1.3.4 Fasttext
- 1.3.5 Glove

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.1 INTRODUCTION TO INFORMATION RETRIEVAL

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

What is Information Retrieval?

Information retrieval (IR) is the task of finding in a large collection of documents those that satisfy the information needs of a user

Examples

- Searching documents in a library
- Searching the Web

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information retrieval deals with the problem of matching information needs of human users with information provided in large document collections. Important aspects of this definition are:

- Documents are largely unstructured data; documents can be based on different media, including text, images, videos
- Document collections are generally large, with the Web being a prime example of a very large document collection
- Information needs are user and context dependent; there exists no formal (mathematical) definition of the information retrieval task

Different Types of Information Retrieval

Documents D can be

- unstructured data like text, images, audio, multimedia, genomic data, but also geographic data, chemical data, software code, …

Queries Q can be both structured and unstructured

- Boolean expressions
- Free text, sample documents

Results R can be sorted or unsorted

- Results sets
- Ranked lists

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information retrieval addresses search in many types of (mostly unstructured) data. The queries can be based on formal languages for retrieval, on unstructured queries, such as text queries, are sample documents that represent the search interest. Results can be both ranked or not ranked.

An information retrieval system must deal with the following tasks:

• Generating structured representations of information items: this process is called **feature extraction** and can include simple tasks, such as extracting words from a text as well as complex methods, e.g., for image or video analysis.

• Generating structured representations of information needs: often this task is solved by providing users with a query language and leave the formulation of structured queries to them. This is the case, for example, for simple keyword-based query languages, as used in Web search engines. Some information retrieval systems also support the user in the **query formulation,** e.g., through visual interfaces.

• Matching of information needs with information items: this is the algorithmic task of computing similarity of information items and retrieval queries. At the heart of this step is the **information retrieval model**. Similarity measures on the structured representations of queries and documents are used to model **relevance** of information for users. As a result, a selection of relevant information items or a ranked result can be presented to the user.

Since information retrieval systems deal usually with large information collections and/or large user communities, the **efficiency** of the implementation of an information retrieval system is crucial. This imposes fundamental constraints on the retrieval model. Retrieval models that would capture relevance very well, but are computationally prohibitively expensive, are not suitable for an information retrieval system.

Example: Text Retrieval

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The most popular information retrieval systems today are Web search engines. To a large degree, they are text retrieval systems, since they exploit mainly the textual content of Web documents for retrieval. However, current Web search engines also exploit many other features of documents, such as link information or image content, and personal features of the users. The three tasks of a Web search engine for retrieval are:

1. extracting the textual features, which are the words or terms that occur in the documents. We assume that the web search engine has already collected the documents from the Web using a Web crawler.
2. support the formulation of textual queries. This is usually done by allowing the entry of keywords through Web forms.
3. computing the similarity of documents with the query and producing from that a ranked result. Standard methods for computing similarity Boolean retrieval and vector space retrieval. We will introduce these methods in detail later in this lecture.

Formally IR: $P(D) \times Q \rightarrow R$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we provide a more formal description of how information retrieval models work in general. A first important observation is that the result depends not only on the properties of the query, but also on (statistical) properties of the document collection C, a subset of all possible documents D. So, formally, the information retrieval task can be described as the computation of a function IR: $P(D) \times Q \rightarrow R$ that for a given collection of documents $C \subseteq D$ and a query $q \in Q$ computes a result $r \in R$. In information retrieval features are generated from the query Q, each document d and the document collection C. This is represented by the feature extraction functions F_c, F_D and F_Q that produce a representation for each document and query, as well as features that are related to the collection as a whole ($F_C(C)$). Those representations are used to compute the similarity between a document and query for each document individually. The collection features need to be extracted only once and stay the same for all similarity computations. This reduces overall the complexity of the similarity computation.

Retrieval Model

The retrieval model determines

- the structure of the document representation Rep_D
- the structure of the query representation Rep_Q
- the similarity matching function sim

Relevance

- determined by the similarity matching function
- should reflect right topic, user needs, authority, recency
- no objective measure

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The heart of an information retrieval system is its retrieval model. The retrieval model is used to capture the meaning of documents and queries and determines from that the relevance of documents with respect to queries. Although there exist different intuitive notions of what determines relevance one must keep clearly in mind that it is not an objective measure and highly context-dependent.

What does the Similarity Function compute?

Two basic models

1. Boolean Retrieval: $V = \{0,1\}$
2. Ranked Retrieval: $V = [0,1]$ or $V = \mathbb{N}$

General Wisdom

- Boolean retrieval suitable for
  - Experts: can formulate accurate queries
  - Machines: can consume large results
- Ranked Retrieval good for ordinary users

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

There exist two basic types of results an information retrieval can produce. In Boolean retrieval the system returns a set of results. The main issue with this approach is that it is in general very difficult to formulate a query that produces a reasonably sized result set, that is not too large and not too small or empty. This is called sometimes the “feast-or-famine” problem. In contrast, ranked retrieval allows to to produce a list of result that is sorted by relevance. We may distinguish further whether scores are returned ($V = [0,1]$) which allows to assess the relative relevance of different results based on the scores, or whether simply a ranked list is returned ($V = \mathbb{N}$). The most common models return score values.

Information Filtering

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information retrieval is also used to describe other tasks than document search. For example, the roles of documents and queries can be swapped in an information retrieval system. As a result, one obtains an information filtering system. Information filtering systems can be based on the same retrieval models as standard information retrieval systems for ad-hoc query access.

Information Retrieval and Browsing

Retrieval

- Produce a ranked result from a user request
- Interpretation of the information by the system

Browsing

- Let the user navigate in the information set
- Relevance feedback by the human

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information retrieval is usually closely connected to the task of browsing. Browsing is the explorative access by users to large document collections. By browsing a user implicitly specifies his/her information needs through selection of documents. This feedback can be used by an information retrieval system in order to improve its query representation and thus the retrieval result. One example of such an approach we will see when introducing relevance feedback. On the other hand, results returned by information retrieval systems are usually large, and therefore browsing is needed by users in order to explore the results. Both activities, retrieval and browsing thus can be combined into an iterative process. We will present methods for obtaining relevance feedback in the following.

Other tasks

In a more general sense Information Retrieval is used for a number of different types of tasks, such as

- Information filtering
- Document summarization
- Question answering
- Recommendation
- Document classification

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

There are a number of other types of tasks performed on large document collections that are associated with the field of information retrieval.

IR is an Information Management Task

Model Usage (Inference): given a model, perform the task

Retrieve result documents for a query

Model Building (Learning): given documents, find a model

In IR the model is often based on simple statistics on the data

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information retrieval is a classical information management tasks. It implies model building, by creating representations of documents and queries, and model usage by performing efficient retrieval of search results from large document collections.

1.2 BASIC INFORMATION RETRIEVAL

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.2.1 Text-based Information Retrieval

Most of the information needs and content are expressed in natural language

- Library and document management systems
- Web Search Engines

Basic approach: use the words that occur in a text as *features* for the interpretation of the content

- This is called the "full text" or "bag of words" retrieval approach
- Ignore grammar, meaning etc.
- Simplification that has proven successful
- Document structure, layout and metadata may be considered additionally (e.g., PageRank/Google)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Traditionally, information retrieval has been concerned with the problem of retrieving information from large bodies of documents with mostly textual content, as they where typically found in library and document management systems. The problems addressed were classification and categorization of documents, systems and languages for retrieval, user interfaces and visualization. The area was perceived as being one of narrow interest for a highly specialized user community, mainly librarians. The advent of the WWW changed this perception completely, as the web is a universal collection of documents with universal access.

Since nowadays a large quantity of the document content is still available in textual form, text retrieval is an important area of information retrieval. Natural language text carries a lot of meaning which is still hard to capture fully, and which has not necessarily to be captured fully to perform an information retrieval task. Therefore, information retrieval systems are based on strongly simplified models of text, ignoring most of the grammatical structure of text and reducing texts essentially to the terms they contain. This approach is called full text retrieval and is a simplification that has proven to be very successful.

Architecture of Text Retrieval Systems

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This figure shows the basic architecture of a text retrieval system, with its different functional components. We can distinguish three main components:

1. the feature extraction component: it performs text processing to transform queries and text documents into a keyword-based representation
2. the ranking system: it implements the retrieval model. In a first step user queries are potentially modified (if user relevance feedback is used), then the documents required for computing the result are retrieved from the database and finally similarity values are computed according to the retrieval model in order to compute the ranked result.
3. the data access system: it supports the ranking system by efficiently retrieving documents containing specific keywords from large document collections. A standard technique to implement this component is **inverted files**.

In addition, we have two interfaces, one with the user for query input and result output, and with a data management systems for handling the text database.

Pre-Processing Text for Text Retrieval

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In full text retrieval each document is represented by a set of representative keywords or index terms. An index term is a document word useful for capturing the document's main topics. Often, index terms are only nouns, because nouns carry meaning by themselves, whereas verbs express relationships between words. These relationships are more difficult to extract.

When using words as text features, the standard sequence of preprocessing steps is the following: first, the document structure, e.g., from XML OR HTML, is extracted and if required stored for further processing. The remaining text is stripped of special characters, producing the full text of the document as a sequence of tokens. Then very frequent words which are not useful for retrieval, so-called "stopwords", are eliminated (e.g., in English words such as "a", "and" etc.). As the same word can occur in natural language in different forms, usually stemming is used: Stemming eliminates grammatical variations of the same word by reducing it to a word root, e.g., the words connecting, connection, connections would be reduced to the same "stem" connect. This step can be followed by a manual intervention, where human curators can select or add index terms based on their understanding of the semantics of the document. The result of the process is a set of index terms which represents the document.

Text Retrieval - Basic Concepts and Notations

Document d:	expresses ideas about some topic in a natural language
Query q:	expresses an information need for documents pertaining to some topic
Index term:	a semantic unit, a word, short phrase, or potentially root of a word
Database DB:	collection of n documents $d_j \in DB$, $j=1,\ldots,n$
Vocabulary T:	collection of m index terms $k_i \in T$, $i=1,\ldots,m$

A document is represented by a set of index terms k_i

Rep_D: The importance of an index term k_i for the meaning of a document d_j is represented by a *weight* $w_{ij} \in [0,1]$; we write $d_j = (w_{1j}, \ldots, w_{mj})$

sim: The IR system assigns a *similarity coefficient* $sim(q, d_j)$ as an estimate for the relevance of a document $d_j \in DB$ for a query q.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We introduce the terminology we will use in the following for describing text retrieval systems. Note that the method of how specific weights are assigned to an index term with respect to a document and of how similarity coefficients are computed are part of the specification of a text retrieval model.

Example: Documents

B1 A Course on Integral Equations
B2 Attractors for Semigroups and Evolution Equations
B3 Automatic Differentiation of Algorithms: Theory, Implementation, and Application
B4 Geometrical Aspects of Partial Differential Equations
B5 Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra
B6 Introduction to Hamiltonian Dynamical Systems and the N-Body Problem
B7 Knapsack Problems: Algorithms and Computer Implementations
B8 Methods of Solving Singular Systems of Ordinary Differential Equations
B9 Nonlinear Systems
B10 Ordinary Differential Equations
B11 Oscillation Theory for Neutral Differential Equations with Delay
B12 Oscillation Theory of Delay Differential Equations
B13 Pseudodifferential Operators and Nonlinear Partial Differential Equations
B14 Sinc Methods for Quadrature and Differential Equations
B15 Stability of Stochastic Differential Equations with Respect to Semi-Martingales
B16 The Boundary Integral Approach to Static and Dynamic Contact Problems
B17 The Double Mellin-Barnes Type Integrals and Their Applications to Convolution Theory

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an example of a (simple) document collection that we will use in the following as running example.

Term-Document Matrix

Matrix of weights w_{ij}

Terms	Documents																
	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17
algorithms	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
application	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
delay	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
differential	0	0	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0
equations	1	1	0	1	0	0	0	1	0	1	1	1	1	1	1	0	0
implementation	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
integral	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
introduction	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
methods	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
nonlinear	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
ordinary	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
oscillation	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
partial	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
problem	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0
systems	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0
theory	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	1

This example

- Vocabulary (contains only terms that occur multiple times, no stop words)
- all weights are set to 1 (equal importance)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In text retrieval we represent the relationship between the index terms and the documents in a term-document matrix. In this example only a selected vocabulary is used for retrieval, consisting of all index terms that occur in more than one document and only weights of 1 are assigned, indicating that the term occurs in the document.

Implementation in Python

```
titles
```

```
['A Course on Integral Equations',
 'Attractors for Semigroups and Evolution Equations',
 'Automatic Differentiation of Algorithms: Theory, Implementation, and Application',
 'Geometrical Aspects of Partial Differential Equations',
 'Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra',
 'Introduction to Hamiltonian Dynamical Systems and the N-Body Problem',
 'Knapsack Problems: Algorithms and Computer Implementation',
 'Methods of Solving Singular Systems of Ordinary Differential Equations',
 'Nonlinear Systems',
 'Ordinary Differential Equations',
 'Oscillation Theory for Neutral Differential Equations with Delay',
 'Oscillation Theory of Delay Differential Equations',
 'Pseudodifferential Operators and Nonlinear Partial Differential Equations',
 'Sinc Methods for Quadrature and Differential Equations',
 'Stability of Stochastic Differential Equations with Respect to Semi-Martingales',
 'The Boundary Integral Approach to Static and Dynamic Contact Problems',
 'The Double Mellin-Barnes Type Integrals and Their Application to Convolution Theory']
```

```
tf = CountVectorizer(analyzer='word', ngram_range=(1,1), min_df = 2, stop_words = 'english')
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Python provides the main functions that we will introduce formally, in the form of libraries. So in most cases a direct implementation of algorithms is no more required.

We will illustrate of how some of the core concepts we introduce, are represented in Python.

Implementation in Python

```
from sklearn.feature_extraction.text import CountVectorizer

tf = CountVectorizer(analyzer='word', ngram_range=(1,1), min_df = 2, stop_words = 'english')
features = tf.fit_transform(titles)
features.todense()
```

```
matrix([[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
        [1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0],
        [1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
        [0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1],
        [0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0],
        [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]])
```

```
vocabulary = list(tf.vocabulary_.keys())
vocabulary.sort()
print(vocabulary)
```

```
['algorithms', 'application', 'delay', 'differential', 'equations', 'implementation', 'integral', 'introduction', 'me
thods', 'nonlinear', 'ordinary', 'oscillation', 'partial', 'problems', 'systems', 'theory']
```

A retrieval model attempts to capture …

1. the interface by which a user is accessing information
2. the importance a user gives to a piece of information for a query
3. the formal correctness of a query formulation by user
4. the structure by which a document is organised

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Full-text retrieval refers to the fact that …

1. the document text is grammatically fully analyzed for indexing
2. queries can be formulated as texts
3. all words of a text are considered as potential index terms
4. grammatical variations of a word are considered as the same index terms

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The entries of a term-document matrix indicate …

1. how many relevant terms a document contains
2. how frequent a term is in a given document
3. how relevant a term is for a given document
4. which terms occur in a document collection

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.2.2 Boolean Retrieval

Users specify which terms should be present in the documents

- Simple, based on set-theory, precise meaning
- Frequently used in (old) library systems
- Still many applications, e.g., web harvesting

Example query

- "application" AND "theory" → answer: B3, B17

Retrieval Language Q

expr ::= term | (expr) | NOT expr | expr AND expr | expr OR expr

Weights for index terms appearing in documents

$w_{ij} = 1$ if $k_i \in d_j$ and 0 otherwise

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Early information retrieval systems (as well as many search systems in use today, such as integrated search tools in operating systems, like spotlight and windows search) use the Boolean retrieval model. This model is in fact comparable to database querying, as requests are specified as first order logical expressions. Term weights are set to 1 when a term occurs in a document, as shown in the term-document matrix previously.

"Similarity" Computation in Boolean Retrieval

Step 1:
Determine the disjunctive normal form of the query *q*

- A disjunction of conjunctions
- Using distributivity and Morgans laws, e.g. *NOT (s AND t)* ≡ *NOT s OR NOT t*
- Thus $q = ct_1$ *OR … OR* ct_l where $ct = \underline{t}_1$ *AND … AND* $\underline{t}_k$ and $\underline{t} \in \{t, NOT\ t\}$

Step 2:
Rep_Q: For each conjunctive term *ct* create its query weight vector *vec(ct)*

- $vec(ct)=(w_1,\ldots,w_m)$:
  $w_i = 1$ if k_i occurs in *ct*
  $w_i = -1$ if *NOT* k_i occurs in *ct*
  $w_i = 0$ otherwise

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Computing the similarity of a document and a query reduces in Boolean retrieval to the problem of checking whether the term occurrences in the document satisfy the logical condition specified by the query. In order to do so in a systematic manner, a Boolean query is first normalized into disjunctive normal form. Using this equivalent representation, checking whether a document matches the query reduces to the problem of checking whether the document vector, i.e., the column of the term-document matrix corresponding to the document, matches one of the conjunctive terms of the query.

"Similarity" Computation in Boolean Retrieval

Step 3:
If one weight vector of a conjunctive term *ct* in *q* matches the document weight vector $d_j = (w_{1j}, \ldots, w_{mj})$ of a document d_j, then the document d_j is relevant, i.e.,

$sim(d_j, q) = 1$

- *vec(ct)* matches d_j if:

  $w_i = 1 \wedge w_{ij} = 1$

  $w_i = -1 \wedge w_{ij} = 0$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A match is established if the document vector contains all the terms of the query vector in the correct way, i.e.,

- if the term occurs in the positive form in the query the term must occur in the document
- if the term occurs in the negated form in the query the term must not occur

If the term does not occur in the query, it may or may not occur in the document.

Example

Index terms *{application, algorithm, theory}*

Query "*application" AND ("algorithm" OR NOT "theory")*

Disjunctive normal form of query

("application" AND "algorithm" AND "theory") OR

("application" AND "algorithm" AND NOT "theory") OR

("application" AND NOT "algorithm" AND NOT "theory")

Query weight vectors $q=\{(1,1,1), (1,1,-1), (1,-1,-1)\}$

Documents $d_1=\{algorithm, theory, application\}$ $(1,1,1)$

$d_2=\{algorithm, theory\}$ $(0,1,1)$

$d_3=\{application, algorithm\}$ $(1,1,0)$

Result $sim(d_1, q) = sim(d_3, q) = 1, sim(d_2, q) = 0$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates a complete Boolean retrieval process for our sample document collection.

The transformation from the original query to the disjunctive normal form proceeds in the following steps:

application AND (algorithm OR NOT theory) ->

(application AND algorithm) OR
(application AND NOT theory)->

(application AND algorithm AND (theory OR NOT theory) OR
(application AND (algorithm or NOT algorithm) AND NOT theory) ->

(application AND algorithm AND theory) OR
(application AND algorithm AND NOT theory) OR
(application AND algorithm AND NOT theory) OR
(application AND NOT algorithm AND NOT theory) ->

(application AND algorithm AND theory) OR
(application AND algorithm AND NOT theory) OR
(application AND NOT algorithm AND NOT theory)

1.2.3 Vector Space Retrieval

Limitations of Boolean Retrieval

- No ranking: problems with handling large result sets
- Queries are difficult to formulate
- No tolerance for errors
- Queries either return far too many results, or none

Key Idea of Vector Space Retrieval

- Use “free text” queries
- represent both the document and the query by a weight vector in the m-dimensional keyword space assigning non-binary weights
- determine their distance in the m-dimensional keyword space

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The main limitation of the Boolean retrieval model is its incapability to rank the result and to match documents that do not contain all the keywords of the query. More complex requests become very difficult to formulate. Finally, for the user it is hard to predict whether a query would produce a reasonably sized set of results. Often either no results are returned, if the query is too restrictive or very large numbers of results are produced in the opposite case (this phenomenon is often called “feast or famine”).

The vector space retrieval model addresses these issues by supporting non-binary weights, i.e., real numbers in [0,1], both for documents and queries, and producing continuous similarity measures in [0,1]. The similarity measure is derived from the geometrical relationship of vectors in the m-dimensional space of document/query vectors. By using free text queries, i.e., users can use any text as query, the vector space model also simplifies the task of query formulation for users.

Similarity Computation in Vector Space Retrieval

$$\vec{d}_j = \left(w_{1j}, w_{2j}, \ldots, w_{mj}\right), w_{ij} > 0 \quad if\ k_i \in d_j$$

$$\vec{q} = \left(w_{1q}, w_{2q}, \ldots, w_{mq}\right), w_{iq} \geq 0$$

$$sim(\vec{q}, \vec{d}_j) = \cos(\theta) = \frac{\vec{d}_j \bullet \vec{q}}{\left|\vec{d}_j\right|\left|\vec{q}\right|} = \frac{\sum_{i=1}^{m} w_{ij} w_{iq}}{\left|\vec{d}_j\right|\left|\vec{q}\right|}$$

$$|v| = \sqrt{\sum_{i=1}^{m} v_i^2}$$

Since $w_{ij} > 0$ and $w_{iq} \geq 0$, $0 \leq sim(q, d_j) \leq 1$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For information retrieval, the distance measure for vectors must satisfy the following properties:

• If two vectors coincide completely their similarity should be maximal, i.e., equal to 1.

• If two vectors have no keywords in common, i.e., if wherever the query vector has positive weights the document vector has weight 0, and vice versa – or in other words if the vectors are orthogonal – the similarity should be minimal, i.e., equal to 0.

• in all other cases the similarity should be between 0 and 1.

The scalar product (which is equivalent to the cosine of the angle of two vectors) has exactly these properties and is therefore (normally) used as similarity measure for vector space retrieval.

A good question is why not use Euclidean distance, instead of cosine distance. The problem with using Euclidean distance is the following: different documents with the same or similar distribution of term weights, but of different vector length (for example, because the documents have different length) would give very different results, whereas their meaning would be the same or similar. For example, documents with vectors (1,1,0) and (2,2,0) would probably have the same meaning. But their Euclidean distances to a query vector would be very different (for illustration, consider the query vector (1,1,0)).

Vector Space Retrieval - Properties

Properties

- Ranking of documents according to similarity value
- Documents can be retrieved even if they don't contain some query keyword

Today's standard text retrieval technique

- Web Search Engines
- The vector model is the basis of most search engines, however they do not rely on it exclusively
- It is simple and fast to compute

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The vector space retrieval model is the standard retrieval technique used both on the Web and for classical text retrieval.

Example

Issue: how to determine the weights for q and d_j?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For vector space retrieval we need to determine of how weights are computed. If we apply the weighting scheme for the document and query vectors we have used for Boolean retrieval, we obtain the results vector space retrieval as shown in the figure. We observe that documents containing only one of the two keywords occurring in the query, can show up in the result, though with a lower similarity value.

Giving the same weight to all occurrences of terms does not account for the fact that different terms might have different importance for the meaning of a document. This was one of the reasons why in preprocessing the stopwords have been removed, as they do not contribute to the specific meaning of the document.

Term Frequency

Documents are similar if they contain the same keywords (frequently)

- Therefore, use the frequency *freq(i,j)* of the keyword k_i in the document d_j to determine the weight of the document vectors

(Normalized) term frequency of term k_i in Document d_j

$$tf(i,j) = \frac{freq(i,j)}{\max_{k \in T} freq(k,j)}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

An obvious difference that can be made among terms is with respect to their frequency of occurrence in a document. Thus, a weighting scheme for documents can be defined by considering the (relative) frequency of terms within a document. The term frequency is normalized with respect to the maximal frequency of all terms occurring within the document.

Example

$$sim(\vec{q},\vec{d}_j) = \frac{\sum_{i=1}^{m} w_{ij} w_{iq}}{\left|\vec{d}_j\right|\left|\vec{q}\right|}$$

Vocabulary T = {information, retrieval, agency}
Query q = (information, retrieval) = (1,1,0)

information …
…retrieval …
…information…
…retrieval …

D1 = (1, 1, 0)
sim(q,D1)=1

"Result"

retrieval …
…retrieval …
…retrieval …
…retrieval …

D2 = (0,1,0)
sim(q,D2)=0.7071…

agency …
…information…
retrieval…
…agency …

D3 = (0.5,0.5, 1)
sim(q,D3)=0.5773…

retrieval …
…agency …
…retrieval…
…agency…

D4 = (0, 1, 1)
sim(q,D4)=0.5

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates the use of term frequency. Assume we form the query vector by simply setting the weight to 1 if the keyword appears in the query. Then we would obtain D1 and D2 as result. This result appears to be non-intuitive, since we would expect that D3 is much more similar to q than D2. What has gone wrong?

The problem is that the term "retrieval", since it occurs very frequently in D2, leads to a high similarity value for D2. On the other hand, the term retrieval has very little power to disambiguate meaning in this document collection, since every document contains this term. From an information-theoretic perspective one can state, that the term "retrieval" does not reduce the uncertainty about the result at all.

Inverse Document Frequency

We have not only to consider how frequent a term occurs within a document (measure for similarity), but also how frequent a term is in the document collection of size n (measure for distinctiveness)

Inverse document frequency of term k_i

$$idf(i) = \log(\frac{n}{n_i}) \in [0, \log(n)]$$

n_i number of documents in which term k_i occurs

Inverse document frequency can be interpreted as the amount of information associated with the term k_i

Term weight (tf-idf) $w_{ij} = tf(i,j)\, idf(i)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Therefore, we should consider not only the frequency of a term within a document, when determining the importance of the term for characterizing the document, but also the discriminative power of the term with respect to the whole document collection. For that purpose, the inverse document frequency is computed and included into the term weight as factor.

With this weighting scheme we notice that eliminating stop words is in fact an optimization of computing similarity measures in vector space retrieval. Since stop words normally occur in every document of a collection, their term weights will normally be 0 and thus these terms will not receive any weight. Therefore, it makes sense to exclude them already on the preprocessing of documents.

$$idf(i) = \log(\frac{n}{n_i}) \in [0, \log(n)]$$

Example

Vocabulary T = {information, retrieval, agency}
Query q = (information, retrieval) = (1,1,0)

idf(information)=idf(agency)=log(2)
idf(retrieval)=log(1)=0

information …
…retrieval …
…information…
…retrieval …

D1 = (log(2), 0, 0)
sim(q,D1)=0.7071…

retrieval …
…retrieval …
…retrieval …
…retrieval …

D2 = (0,0,0)
sim(q,D2)=0

"Result"

agency …
…information…
retrieval…
…agency …

D3 = (0.5 log(2),0, log(2))
sim(q,D3)=0.316…

retrieval …
…agency …
…retrieval…
…agency…

D4 = (0,0, log(2))
sim(q,D4)=0

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With the tf-idf weighting scheme we compute different weights.

We have now: n=4, $n_{information}$=2, $n_{retrieval}$=4, n_{agency}=2

The result corresponds much better to our expectation.

Query Weights

The same considerations as for document term weights apply also to query term weights

Query weight for query q

$$w_{iq} = \frac{freq(i,q)}{\max_{k \in T} freq(k,q)} \log(\frac{n}{n_i})$$

Example: Query q = (information, retrieval)

- Query vector: (log(2), 0, 0)
- Scores: sim(q, D1)= 1
  sim(q, D2)=0
  sim(q, D3)=0.44…
  sim(q, D4)=0

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Finally, we need to determine the weights for the query vector. One approach is to apply the same method as for determining the weights of the document vector, considering the query as a document. In practice, there exist different variations of this approach.

Example

Query q = "application theory"

Boolean retrieval result

- application AND theory: B3, B17
- application OR theory: B3, B11, B12, B17

Vector retrieval result

- Query vector (0, 0.77…, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.63)
- Ranked Result:

B17	1.0
B3	0.69
B12	0.28
B11	0.28

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This examples provides an illustration of the differences of Boolean and vector space retrieval.

Implementation in Python

```
from sklearn.feature_extraction.text import TfidfVectorizer

tf = TfidfVectorizer(analyzer='word', ngram_range=(1,1), min_df = 2, stop_words = 'english')
features = tf.fit_transform(titles)
features.todense()[0]
```

```
matrix([[0.        , 0.        , 0.        , 0.        , 0.47145815,
         0.        , 0.88188844, 0.        , 0.        , 0.        ,
         0.        , 0.        , 0.        , 0.        , 0.        ,
         0.        ]])
```

```
query = tf.transform(["application theory"])
# transform extracts the features for a new text
query.todense()
```

```
matrix([[0.        , 0.77439663, 0.        , 0.        , 0.        ,
         0.        , 0.        , 0.        , 0.        , 0.        ,
         0.        , 0.        , 0.        , 0.        , 0.        ,
         0.63270045]])
```

```
from sklearn.metrics.pairwise import linear_kernel
# linear_kernel computes the scalar products

similarities = linear_kernel(query, features)[0]
result = [i for i in zip(similarities, docids)]
result.sort(reverse=True)
result
```

```
[(1.0, 'B17'),
 (0.6930162031082033, 'B3'),
 (0.2836540422799114, 'B12'),
 (0.2836540422799114, 'B11'),
 (0.0, 'B9'),
 (0.0, 'B8'),
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an implementation of basic vector space retrieval in Python.

Let the Boolean query be represented by {(1, 0, -1), (0, -1, 1)} and the document by (1, 0, 1). The document …

1. matches the query because it matches the first query vector
2. matches the query because it matches the second query vector
3. does not match the query because it does not match the first query vector
4. does not match the query because it does not match the second query vector

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The term frequency of a term is normalized …

1. by the maximal frequency of all terms in the document
2. by the maximal frequency of the term in the document collection
3. by the maximal frequency of any term in the vocabulary
4. by the maximal term frequency of any document in the collection

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The inverse document frequency of a term can increase …

1. by adding the term to a document that contains the term
2. by removing a document from the document collection that does not contain the term
3. by adding a document to the document collection that contains the term
4. by adding a document to the document collection that does not contain the term

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The Role of Document Length

When computing cosine similarity, document vectors are normalized

Result: for Query “information”:
doc2 will be favored because it is shorter

Singhal 1996

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For a long time, it was suspected that the standard vector space retrieval method favors short documents over long documents as a result of the normalization of document vectors. When normalizing document vectors, a term that occurs in the query would receive a lower weight in a longer document, as it shares the total weight of 1 with a larger number of terms, and thus the longer document is less likely to receive a high rank. Still, the question is whether this hurts retrieval performance.

In a seminal paper Singhal and co-authors provided empirical evidence confirming that retrieval performance is suffering from this problem. They compared for a TREC dataset the probability P(relevance) of a document of a given length of being relevant (by relying on the manual evaluation) with the probability P(retrieval) that a document of that length would be retrieved using standard vector space retrieval. The graph clearly shows that shorter documents have better chances to show up in a result, than longer ones, even when they are less relevant. The pivot point is the document length where the both probabilities match.

Normalization of Document Vector

Renormalize document vector

$$\overrightarrow{d_j} = \left(w_{1j}, \dots, w_{mj}\right), w_{ij} = tf_{ij} * idf_i$$

Standard normalization: $\overrightarrow{d^{norm}}_j = \frac{\overrightarrow{d_j}}{|\overrightarrow{d_j}|}$

$n = |\overrightarrow{d_j}|$ original normalization factor

$N(n)$ new normalization factor

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The standard approach for normalizing document vectors is to normalize it to length 1. This approach leads to the afore mentioned problem of favoring shorter documents.

One possibility to address the problem is to modify the normalization factor. If we consider the original normalization factor $n = |\overrightarrow{d_j}|$ the approach would be to replace it with a modified normalization factor $N(n)$ that is a function of the original normalization factor. This new normalization should be designed to favor longer documents. That is achieved by giving more weight to those documents, which again can be achieved by choosing a smaller normalization factor.

Compensating Bias towards Short Documents

New normalization $w = \frac{tf*idf}{N(n)} = \frac{tf*idf}{((1-s)*pivot+s*n)}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To correct for the influence of document length, different normalization schemes have been proposed. We introduce the one introduced in the work from Singhal.

A first observation is that the original and new normalization should be the same for the document length at which the probability of relevance and retrieval are the same, the pivot point. The function $N(n)$ is designed as a linear function that has larger values below the pivot and smaller values above. Therefore, a slope s for the function needs to be chosen. The pivot point and slope parameter needs to be determined empirically, by comparing retrieval performance of parameters for a given document collection.

Length Normalization

Weighting scheme: $\frac{tf*idf}{((1-s)*pivot+s*n)}$

s = slope

n = original normalization factor, i.e. $|\vec{d}|$

Result

- If n < pivot, then N(n) > n and therefore weights will be smaller

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With this normalization factor N(n) the weights of shorter documents, more precisely for those where the normalization factor n is smaller, will be reduced.

Pivoted Unique Query Normalization

Practical implementation of the approach

Weighting scheme: $\frac{tf*idf}{((1-s)*pivot+s*u)}$

with

$s = 0.2$

$pivot =$ average number of distinct terms in a document

$u =$ number of distinct terms in the document

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Determining the pivot and slope parameters empirically might be costly or not possibly in practice. Therefore, heuristics for determining those parameters based on statistical properties of the document collections have been proposed.

One of them is pivoted unique query normalization. It uses the average number of distinct terms in a document as pivot point. The slope parameter is a constant value, where 0.2 has been identified as suitable. Document length is measured in terms of number of distinct terms in a given document.

Variants of Vector Space Retrieval Model

The vector model with tf-idf weights is a good ranking strategy for general collections

- many alternative weighting schemes exist, but are not fundamentally different

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times\mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Different variants of tf-idf weighting schemes have been developed and used over time. They can be combined with each other, also independently for the weighting of document and query terms. One important variant is the logarithmic weighting of term frequencies, which moderates the influence of very frequently occurring terms in documents. In the normalization approaches, the parameter u in pivoted unique, corresponds to the number of unique terms in a document.

Discussion of Vector Space Retrieval Model

Advantages

- term-weighting improves quality of the answer set
- partial matching allows retrieval of docs that approximate the query conditions
- cosine ranking formula sorts documents according to degree of similarity to the query

Disadvantages

- assumes independence of index terms; not clear that this is a disadvantage
- No theoretical justification why the model works

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We summarize here the main advantages of the vector space retrieval model. It has proven to be a very successful model for general text collections, i.e., if there exists no additional (context) information on the documents that could be exploited, e.g., from a specific application domain. Providing a ranked result improves the usability of the approach, as users can more easily distinguish more relevant documents from less relevant documents. The model inherently assumes that there exist no mutual dependencies in the occurrences of the terms, i.e., that certain terms appear together more frequently than others. Studies have shown that taking such co-occurrence probabilities additionally into account, can actually HURT the performance of the retrieval system. The reason is that co-occurrence probabilities are often related to specific application domains and thus do not easily transfer from context to another.

One of the principal criticisms of the vector space model is the lack of theoretical explanation why it works. At the end, it is a heuristics. This drawback has been addressed with other models, in particular probabilistic retrieval models.

1.2.4 Evaluating Information Retrieval

Quality of a retrieval model depends on how well it matches user needs!

Comparison to database querying

– correct evaluation can be formally verified

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The performance of an information retrieval system can be determined by evaluating the satisfaction of users, which is a context-dependent task. This is fundamentally different to database querying, where there exist formal criteria for validating correctness of query answering.

Evaluating Information Retrieval

Test collections with test queries, where the relevant documents are identified manually are used to determine the quality of an IR system (e.g. TREC)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Since there exists no formal criterion whether an information retrieval query is correctly answered, other means for evaluating the quality of an information retrieval system are required. The basic approach is to compare the performance of a specific system to human performance for the same retrieval task. For that purpose benchmark collections of documents, such as TREC (http://trec.nist.gov/), are created and for selected queries human experts select the relevant documents. Note that this approach assumes that humans have an agreed-upon, objective notion of relevance, an assumption that is in general not satisfied.

1.2.4.1 Recall and Precision

Recall is the fraction of relevant documents retrieved from the set of total relevant documents collection-wide

Precision is the fraction of relevant documents retrieved from the total number retrieved (answer set)

	Relevant	Non-relevant
Retrieved	True positives (tp)	False positives (fp)
Not Retrieved	False negatives (fn)	True negatives (tn)

$$R = \frac{tp}{tp + fn} = P(retrieved|relevant)$$

$$P = \frac{tp}{tp + fp} = P(relevant|retrieved)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The results of IR systems can be compared to the expected result in two ways:

1. **Recall** measures how large a fraction of the expected results is actually found.
2. **Precision** measures how many of the results returned are actually relevant.

Important note: This measure evaluates an **unranked** result set. All elements of the result are considered as equally important.

Recall and Precision – A Tradeoff

Suppose you search for "Theory of Relativity".

Optimizing recall: retrieve all pages mentioning "theory" and "relativ*"

- We will have probably most documents talking about the topic
- We might have results such as, "In theory, I feel relatively good", "Relative to the theory of evolution …" etc.

Optimizing precision: retrieve all pages mentioning "relativity theory" and "expanding universe"

- Most likely all results are relevant
- But we might miss "the theory of relativity by Einstein"

Thus, high recall hurts precision and vice versa

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates a fundamental trade-off between recall and precision. Achieving high recall in general lowers precision and vice versa. In the extreme case, it is always possible to achieve 100% recall at the expense of having the lowest possible precision.

Combined Measures

Sometimes we want to characterize the performance of a retrieval system by one number

F-Measure: weighted harmonic mean

$$F = \frac{1}{\alpha \frac{1}{P} + (1-\alpha)\frac{1}{R}}, \qquad \alpha \epsilon [0,1]$$

F1: balanced F-Measure with $\alpha = \frac{1}{2}$: $\quad F1 = \frac{2PR}{P+R}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Sometimes one prefers to characterize the performance of an information retrieval system by a single measure, by combining recall and precision in a single quantity. Arithmetic mean could be a possible choice, but is not very suitable. When choosing 100% recall we would always have at least 50% as the arithmetic mean between recall and precision.

Therefore, the harmonic mean is used. It can be tuned by a parameter alpha. Larger values of alpha emphasize the importance of precision and smaller ones the importance of recall.

Accuracy

$$A = \frac{TP+TN}{TP+TN+FP+FN} = \frac{TP+TN}{N}$$

Appropriate metric when

- Classes are not skewed
- Errors have the same importance

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Accuracy is another possible measure for performance. Accuracy is a standard way to measure performance of binary classifiers, and retrieval can be understood as the task of classifying documents into relevant and non-relevant ones. However, accuracy only works well if the positive and negative cases are approximately equally distributed, and the importance of having false positives and false negatives is similar.

We will illustrate the potential issues in the following.

Accuracy - Pitfall

Classifier 1		Class: Fraud	Class: ¬Fraud
Classified	Fraud	5	10
	¬Fraud	5	80

A = 85/100 = 0.85

Always ¬Fraud		Class: Fraud	Class: ¬Fraud
Classified	Fraud	0	0
	¬Fraud	10	90

A = 90/100 = 0.90

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Assume we want to retrieve documents that indicate a fraud and that in the majority of cases a document does not refer to fraud.

Accuracy as a performance metrics is inappropriate in case of such skewed class distributions. The typical problem is that a trivial classifier that classifies everything as belonging to the majority class, can achieve easily higher accuracies than a classifier that attempts to also correctly classify samples in the minority class.

Which is the “best” classifier?

Classifier 1		Class A	Class B
Classified	A	45	20
	B	5	30

Classifier 2		Class A	Class B
Classified	A	40	10
	B	10	40

A. Classifier 1

B. Classifier 2

C. Both are equally good

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Which is the “best” classifier?

Classifier 1

		Class	
		Cancer	¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

Classifier 2

		Class	
		Cancer	¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

A. Classifier 1

B. Classifier 2

C. Both are equally good

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Precision and Recall: Example

Classifier 1		Class: Cancer	Class: ¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

P_1=45/65=0.69

R_1=45/50=0.9

Classifier 2		Class: Cancer	Class: ¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

P_2=40/50=0.8

R_2=40/50=0.8

Everybody has cancer		Class: Cancer	Class: ¬Cancer
Classified	Cancer	50	50
	¬Cancer	0	0

P = 50/100 = 0.5

R = 50/50 = 1

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With precision and recall we can better control the kind of results we prefer to obtain. For example, for the case of cancer detection we would prefer to have higher recall to miss fewer cancer cases, even if we diagnose erroneously cancer in a few cases. Therefore, classifier 1 would be preferred over classifier 2 (which did better in terms of accuracy). Of course we can increase the recall arbitrarily, e.g., with a trivial classifier diagnosing cancer for everyone. But this also causes that 50% of the patients with no cancer go home worried about their health status.

That is why still precision needs to be considered in the evaluation as second criterion.

Note that by using precision and recall we make the evaluation “asymmetric”. We focus in the evaluation on the positive class, since higher recall means more positive cases identified.

F-Score: Example (alpha = 1/2)

Classifier 1		Class: Cancer	Class: ¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

F_1=2*(0.69*0.9)/(0.69+0.9)
= 0.78

Classifier 2		Class: Cancer	Class: ¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

F_2=2*(0.8*0.8)/(0.8+0.8)
=0.8

Everybody has cancer		Class: Cancer	Class: ¬Cancer
Classified	Cancer	50	50
	¬Cancer	0	0

F=2*(0.5*1)/(0.5+1)=0.66

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With the F1 score still classifier 2 is evaluated as slightly better than classifier 1.

F-alpha-Score: Example (alpha = 1/5)

Classifier 1		Class: Cancer	Class: ¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

Classifier 2		Class: Cancer	Class: ¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

F_1=5*(0.69*0.9)/(4*0.69+0.9) = 0.84

F_2=5*(0.8*0.8)/(4*0.8+0.8) =0.8

Everybody has cancer		Class: Cancer	Class: ¬Cancer
Classified	Cancer	50	50
	¬Cancer	0	0

F=5*(0.5*1)/(4*0.5+1)=0.83

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With the F-alfa score, alfa=1/5, we can give more importance to recall and as a result indeed classifier 1 turns out to be have a better performance under this measure.

Sometimes also an F-beta score is computed substituting alfa = (1/1+beta^2), resulting in $F_\beta = (1+\beta^2)\frac{PR}{\beta^2 P+R}$

In this example we would have beta = 2.

If the top 100 documents contain 50 relevant documents …

1. the precision of the system at 50 is 0.25
2. the precision of the system at 100 is 0.5
3. the recall of the system is 0.5
4. All of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

If retrieval system A has a higher precision at k than system B …

1. the top k documents of A will have higher similarity values than the top k documents of B
2. the top k documents of A will contain more relevant documents than the top k documents of B
3. A will recall more documents above a given similarity threshold than B
4. the top k relevant documents in A will have higher similarity values than in B

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.2.4.2 Precision/Recall Tradeoff in Ranked Retrieval

An IR system ranks documents by a similarity coefficient, allowing the user to trade off between precision and recall by choosing the cutoff level

Precision and recall depend on the number of results retrieved:
P@k = precision for the top-k documents
R@k = recall for the top-k documents

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One of the two measures of recall and precision can always be optimized. Recall can be optimized by simply returning the whole document collection, whereas precision can be optimized by returning only very few results. Important is the trade-off: the higher the precision for a specific recall, the better the information retrieval system. A hypothetical, optimal information retrieval system would return results with 100% percent precision always. If a system ranks the results according to relevance the user can control the relation between recall and precision by selecting a threshold of how many results the user inspects.

Evaluating Ranked Retrieval

Recall-Precision Plot

R N R N R R N N R R R N R N R R
(10 relevant documents)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The recall-precision graph lists on the x-axis the documents according to their ranking and on the y-axis the precision achieved when returning the set of documents up to the rank.

If we draw the Recall-Precision graph for a ranked retrieval result, we will find the following picture. The precision will always drop when non-relevant documents are returned and increase when relevant documents are returned. The green dots correspond to relevant documents and the red dots to non-relevant ones.

Interpolated Precision

$P(r)$ precision after retrieving r relevant documents

Interpolated precision: $P_{int}(r) = \max_{r' \geq r} P(r)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to convert the recall-precision plot into a monotonically decreasing function, interpolated precision is introduced, which returns always the maximum precision over all recalls r' that are greater than r.

Mean Average Precision (MAP)

Given a set of queries Q

For each $q_j \epsilon Q$ the set of relevant documents $\{d_1, \dots d_{m_j}\}$

D_{jr} the top r relevant documents for query q_j

$P_{int}(r)$ interpolated precision of result D_{jr}

$$MAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{r=1}^{m_j} P_{int}(D_{jr})$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Mean Average Precision has been shown to be a robust measure for evaluating the quality of a ranked retrieval system for a query collection Q. When a relevant document is not retrieved at all, the precision value in the MAP is 0.

Example

Assume 4 results are returned for a query q:

R N R N

P@1 = 1, P@2 = 0.5, P@3 = 2/3, P@4 = 0.5

P_{int} (1)= 1, P_{int} (2) = 2/3

Then MAP({q}) = 1/2 (1 + 2/3) = 5/6

(note : only precision values when retrieving a relevant document are considered)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A simple example where the query set Q to be evaluated consists of a single query q. For multiple queries the results would be averaged.

ROC Curve

True positives

False positives

Specificity S: $1 - \mathrm{S} = \frac{fp}{fp+tn} = P(retrieved|not\ relevant)$

Specificity gives information about how many of the true negatives have been retrieved as false positives

- The steeper the curve rises at the beginning, the better
- The larger the area under the curve, the better (AUC)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Another frequently used approach to globally evaluate ranked retrieval is the ROC curve. The ROC curve is frequently loosely called a recall-precision curve, which is not accurate (as the figure shows).

The ROC curve relates specificity (on the x-axis) to recall (on the y-axis). Specificity measures the fraction of true negatives that are detected in proportion to all negatives. Thus 1 – specificity is the rate of false positives that have been retrieved, the so-called **false positive rate (FPR)**. In other words, it is the fraction of non-relevant documents among all non-relevant documents that occur in the result. The desired behavior of an IR system is that the curve raises quickly at the beginning, which means that many relevant results are retrieved without having a high fraction of non-relevant documents. This is also equivalent to saying that the area under the curve should be large. Therefore, the area under the ROC curve is sometimes considered similarly as an evaluation of the overall retrieval quality of a retrieval system, analogous to the MAP measure.

Example:

- When the recall is at 0.5 then 1-S is at 0.1. This implies that S is 0.9. Then we can conclude that fp = 1/9 tn, or in other words when retrieving half of the relevant documents, then we have also retrieved about 10% of the non-relevant documents.

Considering Similarity Scores and Position

Evaluating the top-k results returned for a query

- Let s(d_i) be the score given for document d_i (e.g. 1 = relevant, 0 = non relevant)

Cumulative gain (CG): $\mathrm{CG} = \sum_{i=1}^{k} s(d_i)$

- Does not consider position of scored documents

Discounted cumulative gain (DCG):

$$\mathrm{DCG} = \sum_{i=1}^{k} \frac{s(di)}{\log_2(i+1)}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

NDCG builds on the concepts of cumulative gain (CG) and discounted cumulative gain (DCG).

With CG the scores of the top-k ranked items are added up. The higher this value, the higher the quality of the predicted scores.

CG does not consider the position of the scores, whereas in practice the earlier positions are more important. Therefore, with DCG a weighting function for the positions is included, which gives higher weight to the earlier positions. The method has its background in finance, where for investments gains in earlier years are higher weighted than gains further into the future.

Normalized DCG (NDCG)

Values of DCG are not normalized

- depend on score distribution and choice of k

Compute the DCG for the top-k items sorted in optimal order, i.e., by decreasing scores: $iDCG(k)$

Then normalize the DCG: $NDCG = \frac{DCG}{iDCG(k)}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Finally, the metrics provided by DCG is not normalized and can thus not be compared for different rating schemes and choices of k. To calibrate the values, the DCG for the optimally sorted items is considered as baseline. By normalizing the DCS using this baseline, one obtains the NDCG, as a standardized measure for ranking quality in the interval [0,1].

Example

Assume the following score distribution:

(3,3,3,2,2,1)

Two methods A, B ranking top-3 documents

A ranks (2,3,3)

B ranks (3,3,2)

Then $CG(A) = CG(B)$ but

$$DCG(A) = \frac{2}{\log_2 2} + \frac{3}{\log_2 3} + \frac{3}{\log_2 4}$$
$$< \frac{3}{\log_2 2} + \frac{3}{\log_2 3} + \frac{2}{\log_2 4} = DCG(B)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The example illustrates the different steps in the calculation of NDCG.

Example

Then

$$iDCG(3) = \frac{3}{\log_2 2} + \frac{3}{\log_2 3} + \frac{3}{\log_2 4} = 6.39$$

Therefore, normalized values

$$NDCG(A) = \frac{5.39}{6.39} = 0.84$$

$$NDCG(B) = \frac{5.89}{6.39} = 0.92$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In this step the DCG values are normalized.

Let the first four documents retrieved be R N N R. Then the MAP is

1. 1/2
2. 3/4
3. 3/2
4. 5/6

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Let the first four documents retrieved be R N R N (scores: R = 1, N = 0) Then the DCG is

1. 1/2
2. 3/4
3. 3/2
4. 5/6

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

References

Course material based on

- Ricardo Baeza-Yates, Berthier Ribeiro-Neto, Modern Information Retrieval (ACM Press Series), Addison Wesley, 1999.
- Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008 (http://www-nlp.stanford.edu/IR-book/)
- Course Information Retrieval by TU Munich (http://www.cis.lmu.de/~hs/teach/14s/ir/)
- Singhal, A., Salton, G., Mitra, M., & Buckley, C. (1996). Document length normalization. Information Processing & Management, 32(5), 619-633.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.2.5 Probabilistic Information Retrieval

The notion of similarity in the vector space model does not have an explanation how it relates to relevance

- The similarity values are just used to rank

An information retrieval model deals with uncertainty on the user's information needs

- Probability theory provides a principled approach to reason about this uncertainty

Probabilistic IR models attempt to provide an explainable model of relevance

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One of the key drawbacks of the vector space retrieval model is the lack of interpretability of the similarity values. This gave rise to the development of probabilistic retrieval models, that attempt to determine relevance as a probabilistic concept with an explainable probabilistic model.

Query Likelihood Model

Given query q, determine the probability $P(d|q)$ that document d is relevant to query q

Bayes Rule $P(d|q) = \frac{P(q|d)P(d)}{P(q)}$

Assumptions

- $P(d)$, the probability of a document occurring is uniform across a collection
- $P(q)$ is the same for all queries

Thus: $P(d|q)$ can be derived from $P(q|d)$, the query likelyhood

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In probabilistic retrieval the objective is to introduce a notion of probability that relates to the relevance of a document d with respect to a query q.

Since we can assume that the probability of a document to occur in a collection is constant (which is correct assuming all documents are different), and the probability of a query to occur is the same for all documents, using Bayes rule, the problem of determining whether a document is relevant for a query is equivalent to the problem of determining whether a query is relevant to a document. The latter probability $P(q|d)$ is also called the query likelihood.

So probabilistic IR can be framed as the question of what is the probability for a specific query to occur, given a specific document. One way to understand why the problem is formulated this way, is that it is more feasible to derive a rich model from a document, than from a query, which contains little information.

Language Modeling

Query likelihood: determine $P(q|d)$

Assume each document d is generated by a Language Model M_d

- a language model is a mechanism that generates the words of the language

Then $P(q|d)$ can be interpreted as the probability that the query q was generated by the language model M_d

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The concept of query likelihood gives now rise to the following approach to capture relevance of a document to a query. We assume that documents are the result of language model. A language model is a (in general probabilistic) process that produces text, and a given document d is assumed to be produced by its specific language model M_d. Then the problem of retrieval can be viewed in the following way: if a query is relevant to a document, it should have been produced by the same language model as the document. Using this argument, the query likelihood corresponds to the probability that the query has been produced by the same language model as the document.

Let's have now a more detailed look in what a language model is and how we use it implement this approach practically.

What is a Language Model?

Deterministic language model = automaton = grammar

This model can produce:

"information retrieval"
"information retrieval information retrieval"

It cannot produce:

"retrieval information"

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A very simple example of a generative language model is a deterministic automaton. Deterministic automatons are used to recognize or produce regular languages.

Probabilistic Language Model

Unigram model: assign a probability to each term to appear

- More complex models can be used, e.g., bigrams

Model M_1		Model M_2	
STOP	0.2	STOP	0.2
the	0.2	the	0.15
a	0.1	a	0.12
frog	0.03	frog	0.0002
toad	0.03	toad	0.0001
said	0.02	said	0.01
likes	0.015	likes	0.01
dog	0.01	dog	0.04

Two different language models derived from 2 documents

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Instead of using a deterministic automaton, we can also use a probabilistic state automaton, in other words, a Markov process. In the simplest case the automaton has a single state, and every state transition emits with a certain probability one term out of a vocabulary. In addition, the automaton can stop with a certain probability. The table captures the transition probabilities of two possible models M1 and M2. In the two models, the probability to stop is given as P(STOP|Q) = 0.2.

Probability to Create a Query

What is the probability that a query q has been generated by model M?

Example: q = the frog said dog STOP
$P(q|M_1)$ = 0.2 * 0.03 * 0.02 * 0.01 * 0.2 = 0.000 000 24
$P(q|M_2)$ = 0.15 * 0.0002 * 0.01 * 0.04 * 0.25 = 0.000 000 003

Retrieval becomes the problem of computing for a query q the probability $P(q|M_d)$ for all the documents d

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given a language model for generation of documents, we can compute using that model the probability that a given query q has been generated by the model of a document d. We give an example showing such a computation. With this approach we are now ready to compute query likelihood for all documents of a document collection.

Learning the Model

Learning the model means we must estimate the probability of a query to occur

First step: estimate how likely a single term t occurs

Maximum Likelihood Estimation (MLE) of probabilities under Unigram Model

$$\hat{P}_{mle}(t|M_d) = \frac{tf_{t,d}}{L_d}$$

where

- $tf_{t,d}$ is the number of occurrences of t in d (term frequency)
- L_d is the number of terms in the document (document length)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For applying the probabilistic retrieval method described before, we need to learn a language model of each document. The learning is performed using Maximum Likelihood Estimation (MLE). In the case of the unigram model, this is a straightforward task. We need to estimate the probabilities of terms to occur in a document. This is done by counting the number of occurrences of terms and normalizing by document length.

Using the Model

Independence assumption: different terms in a query are assumed to occur independently

$$\hat{P}(q|M_d) = \prod_{t \epsilon q} \hat{P}_{mle}(t|M_d)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Based on the estimation of term probabilities we can compute a query probability, by making an independence assumptions on terms. This results in an estimation of the probability of a query to occur under a given document model.

Consider the document:
"Information retrieval is the task of finding the documents satisfying the information needs of the user"

Using MLE to estimate the unigram probability model, what is $P(\text{the}|M_d)$ and $P(\text{information}|M_d)$?

1. 1/16 and 1/16
2. 1/12 and 1/12
3. 1/4 and 1/8
4. 1/3 and 1/6

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Consider the following document

d = “information retrieval and search”

1. P(information search | M_d) > P(information | M_d)
2. P(information search | M_d) = P(information | M_d)
3. P(information search | M_d) < P(information | M_d)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Issues with MLE

Problem 1: if the query contains a term not occurring in the document, then $\hat{P}(q|M_d) = 0$!

Problem 2: this is an estimation! A term that occurs once, might have been “lucky”, whereas another one with same probability to occur is not contained in the document

- need to give non-zero probability to unseen terms!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Applying the described approach to estimate relevance of a document to a query has a practical problem: if the query contains a term not occurring in the document the estimated probability will be unavoidably zero, since one of the factors of the product computing that probability will be zero. In other words, the query cannot be generated by the document model, thus the document is not relevant to the query. This is not only impractical, but also not meaningful from a more another perspective. Since we used MLE to generate the model, we were using the statistics of one specific document, that has been generated by a potentially complex document model. It might be the case that the specific generated document by chance does not contain certain terms that are part of the possible terms in the document according to the document model.

Smoothing

Idea: add a small weight for terms not occurring in a document

- the weight should be smaller than the normalized collection frequency

$$\hat{P}(t|M_c) \leq cf_t/T$$

where

- cf_t = number of times term t occurs in collection
- T = total number of terms in collection

Smoothed estimate

$$\hat{P}(t|d) = \lambda \hat{P}_{mle}(t|M_d) + (1-\lambda)\hat{P}_{mle}(t|M_c)$$

M_c = language model of the whole collection

λ = tuning parameter

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To address these problems an approach called smoothing is applied. The idea is to assume that in fact every term potentially could occur in the document generated by its document model, including those that are not part of the actual document; only that the probability of terms not seen in the document is presumably smaller than the probability of the term to occur in the overall document collection. The smoothed estimate then combines the estimated likelihood to occur in the document according to the document model, with the estimated likelihood of a term occurring in the general document collection, modeled as a generic language model using the statistics from the whole document collection.

Probabilistic Retrieval

With smoothing the relevance is computed as

$$P(d|q) \propto P(d) \prod_{t \epsilon q} ((1-\lambda)P(t|M_c) + \lambda P(t|M_d))$$

From a technical perspective the probabilities are computed using term and document frequencies

- the same data is used as in vector space retrieval

Probabilistically motivated models show generally better performance

- But parameter tuning (λ) is critical
- λ can be query-dependent, e.g., query size

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we summarize the approach for probabilistic retrieval. From a technical perspective computational cost of probabilistic retrieval is comparable to that vector space retrieval. The computation of the likelihoods for the document models requires to determine term frequencies, so in that sense it is equivalent. For deriving the collection model, the global term frequencies need to be computed, which again is like computing inverse document frequencies in a document collection.

In practice, the fine tuning of the model parameters (in that case λ) is essential for the model to perform well. It is also possible to make the parameter dependent on the query, in particular on the query size.

Example

Collection consisting of d_1 and d_2

d_1: Einstein was one of the greatest scientists

d_2: Albert Einstein received the Nobel prize

Query q: Albert Einstein

Using λ=1/2:

$P(q|d_1) = ½ * (0/7 + 1/13) * ½ * (1/7 + 2/13) \approx 0.0057$

$P(q|d_2) = ½ * (1/6 + 1/13) * ½ * (1/6 + 2/13) \approx 0.0195$

Albert Einstein

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is a simple example illustrating the use of probabilistic retrieval. Note that the document lengths of d1 and d2 are 7 and 6, and that the collection length is 13.

Example: Comparing VS and PR

	Precision			
Rec.	tf-idf	LM	%chg	
0.0	0.7439	0.7590	+2.0	
0.1	0.4521	0.4910	+8.6	
0.2	0.3514	0.4045	+15.1	*
0.3	0.2761	0.3342	+21.0	*
0.4	0.2093	0.2572	+22.9	*
0.5	0.1558	0.2061	+32.3	*
0.6	0.1024	0.1405	+37.1	*
0.7	0.0451	0.0760	+68.7	*
0.8	0.0160	0.0432	+169.6	*
0.9	0.0033	0.0063	+89.3	
1.0	0.0028	0.0050	+76.9	
Ave	0.1868	0.2233	+19.55	*

Ponte & Croft, 1998

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is a result reported from comparing vector space retrieval with probabilistic retrieval. In this experiment probabilistic retrieval improves precision significantly, in particular for higher values of recall. (LM = language model).

Properties of Retrieval Models

	Vector Space Model	Language Model	BM25 (another prob. Model)
Model	geometric	probabilistic	probabilistic
Length normalization	Requires extensions (pivot normalization)	Inherent to model	Tuning parameters
Inverse document frequency	Used directly	Smoothing and collection frequency has similar effect	Used directly
Multiple term occurrences	Taken into account	Taken into account	Ignored
Simplicity	No tuning required	Tuning essential	Tuning essential

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we compare the characteristics of the vector space model with the probabilistic retrieval model based on language models that we have introduced. BM25 is another probabilistic model, that is today considered as one of the most performant retrieval models.

One aspect that is taken implicitly care off in the probabilistic retrieval model based on language models is normalization for document length. For vector space retrieval, specific modifications of the model are used, as we have seen earlier. For collections with widely varying document lengths this proved to be a useful improvement.

In general, the vector space model is preferred, when a quick and simple solution is sought. For probabilistic models, better performance can be achieved, but this depends on careful parameter tuning.

1.2.6 Query Expansion

If the user query does not contain any relevant term, a corresponding relevant document will not show up in the result

Example: query “car” will not return “automobile”

How to add such documents (increase recall)?

Idea: System adds query terms to user query!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Users cannot predict or imagine all possible ways of how the concepts they are interested to find in their search can be expressed in natural language. Consequently, even under the vector space retrieval model, relevant results are missed, for the example, when the user does not provide different synonyms (different terms with the same meaning) in a query.

In information retrieval we are most of the time concerned about precision, since the assumption is that there exist many relevant documents. But in certain applications recall is more important. Good examples for this is searching for scientific publications, where only a few papers might be of interest or security-relevant applications where also rare risks need to be detected.

In the following we will see one possible approach to deal with this problem, namely extending the user query automatically by the system with additional query terms.

Two Methods for Extending Queries

1. Local Approach:

- Use information from **current query results**: *user relevance feedback*

2. Global Approach:

- Use information from a **document collection**: *query expansion*

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the following we will present two types of approaches to query extension. They differ in the way of how new additional query terms are obtained. In the local approach the source of information is the current user query, respectively results produced by answering the user query. In the global approach, the source of information is an existing document collection, either the documents corpus that is queried by the user, or another, external collection of documents.

1.2.6.1 User Relevance Feedback

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In general, a user does not necessarily know what is his information need and how to appropriately formulate a query. But usually, a user can well identify relevant documents. Therefore, the idea of user relevance feedback is to reformulate a query by taking into account feedback of the user on the relevance of already retrieved documents.

The advantages of such an approach are the following:

- The user is not involved in query formulation, but just points to interesting data items.
- The search task can be split up in smaller steps.
- The search task becomes a process converging to the desired result.

Feedback from Users

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The situation when receiving feedback from users can be described as follows: the retrieval system returns some result set R that is presented to the user. This result set overlaps with the set of relevant documents (C_r). The user can the identify within the result set both documents that are relevant and non-relevant. This gives the two feedback sets D_r and D_n.

Rocchio Algorithm

Rocchio algorithm: find a query that optimally separates relevant from non-relevant documents

$$\vec{q}_{opt} = \arg\max_{\vec{q}} \left[\mathrm{sim}\left(\vec{q}, \mu(D_r)\right) - \mathrm{sim}\left(\vec{q}, \mu(D_n)\right)\right]$$

Centroid of a document set

$$\mu(D) = \frac{1}{|D|} \sum_{d \in D} \vec{d}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic approach for user relevance feedback was introduced by Rocchio. It is based on the observation, that the centroid of all document vectors of a document set D can be considered as the most characteristic representation of the document set. In order to to construct a query q_{opt} that optimally separates relevant from non-relevant documents, such a query has to have maximal similarity with the set of relevant documents, respectively its centroid, and maximal dissimilarity with the set of non-relevant documents, respectively its centroid. This can be achieved by finding a query that maximizes the difference among these two similarity values.

Illustration of Rocchio Algorithm

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We now motivate of how the optimal query vector can be found with an illustration. Assume that the relevant documents are marked by circles, and the non-relevant documents are marked by crosses, and that the vector space has 2 dimensions. When we consider the simply centroid of the relevant documents as a search query, then we see that we cannot achieve optimal precision and recall at the same time.

Illustration of Rocchio Algorithm

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We therefore consider also the centroid of the non-relevant documents as part of the user relevance feedback. We compute the difference vector among the two centroids, and we will use this difference vector to “move away” the query from the non-relevant documents.

Illustration of Rocchio Algorithm

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We add the difference vector to the centroid for the relevant documents. The resulting optimal query vector now can include all relevant documents in its result, without including non-relevant ones. In practice, such a clear separation will not always be achieved, but it has been shown that under some additional assumptions, this method is the optimal way to construct a query separating relevant from non-relevant documents.

Identifying Relevant Documents

Following the previous reasoning the optimal query is

$$\vec{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_n)]$$

Under cosine similarity

$$\vec{q}_{opt} = [\mu(D_r) - \mu(D_n)]$$

Practical issues

- User relevance feedback is not complete
- Users do not necessarily identify non-relevant documents
- Original query should continue to be considered

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We derived in the previous illustration an optimal query vector, under a model that uses Euclidean distance as metrics. This approach is frequently used for illustration of how such a vector can be constructed. Since in the vector space model we use cosine similarity as similarity measure, the optimal vector under this metric is different. It is given as the difference vector of the two centroids.

Constructing an optimal query vector as described is only theoretically possible, since the complete information on relevant and non-relevant documents is missing in practice. Therefore, the theoretical considerations serve as a basis to devise a practical scheme, that is **approximating** the theoretical optimal vector by a vector that can be constructed from available data.

SMART: Practical Relevance Feedback

Approximation scheme for the theoretically optimal query vector

If users identify some relevant documents D_r from the result set R of a retrieval query q

- Assume all elements in $R \setminus D_r$ are not relevant, i.e., $D_n = R \setminus D_r$
- Modify the query to approximate theoretically optimal query

$$\vec{q}_{approx} = \alpha\vec{q} + \frac{\beta}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \frac{\gamma}{|R \setminus D_r|}\sum_{\vec{d}_j \notin D_r}\vec{d}_j$$

- α, β, γ are tuning parameters, $\alpha, \beta, \gamma \geq 0$
- Example: $\alpha = 1$, $\beta = 0.75$, $\gamma = 0.25$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The approximation scheme for user relevance feedback is called SMART. It assumes that users have identified some relevant documents. Then the scheme assumes that all other documents should be considered as non-relevant. This results in a modification of the original query that is controlled by 3 tuning parameters.

Since the assumption that all documents that have not been marked relevant are non-relevant is of course not correct, two mechanisms are used to moderate the impact of this wrong assumption:

1. The original query vector is maintained, in order not to drift away too dramatically from the original user query.
2. The weight given for the modification using the centroid of non-relevant documents is generally kept lower than the weight for the centroid of the relevant documents

Example

Query q= "application theory"

Result

0.77: B17 The Double Mellin-Barnes Type Integrals and Their Applications to Convolution Theory
0.68: B3 Automatic Differentiation of Algorithms: Theory, Implementation, and Application
0.23: B11 Oscillation Theory for Neutral Differential Equations with Delay
0.23: B12 Oscillation Theory of Delay Differential Equations

Query reformulation

$$\vec{q}_{approx} = \frac{1}{4}\vec{q} + \frac{1}{4}\vec{d}_3 - \frac{1}{12}(\vec{d}_{17} + \vec{d}_{12} + \vec{d}_{11}), \alpha = \beta = \gamma = \frac{1}{4}$$

Result for reformulated query

0.87: B3 Automatic Differentiation of Algorithms: Theory, Implementation, and Application
0.61: B17 The Double Mellin-Barnes Type Integrals and Their Applications to Convolution Theory
0.29: B7 Knapsack Problems: Algorithms and Computer Implementations
0.23: B5 Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example shows how the query reformulation works. By identifying document B3 as being relevant and modifying the query vector it turns out that new documents (B5 and B7) become relevant. The reason is that those new documents share terms with document B3, and these terms are newly considered in the reformulated query.

Discussion

Underlying assumptions of SMART algorithm

1. Original query contains sufficient number of relevant terms
2. Results contain new relevant terms that co-occur with original query terms
3. Relevant documents form a single cluster
4. Users are willing to provide feedback (!)

All assumptions can be violated in practice

Practical considerations

- Modified queries are complex → expensive processing
- Explicit relevance feedback consumes user time → could be used in other ways

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Concerning the first assumption, if the initial query of the user does not contain sufficient information to retrieve a sufficiently large number of documents that are relevant to the true interest of the user (i.e., has sufficient recall), the relevance feedback system will not be able to discover additional relevant terms.

Concerning the second assumption, new terms can only be included as part of the modified query, if they co-occur at least in some documents together with original query terms. Otherwise, these terms could never be part of relevant documents in the result of the original query (why?).

Concerning the third assumption, implicitly the SMART algorithm assumes that all relevant documents are part of one cluster in the vector space. If they form multiple clusters, it is not able to correctly produce a query that can retrieve the relevant documents.

Can documents which do not contain any keywords of the original query receive a positive similarity coefficient after relevance feedback?

1. No
2. Yes, independent of the values β and γ
3. Yes, but only if $\beta > 0$
4. Yes, but only if $\gamma > 0$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Which year Rocchio published his work on relevance feedback?

A. 1965
B. 1975
C. 1985
D. 1995

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Pseudo-Relevance Feedback

If users do not give feedback, automate the process

- Choose the top-k documents as the relevant ones
- Extend the query by selecting from the top-k documents the most relevant terms, according to some weighting scheme
- Alternatively: apply the SMART algorithm

Works often well

- But can fail horribly: query drift

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The idea of relevance feedback has also been adopted for an automated extension of queries. Instead of the user selecting relevant documents, the system automatically chooses the top-k results as the set of relevant documents and then extends the query either by selecting some most relevant terms using a weighting scheme or applying the SMART algorithm.

This works well if the original query already separates well the topic of interest from other topics. If this is not the case, the method can fail catastrophically, driving the query towards a topic that is different from the originally intended one, where irrelevant query terms reinforce each other.

Weighting Schemes

Term ranking algorithm	Formula
Algorithms based on frequency heuristics	
total_freq	$\text{Score}(t) = f_{R,t}$, where $f_{R,t}$ is the total frequency of term t within the set of pseudo-relevant documents
IDF	$\text{Score}(t) = \log \frac{N}{n_t}$
r_lohi [1]	$\text{Score}(t) = r_t$ for ties, n_t in ascending order, where r_t is the number of pseudo-relevant documents containing term t
Algorithm based on vector space model	
Rocchio	$\text{Score}(t) = \sum_{\forall d \in R} w_{d,t}$
Algorithms based on distribution analysis	
F4MODIFIED [2,3]	$\text{Score}(t) = \log \frac{p_t}{1 - p_t} - \log \frac{q_t}{1 - q_t}$
EMIM [1]	$\text{Score}(t) = \sum_{i \in \{t, \bar{t}\}, j \in \{R, \bar{R}\}} P(i,j) \log \frac{P(i,j)}{P(i)P(j)}$
RSV [4]	$\text{Score}(t) = w_t (p_t - q_t)$
KLD [5]	$\text{Score}(t) = p_t \cdot \log \frac{p_t}{c_t}$
CHI2 [5]	$\text{Score}(t) = \frac{(p_t - c_t)^2}{c_t}$
CHI1 [5]	$\text{Score}(t) = \frac{(p_t - c_t)}{}$

p_t: the probability of occurrence of term t in the set of pseudo-relevant documents, q_t: the probability of occurrence of term t in the set of non-relevant documents, c_t: the probability of occurrence of term t in the whole document collection, w_t: the weight to be assigned to term t, EMIM: expected mutual information measure, F4: F4MODIFIED, IDF: inverse document frequency, KLD:

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

These are examples of weighting schemes that have been considered for pseudo-relevance feedback.

1.2.6.2 Global Query Expansion

Query is expanded using a global, *query-independent* resource

- Manually edited thesaurus
- Automatically extracted thesaurus
- Query logs

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Global methods for expanding user queries can rely on a variety of resources. These may be thesauri that are manually constructed or automatically derived (a thesaurus is a database that contains words and their synonyms), or the automated analysis of query logs.

Manually Created Thesaurus

Expensive to create and maintain

- Used mainly in science and engineering

Example: Pubmed

https://www.ncbi.nlm.nih.gov/pubmed/?term=cancer

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Performing query expansion using a manually thesaurus requires the (expensive) effort of creating and maintaining such a thesaurus. This task is mainly performed in highly specialized technical fields in science and engineering. One prominent example of such a Thesaurus is maintained by Pubmed, the biggest publication database for medical literature maintained by the NIH, the National Institute of Health in the US. When using its search engine, you will find a window ”Search details” that shows how the user query is automatically expanded using the Pubmed thesaurus. In this example we see that the search system identifies that “cancer” is an entry on the concept “neoplasms”, and thus extends the query with all entries that it finds associated in the thesaurus (e.g., it would also search for “tumor”).

Automatic Thesaurus Generation

Generate a thesaurus automatically by analyzing the distribution of words in documents

- Problem: find words with similar meaning (synonyms)

Approach 1: Two words are similar if they **co-occur** with similar words
"switzerland" ≈ "austria" because both occur with words such as "national", "election", "soccer" etc., so they must be similar.

Approach 2: Two words are similar if they occur in the same **text pattern**
"live in *", "travel to *", "size of *" are all phrases in which both "switzerland" or "austria" can occur

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to avoid the effort of manually creating a thesaurus one can find methods to create it automatically by studying large numbers of documents and the distribution of words in those. This leads to the concept of word similarity. There exists two basic methods to study this similarity, either statistically, by observing which words occur together in documents, or in a more accurate way by identifying whether the words occur in the same text patterns.

We will study later in the lecture such methods in more detail. For the first approach, we will study so-called "word embeddings". For the second approach, we will learn about this type of methods in "information extraction".

Expansion using Query Logs

Query logs are an important resource for query expansion with search engines

- Exploit correlations in user sessions

Example 1: users extend query

- After searching “Obama”, users search “Obama president”
- Therefore, ”president” might be a good expansion

Example 2: users refer to same result

- User A accesses URL epfl.ch after searching “Aebischer”
- User B accesses URL epfl.ch after searching “Vetterli”
- “Vetterli” might be a potential expansion for the query “Aebischer”

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Query logs contain potentially rich information for query expansion. There are different ways of how such knowledge can be exploited. We show here two possibilities.

Other methods rely on mining query logs using various techniques, including clustering and association rule mining, that we will introduce later in the lecture in the part on “data mining”.

References

Papers

- Rocchio, J. (1971). Relevance feedback in information retrieval. The Smart retrieval system-experiments in automatic document processing, 313-323.
- Ponte, Jay Michael, and W. Bruce Croft. "A language modeling approach to information retrieval." PhD diss., University of Massachusetts at Amherst, 1998.
- Yoo, S., & Choi, J. (2011). Evaluation of term ranking algorithms for pseudo-relevance feedback in MEDLINE retrieval. Healthcare informatics research, 17(2), 120-130.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.3 EMBEDDING TECHNIQUES

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.3.1 Latent Semantic Indexing

Vector space retrieval is vague and noisy

- Based on index terms
- Unrelated documents might be included in the answer set
  - apple (company) vs. apple (fruit)
- Relevant documents that do not contain at least one index term are not retrieved
  - car vs. automobile

Observation

- The user information need is more related to concepts and ideas than to index terms

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Despite its success and widespread use, the vector space retrieval model suffers from some problems. If the user misses relevant terms in the query, documents that might contain semantically related terms, i.e., terms with the same meaning, will be missed. Also, in the case of terms with multiple meanings irrelevant documents can be returned.

The important insight is that terms may indicate a concept a user is interested in, but there does not necessarily exist a one-to-one correspondence between terms and concepts. As a consequence, retrieval results may contain irrelevant documents, and relevant documents may be missed.

The Problem

Vector Space Retrieval handles poorly the following two situations

1. *Synonymy*: different terms refer to the same concept, e.g., car and automobile
   - Result: poor recall
2. *Homonymy*: the same term may have different meanings, e.g., apple, model, bank
   - Result: poor precision

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

These problems are related to the fact that the same concepts can be expressed through many different terms (synonyms) and that the same term may have multiple meanings (homonyms). Studies show that different users use the same keywords for expressing the same concepts only 20% of the time.

Let's illustrate the problem resulting from synonyms and homonyms by an example. Among these three documents at the level of terms doc1 and doc2 are highly related, whereas doc3 has no similarity with the other two documents. Understanding the meaning of the words, it is however easy to see that in reality doc2 and doc3 are closely related, as they talk about mobile communications, whereas doc1 is completely unrelated to the others as it is about health and nutrition.

Key Idea

Map documents and queries into a lower-dimensional space composed of higher-level concepts

- Each concept represented by a combination of terms
- Fewer concepts than terms
- e.g., vehicle = {car, automobile, wheels, auto, sportscar}

Dimensionality reduction

- Retrieval (and clustering) in a reduced concept space might be superior to retrieval in the high-dimensional space of index terms

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Thus, it would be interesting to represent the meaning of documents and queries in information retrieval models on concepts, instead on terms. To that end, it is first necessary to define a "concept space" to which documents and queries are mapped, and compute similarity within that concept space. The concept space should ideally have much lower dimension than the term space, whose dimensionality is determined by the size of the vocabulary.

This figure illustrates the approach: rather than directly relating documents d and terms t, as in vector space retrieval, there exists an intermediate layer of concepts c to which both queries and documents are mapped. The concept space can be of a smaller dimension than the term space. In this small example we can imagine that the terms t1 and t2 are synonyms and thus related to the same concept c1. If now a query t2 is posed, in the standard vector space retrieval model only the document d1 would be returned, as it contains the term t2. By using the intermediate concept layer the query t2 would also return the document d2.

Applying this idea to our example before, we can consider a concept space consisting of four concepts, two related to health, two related to mobile communication. For doc2 and doc3 we recognize much better the close conceptual relationship the two documents have.

Similarity Computation in Concept Space

Concept represented by terms, e.g.

device = {iOS, iPad, RIM, mobile, handy, tablet, apple, blackberry}

Document represented by concept vector, counting number of concept terms, e.g.

$doc_1 = (4, 3, 3, 1)$
$doc_3 = (0, 0, 5, 2)$

Similarity computed by scalar product of normalized concept vectors

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We may consider concepts as being represented by sets of terms, and documents by concept vectors that count how many concept terms occur in the document. Using this approach we would obtain the non-normalized concept vectors doc1 = (4,3,3,1), doc2=(3,1,3,3) and doc3=(0,0,5,2).

After normalizing these vectors we compute the cosine similarities among the resulting concept vectors and obtain:

Sim (2,3) = 0.3

Sim (1,2) = 0.245

Sim (1,3) = 0.22

This result shows that indeed documents 2 and 3 are semantically closer, though still some confusion remains due to the large number of homonyms occurring in these documents.

Basic Definitions

Problem: how to identify and compute "concepts" ?

Consider the term-document matrix

- Let M_{ij} be a term-document matrix with m rows (terms) and n columns (documents)
- To each element of this matrix is assigned a weight w_{ij} associated with t_i and d_j
- The weight w_{ij} can be based on a tf-idf weighting scheme

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The challenge is to find a method that identifies and characterizes important concepts in document collections. One approach would be to perform this task manually, e.g. by using a predefined ontology and let users annotate documents using terms of the ontology. This is an approach that his been used in libraries, but is labour intensive. Thus we will now present a method that performs the task of concept identification and document classification by concepts automatically. Starting point for the method is the term-document matrix that we have introduced for vector space retrieval with weights based on a tf-idf weighting scheme.

We can interpret the process of producing a retrieval result in vector space retrieval as a matrix operation. This is illustrated in this figure. The ranking results from computing the product of a query vector q with the term-document matrix M. We assume that all columns in M and q are normalized to 1.

In vector space retrieval each row of the matrix M corresponds to

A. A document
B. A concept
C. A query
D. A term

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One way to understand of how concepts can be extracted from a document collection is to consider the effect of the term-document matrix on data. If we apply this matrix to a (high-dimensional) unit ball it will distort this ball into an ellipsoid. This ellipsoid will have one direction with the strongest distortion. We may think of this direction as corresponding to a particularly important concept in the document collection.

Singular Value Decomposition (SVD)

Represent Matrix M as $M = K.S.Dt$

- K and D are matrices with orthonormal columns
  $$K.Kt = I = D.Dt$$
- S is an $r \times r$ diagonal matrix of the singular values sorted in decreasing order where $r = \min(m, n)$, i.e., the rank of M
- Such a decomposition always exists and is unique (up to sign)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To identify the principal direction of distortion induced by the matrix transformation, a standard tool from linear algebra can be used, the singular value decomposition (SVD). SVD decomposes a matrix into the product of three matrices. The middle matrix S is a diagonal matrix, where the elements of this matrix are the singular values of the matrix M.

Construction of SVD

K is the matrix of eigenvectors derived from $M.Mt$

D is the matrix of eigenvectors derived from $M^t.M$

Algorithms for constructing the SVD of a $m \times n$ matrix have complexity $O(n^3)$ if $m \leq n$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Formally the SVD can be computed by constructing Eigenvectors of matrices derived from the original matrix M. This computation can be performed in time O(n^3). Note that the time complexity is considerable for large matrices, which makes the approach computationally expensive. There exist however also approximation and randomized techniques to perform this decomposition more efficiently (e.g. https://research.fb.com/blog/2014/09/fast-randomized-svd/).

Interpretation of SVD

We can write $M = K.S.Dt$ as sum of outer vector products

$$M = \sum_{i=1}^{r} s_i k_i \otimes d_i^t$$

The s_i are ordered in decreasing size

By taking only the largest ones we obtain a «good» approximation of M (least square approximation)

The singular values s_i are the lengths of the semi-axes of the hyperellipsoid E defined by

$$E = \left\{ Mx \mid \|x\|_2 = 1 \right\}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One way to understand of how the SVD extracts the important concepts from the term-document matrix is the following: the decomposition can be used to rewrite the original matrix as the sum of components that are weighted by the singular values. Thus we can obtain approximations of the matrix by only considering the larger singular values. The SVD after eliminating less important dimensions (smaller singular values) can be interpreted as a least square approximation to the original matrix. The symbol ⊗ denotes the **outer product** of two vectors, d_i is the i-th row of D. The outer product of vectors u =. (u_1,…,u_m) and v = (v_1,…,v_n) is given by the mxn matrix m with entries m_ij = u_i v_j.

The singular values have also a geometric interpretation, as they tell us how a unit ball (||x||=1) is distorted when the linear transformation defined by the matrix M is applied to it. We can interpret the axes of the hyperellipsoid E as the dimensions of the concept space.

This figure illustrates the structure of the matrices generated by the SVD. In general, m <= n, i.e., the number of documents is larger than the size of the vocabulary. The rows in K can then be interpreted as the representation of terms in the concept space, and the rows in D as the representation of documents in the concept space.

We can also see of how the concepts are represented by terms respectively documents. In particular, each concepts is now described as a vector of weighted terms.

Latent Semantic Indexing (LSI)

In the matrix S, select only the s largest singular values

- Keep the corresponding columns in K and D

The resultant matrix is called M_s and is given by

- $M_s = K_s . S_s . D_s^t$ where $s, s < r$, is the dimensionality of the concept space

The parameter s should be

- large enough to allow fitting the characteristics of the data
- small enough to filter out the non-relevant representational details

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Using the singular value decomposition, we can now derive an "approximation" of M by retaining only the s largest singular values in matrix S. The choice of s determines on how many of the "important concepts" the ranking will be based on. The assumption is that concepts with small singular value in S are rather to be considered as "noise" and thus can be dismissed. The resulting method is called Latent Semantic Indexing (LSI).

This figure illustrates the structure of the matrices after reducing the dimensionality of the concept space to s, when only the first s singular values are kept for the computation of the ranking. The rows in matrix K_s correspond to term vectors, whereas the columns in matrix D_s^t correspond to document vectors. By using the cosine similarity measure between columns of matrix D_s^t a similarity measure of documents can be computed.

Answering Queries

Documents can be compared by computing cosine similarity in the concept space, i.e., comparing their columns $(D_s{}^t)_i$ and $(D_s{}^t)_j$ in matrix $D_s{}^t$

A query q is treated like one further document

- it is added as an additional column to matrix M
- the same transformation is applied to this column as for mapping M to D

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

After performing the SVD, the similarity of different documents can be determined by computing the cosine similarity measure among their representation in the concept space (the columns of matrix $D_s{}^t$). Queries are considered like documents that are added to the document collection. For answering queries, the query is treated like a document, and its representation in the concept space is used to compute the similarity to documents.

Mapping Queries

Mapping of M to D

$$M = K.S.Dt$$
$$S^{-1}.K^{t}.M = D^{t} \quad \text{(since } K.Kt = 1\text{)}$$
$$D = M^{t}.K.S^{-1}$$

Apply same transformation to q:

$$q^{*} = q^{t}.K_{s}.S^{-1}$$

Then compare transformed vector by using the standard cosine measure

$$sim(q^{*}, d_i) = \frac{q^{*} \bullet (D_s^t)_i}{|q^{*}| |(D_s^t)_i|}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The transformation of queries into concept vectors works as follows: when a new column (the query) is added to M, we have to apply the same transformation to this new column, as to the other columns of M, in order to produce the corresponding column in the matrix D_s^t, representing documents in the concept space. We exploit the fact that $K_s^t.K_s = 1$.

Since $M_s = K_s.S_s.D_s^t$ we obtain $S_s^{-1}.K_s^t.M_s = D_s^t$ or $D_s = M_s^t.K_s.S_s^{-1}$.

This transformation is applied to the query vector q to obtain a query vector q* in the concept space. After that step, the similarity of the query to the documents in the concept space can be computed. ($(D_s^t)_i$ denotes the i-th column of matrix D_s^t)

This figure illustrates of how a query vector is treated like an additional document vector.

Example: Documents

B1 A Course on Integral Equations
B2 Attractors for Semigroups and Evolution Equations
B3 Automatic Differentiation of Algorithms: Theory, Implementation, and Application
B4 Geometrical Aspects of Partial Differential Equations
B5 Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra
B6 Introduction to Hamiltonian Dynamical Systems and the N-Body Problem
B7 Knapsack Problems: Algorithms and Computer Implementations
B8 Methods of Solving Singular Systems of Ordinary Differential Equations
B9 Nonlinear Systems
B10 Ordinary Differential Equations
B11 Oscillation Theory for Neutral Differential Equations with Delay
B12 Oscillation Theory of Delay Differential Equations
B13 Pseudodifferential Operators and Nonlinear Partial Differential Equations
B14 Sinc Methods for Quadrature and Differential Equations
B15 Stability of Stochastic Differential Equations with Respect to Semi-Martingales
B16 The Boundary Integral Approach to Static and Dynamic Contact Problems
B17 The Double Mellin-Barnes Type Integrals and Their Applications to Convolution Theory

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an example of a (simple) document collection that we will use in the following as running example.

Implementation in Python

```python
from sklearn.feature_extraction.text import TfidfVectorizer

tf = TfidfVectorizer(analyzer='word', ngram_range=(1,1), min_df = 2, stop_words = 'english')
features = tf.fit_transform(titles)
M = np.transpose(np.array(features.todense()))
```

```python
# compute SVD
K, S, Dt = np.linalg.svd(M, full_matrices=False)

# LSI select dimensions
K_sel = K[:,0:2]
S_sel = np.diag(S)[0:2,0:2]
Dt_sel = Dt[0:2,:]
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given Python libraries that compute the SVD, we can implement the LSI method directly using the definitions and based on the term-document matrix that we have already computed for the vector space retrieval model.

Results (s=2)

```
K_sel
```

```
array([[ 0.01781272, -0.4729881 ],
       [ 0.03264057, -0.43230378],
       [ 0.15088442, -0.17568951],
       [ 0.55589867,  0.07082109],
       [ 0.6843092 ,  0.1075997 ],
       [ 0.01570413, -0.37133288],
       [ 0.09073864, -0.07173948],
       [ 0.01775573, -0.20943739],
       [ 0.19758761,  0.08201858],
       [ 0.11060875,  0.05205271],
       [ 0.19758761,  0.08201858],
       [ 0.15088442, -0.17568951],
       [ 0.20802226,  0.09313466],
       [ 0.01555703, -0.22913745],
       [ 0.10330872, -0.00853892],
       [ 0.14994428, -0.49674497]])
```

```
S_sel
```

```
array([[2.12109044, 0.        ],
       [0.        , 1.50037763]])
```

```
np.transpose(Dt_sel)
```

```
array([[ 0.18982901, -0.0083562 ],
       [ 0.32262142,  0.07171508],
       [ 0.04727092, -0.58437076],
       [ 0.33399497,  0.1003478 ],
       [ 0.01183895, -0.31440669],
       [ 0.03875274, -0.10771362],
       [ 0.01329859, -0.40782546],
       [ 0.3018997 ,  0.09205811],
       [ 0.07134057,  0.02202618],
       [ 0.33016936,  0.09458633],
       [ 0.29162009, -0.24019296],
       [ 0.29162009, -0.24019296],
       [ 0.29566823,  0.10051203],
       [ 0.33016936,  0.09458633],
       [ 0.40993831,  0.08283115],
       [ 0.03543573, -0.14179904],
       [ 0.05664377, -0.43260133]])
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Her we inspect the resulting matrices, when selecting s=2 dimensions.

Since in our example the concept space has two dimensions, we can plot both the documents and the terms in this 2-dimensional space. It is interesting to observe of how semantically "close" terms and documents cluster in the same regions. This illustrates very well the potential of latent semantic indexing in revealing the « essential concepts » in document collections.

Applying SVD to a term-document matrix M. Each concept is represented in K

A. as a singular value
B. as a linear combination of terms of the vocabulary
C. as a linear combination of documents in the document collection
D. as a least squares approximation of the matrix M

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The number of term vectors in the matrix K_s used for LSI

A. Is smaller than the number of rows in the matrix M
B. Is the same as the number of rows in the matrix M
C. Is larger than the number of rows in the matrix M

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A query transformed into the concept space for LSI has …

A. s components (number of singular values)
B. m components (size of vocabulary)
C. n components (number of documents)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Discussion of Latent Semantic Indexing

Latent semantic indexing provides an interesting conceptualization of the IR problem

Advantages

- It allows reducing the complexity of the underlying concept representation
- Facilitates interfacing with the user

Disadvantages

- Computationally expensive
- Poor statistical explanation

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

LSI has been a fundamental advance in the development of informational retrieval. However, it also suffers from conceptual problems.

For example, least squares approximation is based on the assumption that term frequencies are normally distributed. This is not consistent with the observation that term frequencies are power-law distributed.

Alternative Techniques

Probabilistic Latent Semantic Analysis

- Based on Bayesian Networks

Latent Dirichlet Allocation

- Based on Dirichlet Distribution
- State-of-the-art method for concept extraction

Same objective of creating a lower-dimensional concept space based on the term-document matrix

- Better explained mathematical foundation
- Better experimental results

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

LSI has triggered the search for different alternative approaches for better capturing the semantic properties of text in information retrieval.

One recent successful approach is LDA, based on the use of Dirichlet distributions. Like LSI it generates a concept space, where concepts are represented as term vectors. The method has better theoretical foundations and empirically it produces better results. It is nowadays considered as a golden standard. The approach is mathematically more involved than LSI, and therefore we will not be able to develop this method in this lecture.

This illustration shows the basic idea underlying LDA: it is assumed that there exist topics that are represented as mixtures of words. In the example we see words that are related to two meanings of “bank”, the financial institution and the river bank, and are represented by different related works. Then, documents are assumed to be generated from a mixture of topics, where the mixture is determined by weights, as indicated in the figure. As a result, each document contains terms form its associated topics in the right proportion.

Like the probabilistic language model used in information retrieval, this approach us an example of a probabilistic generative model.

Document Generation using a Probabilistic Process

For each document, choose a mixture of topics

For every word position, sample a topic from the topic mixture

For every word position, sample a word from the chosen topic

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given the topic model, we obtain a probabilistic process for generating documents.

LDA: Topic Identification

Approach: Inverting the process: given a document collection, reconstruct the topic model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

While it is straightforward to generate documents, once the probabilistic topic model is given, deriving a topic model for a given document collection is a hard problem. To solve this problem, a probabilistic topic model needs to be derived, that is likely to generate the document collection under consideration.

Latent Dirichlet Allocation

Topics are **interpretable** unlike the arbitrary dimensions of LSI

- Topic and word distributions are Dirichlet distributions
- Construction of topic model is mathematically involved, but computationally feasible
- Considered as a state-of-the art method for topic identification

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We do not present the details of LDA here, as the method is mathematically fairly involved. However, it is important to note that it is considered today as a state-of-the-art method of topic detection.

Use of Topic Models

Unsupervised Learning of topics

- Understanding main topics of a topic collection
- Organizing the document collection

Use for document retrieval: use topic vectors instead of term vectors to represent documents and queries

Document classification (Supervised Learning): use topics as features

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Topic models provide a representation of documents at a conceptual level. Therefore, they are applied in many different contexts, including document retrieval and classification.

This illustration summarizes the connection between the vector space model, which we introduced for information retrieval and topic models that produce low-dimensional (dense) representation of documents.

1.3.3 Word Embeddings

The neighborhood of a word expresses a lot about its meaning

- "You shall know a word by the company it keeps" (J. R. Firth 1957)

government debt problems turning into banking crises as has happened in

saying that Europe needs unified banking regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With latent semantic indexing we have exploited the idea that words that occur together in a document are likely to have some shared meaning. The idea that words occurring in a common context have a shared meaning is quite old and has been already stated by Firth in 1957.

Different to LSI we will in the following exploit this idea in different way. Instead of considering a complete document as the context of a word, we will consider only the immediate neighborhood of a word, in the spirit of the statement of Firth.

Word Context

government debt problems turning into banking crises as has happened in

saying that Europe needs unified banking regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

Context: words in a window of size s, e.g., s = 2

Example:

- Center word: w = banking
- Context 1: $C_1(w)$ = {turning, into, crises, as}
- Context 2: $C_2(w)$ = {needs, unified, regulation, to}

Word-Context occurrence: $(w, c), c \in C_i(w)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For a given word w we can identify its neighboring words, which will call its context and denote by C(w). The context can be determined by choosing a window of preceding and succeeding words. A typical window size used in practice would by n = 5. We will call the word w in the following center word and a word c a context word.

Since a word can occur multiple times, each occurrence of the word induces a new context. We will denote these contexts by $C_i(w)$ with an index i running over all occurrences of a word in a document collection.

Similarity-Based Representation

One of the most successful ideas in natural language processing

Two words are considered as similar,
when they have similar contexts

- Context captures both syntactic and semantic similarity
  - Syntactic: e.g., king – kings
  - Semantic: e.g., king – queen
- Implicitly used with the term-document matrix M
  - Less localized context
  - No distinction between roles of context and word

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The idea of context representing meaning is one of the most successful ideas in natural language processing. The neighborhood captures not only semantic relationships among words (as would co-occurrence in the same document). It, at the same time, also captured syntactic relationships. This is an important difference compared to methods based on document-level co-occurrence like LSI.

A second important difference is that methods based on this idea distinguish between a word occurring as the center word of a context and as a context word. For example, it is very unlikely the word “king” would have in its context a second occurrence of the word “king”. Thus, king as a “context word” needs to be treated differently from king as a center word. This distinction is not made in approaches based exclusively on co-occurrence statistics at the document level.

Idea: Word Embeddings

Model how likely a word and a context occur together

Approach:

- Map words into a low-dimensional space (e.g., d = 200)
- Map context words into the same low-dimensional space
  - A different mapping for the same words
- Interpret the vector distance (product) as a measure for how likely the word and its context occur together

Due to projection in low-dimensional space, semantically and syntactically related words and contexts should be close

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic idea for word embeddings is straightforward. Both, center words and context words, are mapped into a low-dimensional space of the same dimension. Their representations in that space should be similar, if the center word and the context word occur frequently together. Thus, we capture the information on word context in a low-dimensional representation. Using dimensionality reduction, the expectation is that words and contexts that play similar roles would also be close to each other and would capture syntactic and semantic similarity.

Dimensionality reduction is helpful, since vocabularies can become very large. This is especially true when not only words, but also short phrases are considered. Thus, data would be very sparse in the original vocabulary space, and it would be difficult to represent similarity.

Mapping to Concept Space

Vocabulary T of size m

Words $w_i \in T$ are encoded as "1-hot vectors" of length m, i.e., the component at position i is 1 all others are 0

Word vector w_i

$$\begin{array}{c} 0 \\ 0 \\ . \\ . \\ 1 \\ . \\ . \\ 0 \\ 0 \\ 0 \\ 0 \end{array} \quad \textbf{Position i}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We introduce now the representations of center words and context words in an m-dimensional space. The representation where the i-th component of a vector is set to 1 for the i-th word in the vocabulary is called often the 1-hot encoding of the word.

Model Parameters

Dimension of concept space d

The mapping to concept space is represented by matrices $W^{(w)}$ and $W^{(c)}$ of dimension $d \times m$

Mapping a word w_i

$$\boldsymbol{w}_i = W^{(w)} w_i \qquad \boldsymbol{w}_i \text{ is the column } i \text{ in } W^{(w)}$$

Mapping a context word c_i

$$\boldsymbol{c}_i = W^{(c)} w_i \qquad \boldsymbol{c}_i \text{ is the column } i \text{ in } W^{(c)}$$

Parameters of model θ : All coefficients of $W^{(w)}$ and $W^{(c)}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For creating a word embedding we first choose a dimension that is significantly lower than the size of the vocabulary. Typical values of the dimension are in the hundreds, whereas vocabularies can have millions of terms.

In order to map a 1-hot vector from the vocabulary space into the word embedding space, we apply linear mappings represented by a matrices $W^{(w)}$ and $W^{(c)}$. As mentioned before. the mappings for center words and context words are different.

Note that the i-th columns of the matrices correspond exactly to the word embedding representations of the i-th word in the vocabulary.

For notational convenience we will denote in the following the combined set of coefficients of the two matrices also by θ.

Here we illustrate the mapping of words into concept space graphically. Since the word vectors use the 1-hot encoding, we observe that the columns of the matrix $W^{(w)}$ in fact are exactly the representation of the center words in the concept space. The same holds also for the mapping of context words.

Question

A row of matrix $W^{(c)}$ represents

1. How relevant each word is for a dimension
2. How often a context word c co-occurs with all words
3. A representation of word c in concept space

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Learning the Model from Data

Given a document collection, how to obtain the parameters θ?

Formulate a classification problem

- Given a word-context pair, predict whether it occurs in the document collection

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The problem we face now, is how to determine the model parameters, given a document collection.

We can approach this by introducing learning problem. We want to be able to predict whether a given word-context pair occurs in the document collection. By solving this problem, we will obtain the embedding parameters.

Statistical Learning

We aim to learn a function

$$f: S \rightarrow R$$

We have training data:

samples $f(s_1), f(s_2), \ldots, f(s_t)$

We define a parametrized function (model)

$$f_\theta: S \rightarrow R$$

Learning is obtaining good parameters θ

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Before we introduce in detail, how to obtain the parameters for the word embedding, let's first recall the principles of the approach we are going to use, statistical learning.

The general setting of statistical learning is to obtain an approximation of a given function f, from some sample values of the function that are known. The function f is approximated by a function f_θ that has a given form and contains a set of parameters θ. By optimizing the parameters θ we obtain the function approximation. The optimization step is what is generally called learning.

A standard example of statistical learning is linear regression.

Word Context Learning Problem

Domain S	$S = \{(w,c) \mid w,c) \in T \times T\}$ w, c are word-context pairs
$f: S \rightarrow R$	$f(w,c) = P(w,c)$ the probability for a given word-context pair to occur, thus $R = [0,1]$
$f(s_1), f(s_2), \ldots, f(s_t)$	$f(w,c) = 1$ iff (w,c) occurs in text, otherwise $f(w,c) = 0$
$f_\theta(w,c) = P_\theta(w,c)$	approximation of $P(w,c)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given that we want to predict whether a word-context pair occurs in a document collection, the statistical learning problem we formulate is for a function that takes as input word-context pairs and returns as output a probability for that this pair occurs in the documents. The samples from which we learn are derived from the document collection. For word-context pairs that occur in the collection we set the function value to 1, and for any pair that does not occur, we can set it to 0. The problem is then to find a function f_θ that approximates well the original probability distribution P. We have now to determine what form the function f_θ can take, and how the quality of approximation is measured.

Representation Learning

A learning algorithm that creates for some input data a new representation with desirable properties

- Most often the representation is in a vector space
- For example, represent words of a vocabulary as vectors

Idea: create the word embeddings using representation learning

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

So far, the idea of using statistical learning does not explain how we would obtain word embedding representations. To this end, we introduce now another concept, which is called representation learning. Representation learning aims at creating a new representation of input data, into a space of interest and with useful properties. Most often, the space for representation is a vector space.

In order to obtain the representations as a by-product of solving a statistical learning problem, the “trick” is to create a function f_θ that is based on the desired representations of the input data, in our case on the vector representations of the words. As a result, the “real objective” of solving the previous learning problem is no longer doing good predictions for the function f, but obtaining the vector representations.

Word Embedding Learning Problem

Specify the form of the function f_θ

$$f_\theta(w, c) = \boldsymbol{f}(\boldsymbol{w}, \boldsymbol{c}) = P_\theta(w, c)$$

$\boldsymbol{w}, \boldsymbol{c}$ are the embedding vectors of w, c
$\boldsymbol{w} = W^{(w)}w, \boldsymbol{c} = W^{(c)}c$

$\boldsymbol{f}$ is some function that takes as input the embedding vectors

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The function f_θ we learn has a specific form: it first maps the words into a vector space, their representation, and then applies another function $\boldsymbol{f}$ for approximating the probabilities.

Note that the function$\boldsymbol{f}$ is using as input the vector representations of the words. The matrices $W^{(w)}$ and $W^{(c)}$ are therefore part of the parameters of the function f_θ.

Properties of Embeddings

The function we want to learn, is the probability $P_\theta(w, c)$ for the word w to appear with the context word c

Derive an approximation of this probability from the representations

Idea for $\boldsymbol{f}$

- require that if w and c are likely to co-occur their representations $\boldsymbol{w}$ and $\boldsymbol{c}$ should be similar, i.e., the scalar product $\boldsymbol{w} \cdot \boldsymbol{c}$ should be large

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

So far, we have not imposed any specific constraint on the function $\boldsymbol{f}$ that is applied to the embedding vectors. Now we come back to the requirement that low-dimensional representations of words should be similar, if they have a close semantic or syntactic relationship. If consider co-occurrence as a word-context pair as an indicator for such a relationship, we could also state the related requirement that the low-dimensional representations of center words and context words should be similar. This is the approach we will consider in the following.

For determining similarity of vector representations, we can rely on a well-known approach, the use of the scalar product.

Here we illustrate the embedding of words and context words in the same low-dimensional space. The idea is that the context words that typically co-occur together with a word, have similar embedding vectors, whereas others (like monkey, which is rarely occurring together with bank) have very different ones.

Deriving Probabilities

Normalized scalar product?

- Produces values in [-1,1]

Using the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$

$$f_\theta(w,c) = P_\theta(w,c) = \boldsymbol{f}(\boldsymbol{w},\boldsymbol{c}) = \sigma(\boldsymbol{c}\cdot\boldsymbol{w}) = \frac{1}{1+e^{-\boldsymbol{c}\cdot\boldsymbol{w}}}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Since we have formulated the learning problem as a prediction of probabilities, the range of values of the learnt function f_θ should fall into the interval [0,1]. Using the scalar product, even if we use normalize by using cosine similarity, does not achieve this.

Therefore, a common function to map real values into the interval [0,1] is used, the sigmoid function. The sigmoid function has other useful properties, in particular, it is differentiable.

This results in the final structure of the function f_θ we learn. It is fully specified by its parameters θ.

Illustration

Banking • Crisis = +3
Banking • Debt= +2
Banking • Monkey = -1

P(Banking, Crisis) = 0.95
P(Banking, Debt= 0.88
P(Banking, Monkey) = 0.27

```
import numpy as np
def sigmoid(z):
    return 1/(1 + np.exp(-z))
[sigmoid(3),sigmoid(2),sigmoid(-1)]
```

```
[0.9525741268224334, 0.8807970779778823, 0.2689414213699951]
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we illustrate how by applying the softmax function to the scalar product of the embedding vectors results in probabilities predicting the occurrence of word-context pairs.

Question

Which of the following functions is different from the three others?

1. $f(w, c)$
2. $f_\theta(w, c)$
3. $\boldsymbol{f}(\boldsymbol{w}, \boldsymbol{c})$
4. $\sigma(\boldsymbol{c} \cdot \boldsymbol{w})$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Learning the Parameters

Assume we have positive examples D for (w, c), as well as negative examples $\widetilde{D}$ for (w, c) that are not occurring in the document collection

Maximizing the overall probabilities

$$\theta = \underset{\theta}{\operatorname{argmax}} \prod_{(w,c)\epsilon D} P_\theta(w, c) \prod_{(w,c)\epsilon \widetilde{D}} \left(1 - P_\theta(w, c)\right) =$$

$$\underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\epsilon D} \log \sigma(\boldsymbol{c} \cdot \boldsymbol{w}) + \sum_{(w,c)\epsilon \widetilde{D}} \log \sigma(-\boldsymbol{c} \cdot \boldsymbol{w})$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Now that we have defined the structure of the function we want to learn, we must perform the learning. To that end, we formulate an optimization problem which states that learning consists of finding the optimal parameters θ for achieving the highest predicted probability for the available data. The available data consists of samples of word-context pairs. There exist two types of such samples, first, word-context pairs that occur in the text collection and thus must have a high probability to occur, and word-context pairs that do not occur in the document collection and thus must have a low probability. We cannot assume that we achieve perfect prediction of those probabilities, i.e., values of 0 and 1, under the specific choice of the function P_θ we decided to learn. Therefore, we try to maximize the product of the probabilities of the positive samples and the complement of the probabilities of the negative samples.

The optimization problem can be reformulated by taking the logarithm of the expression to be optimized. This is a standard operation to make the optimization problem easier to treat from an algebraic and numerical perspective.

Positive and Negative Samples

Let D be the set of all word-context occurrences

$$(w_t, c_t) \in D, t = 1, .., s, |D| = s$$

For each (w_t, c_t) define a set $P_n(w_t)$ of K negative samples, i.e., word-context pairs not occurring in the collection

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we describe the approach for obtaining the positive and negative samples. We consider the set D to be the set of all word-context occurrences in the document collection. For each element of D we then choose a set of negative samples of a given size s. This set consists of word-context pairs that do not occur in the collection. We will describe a concrete approach for selecting such negative samples later.

Obtaining Negative Samples

Negative samples are taken from $P_n(w) = V \setminus C(w)$

Empirical approach

- If p_c is the probability of a candidate context word c in collection, choose the word with probability ${p_c}^b$ with $b < 1$, e.g., $b = 0.75$
- Less frequent words are sampled more often
- Practically: approximate the probability by sampling a few non-context words

Example

- If $p_c = 0.9$, then ${p_c}^{0.75} = 0.92$
- If $p_c = 0.01$, then ${p_c}^{0.75} = 0.032$, boosted by factor 3

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A concrete method to obtain the negative samples is described now. For a given word w from the vocabulary V, the context words for negative samples are chosen from all words that do not occur as context word of w. This could be done, for example, uniformly at random. However, with such choice high frequency words would have too much influence on the learning process, whereas low- frequency words would not be sufficiently considered. Therefore, based on empirical evaluation, it turns out that it is of advantage to favor low frequency words. The probability to choose a context word is thus biased towards low frequency words, by attenuating higher probabilities with an exponent $b < 1$. In this way, less frequent words will be considered more frequently.

In order to apply this approach, the word frequencies in the collection of words not occurring in the context need to be known. Since computing this statistics for every word separately would be expensive, in practice it is obtained by sampling.

Loss Function

Maximizing the overall probabilities is equivalent to minimizing the following function

$$J(\theta) = \frac{1}{s}\sum_{t=1}^{s} J_t(\theta)$$

$$J_t(\theta) = -\log\sigma(\boldsymbol{c_t}\cdot\boldsymbol{w_t}) - \sum_{(w_t,c_k)\in P_n(w_t)} \log\sigma(-\boldsymbol{c_k}\cdot\boldsymbol{w_t})$$

$J(\theta)$ is a **loss function**

- A function that measures how well the model fits the samples (training data)
- Learning is applying an algorithm that minimizes this loss function

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Having defined the set of positive and negative samples, we can further reformulate the optimization problem for learning the parameters θ. We convert the maximization problem to a minimization problem, which is the conventional formulation of learning problems, and replace the specific choice of samples. This allows to decompose the function to be optimized into a sum of functions $J_t(\theta)$, corresponding to the positive samples (w_t, c_t). This decomposition will be important in the following for the method used to solve the optimization problem.

According to the standard terminology in machine learning, $J(\theta)$ is called the loss function, and is a function that needs to be minimized in order to solve the optimization resp. learning problem.

Skipgram Model with Negative Sampling

The approach to derive word embeddings described is called the skipgram model

- $\boldsymbol{w} = W^{(w)}w, \boldsymbol{c} = W^{(c)}c$, embedding vectors
- $P_\theta(w, c)$, structure of function to represent probabilities of word-context occurrences
- $J(\theta)$, definition of loss function
- $P_n(w)$, choice of negative samples

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The different elements for defining an optimization problem to learn the embedding vectors that we have introduced correspond to one specific word embedding model which is called the skipgram model with negative sampling. Note the several choices that are made, in particular on the structure of the probability function, the specific loss function and the choice of negative samples.

Observations

1. The function P_θ can be learnt from a document corpus
2. We can apply optimization to learn the function
3. As we learn the parameters θ, implicitly by learning the function we learn a representation for words (the word embedding vectors)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We have not set up a formulation of an optimization problem that can be solved for any given document collection. By solving the optimization problem, we will obtain as a side effect the embedding vectors. This, in fact, has been the initial objective of defining the learning problem. What remains to be seen is of how concretely the optimization problem can be solved.

Question

From which data samples the embeddings are learnt?

1. Known embeddings for (w,c) pairs
2. Frequency of occurrences of (w,c) pairs in the document collection
3. Approximate probabilities of occurrences of (w,c) pairs
4. Presence or absence of (w,c) pairs in the document collection

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Question

With negative sampling a set of negative samples is created for

1. For each word of the vocabulary
2. For each word-context pair
3. For each occurrence of a word in the text
4. For each occurrence of a word-context pair in the text

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Question

The loss function is minimized

1. By modifying the word embedding vectors
2. By changing the sampling strategy for negative samples
3. By carefully choosing the positive samples
4. By sampling non-frequent word-context pairs more frequently

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Solving the Learning Problem

Gradient Descent: compute using all samples

$$\theta^{new} = \theta^{old} - \alpha \nabla_{\theta} J\,(\theta^{old})$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given the loss function, finding the optimal parameters θ can then be achieved by applying standard methods from machine learning. Concretely, θ can be determined using gradient descent, a standard method for searching the minima of a function using derivatives. Gradient descent is an iterative approach for locally improving a current minimum. By following the local gradient, the optimal value is incrementally updated till it converges to a (local) minimum.

Stochastic Gradient Descent (SGD)

For every $(w_t, c_t) \in D$, update θ separately

$$\theta^{new} = \theta^{old} - \alpha \nabla_\theta J_t(\theta^{old})$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In basic gradient descent, the parameters are updated based on the complete data (set of samples). In stochastic gradient descent, the parameters are updated for a randomly selected subset of the data. Therefore, the decomposition of the loss function into component functions corresponding to each element of the data was useful. In our case we update the parameters for each word-context pair separately. With stochastic gradient we reduce the cost of each iteration, but on the other hand slow down the convergence rate. In practice, using SGD is the standard method of learning with complex loss functions.

Computing the Derivatives

$$J_t(\theta) = -\log\big(\sigma(\boldsymbol{c}_t \cdot \boldsymbol{w}_t)\big) - \sum_{(w_t, c_k) \in P_n(w_t)} \log \sigma(-\boldsymbol{c}_k \cdot \boldsymbol{w}_t)$$

with $\theta = \boldsymbol{w}_1, \dots, \boldsymbol{w}_m, \boldsymbol{c}_1, \dots, \boldsymbol{c}_m$

$$\nabla_\theta J_t(\theta) = \left(\frac{\partial}{\partial \boldsymbol{w}_1} J_t, \dots \frac{\partial}{\partial \boldsymbol{w}_m} J_t, \frac{\partial}{\partial \boldsymbol{c}_1} J_t, \dots \frac{\partial}{\partial \boldsymbol{c}_m} J_t\right)$$

then $\frac{\partial}{\partial \boldsymbol{w}_i} J_t = 0$ and $\frac{\partial}{\partial \boldsymbol{c}_i} J_t = 0$, if $\boldsymbol{w}_i \neq \boldsymbol{w}_t$ and $\boldsymbol{c}_i \neq \boldsymbol{c}_t, \boldsymbol{c}_k$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When stochastic gradient descent is used, only columns of the matrices θ that contain the word, or a context word related to the current word-context pair and its negative samples need to be updated. This implies that the data must be organized that these rows can be efficiently accessed (e.g., using hashing).

Updating the Model Parameters

For J_t updates only affect columns in $W^{(w)}$ and $W^{(c)}$ that correspond to $\boldsymbol{w}_t$, $\boldsymbol{c}_t$ and $\boldsymbol{c}_k$,

And the updates are according to the corresponding partial derivative

e.g.,

$$\boldsymbol{w}_t^{new} = \boldsymbol{w}_t^{old} - \alpha \frac{\partial}{\partial \boldsymbol{w}_t} J_t(\boldsymbol{w}_t^{old}, \boldsymbol{c}_t^{old}, \boldsymbol{c}_1^{old}, \dots, \boldsymbol{c}_K^{old})$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The embedding vectors of a word are affected only by the derivatives of the loss function with respect to the word vector.

Computing the Derivative (Backpropagation)

Some notation

$$x = \mathbf{c} \cdot \boldsymbol{w}, \mathrm{p} = \sigma(x), z_k = -\,\mathbf{c}_k \cdot \boldsymbol{w},\ q_k = \sigma(z_k),\ |P_n(w)| = K$$

Then $$J_t(\boldsymbol{w}, \mathbf{c}, \mathbf{c}_1, \ldots, \mathbf{c}_k) = -\log(p) - \sum_{k=1}^{K} \log q_k$$

$$\frac{\partial J_t}{\partial \boldsymbol{w}} = \frac{\partial J_t}{\partial p}\frac{\partial p}{\partial x}\frac{\partial x}{\partial \boldsymbol{w}} + \sum_{k=1}^{K} \frac{\partial J_t}{\partial q_k}\ \frac{\partial q_k}{\partial z_k}\ \frac{\partial z_k}{\partial \boldsymbol{w}}$$

$$\frac{\partial J_t}{\partial p} = -\frac{1}{p}\ , \frac{\partial p}{\partial x} = \mathrm{p}(1-\mathrm{p})\,, \frac{\partial x}{\partial \boldsymbol{w}} = \mathbf{c}$$

Finally $$\frac{\partial J_t}{\partial \boldsymbol{w}} = -(1-\mathrm{p})\mathbf{c} + \sum_{k=1}^{K}(1-q_k)\,\mathbf{c}_k$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Computing the derivative of the loss function, for updating the model parameters, requires evaluating derivatives along the operators that compose the loss function. This step is part of what is in machine learning commonly called backpropagation. We illustrate the essential steps of this computation for computing the derivative for a given word $\frac{\partial J_t}{\partial \boldsymbol{w}}$. To that end, it is helpful to identify the different components that constitute the loss function and compose then the derivatives of those components using the chain rule. This reveals that the updates to the parameters result in simple operations. As an exercise one can complete the computation for the other derivatives.

.

Regularization

Regularization: limit model complexity

- In order to avoid overfitting
- In order the model to generalize well to new data

Possible approaches

- Limit the complexity of the model (e.g., linear)
- Add a regularization term to the loss function

Example:

$$J(\theta) = \sum_{t=1}^{s} \left(f(s_t) - f_\theta(s_t)\right)^2 + \gamma|\theta|_2$$

The skipgram model does not use regularization

- Is already a simple model, no overfitting

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One key problem in statistical learning (or machine learning) is overfitting. If we provide a large number of parameters θ we can easily learn a good, or even perfect approximation of the function applied to the training data, but the function will perform poorly on new data. Therefore, the idea is to “limit” the amount of information that can be stored in the model parameters, by minimizing some measure on them by including it into the loss function. This is called regularization. In the method we have introduced for deriving word embeddings no regularization is applied. The reason is that a linear model is nto considered to be a complex model and therefore the risk of overfitting is limited.

Question

A word embedding for given corpus …

1. depends only on the dimension d
2. depends on the dimension d and number of iterations in gradient descent
3. depends on the dimension d, number of iterations and chosen negative samples
4. there are further factors on which it depends

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

CBOW Model

Consider a pair $(w, \{c_1, \dots, c_n\})$, does it origin from the data?

Compute $\bar{\boldsymbol{c}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{c_i}$

$$P_\theta(w, \{c_1, \dots, c_n\}) = \frac{1}{1 + e^{-\bar{\boldsymbol{c}} \cdot \boldsymbol{w}}} = \sigma(\bar{\boldsymbol{c}} \cdot \boldsymbol{w})$$

Observation: CBOW produces better results for frequent terms, skipgram for rare terms

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With the skipgram model, we have considered the probability of word-context word pairs as the objective for learning. An alternative approach is to consider the complete context of a word and predict whether a set of context words is likely to occur with a given word. In order to represent a set of words in the embedding model we can choose the mean of the embedding vectors of the context words and require that this vector is similar to the embedding vector of center word, if this specific selection of context words appears with the center word in the document collection. The remaining aspects of the method remain the same. This alternative model is called CBOW. Empirical evaluations show that skipgram gives better representations for rare terms.

Result

Matrices $W^{(w)}$ and $W^{(c)}$ that capture information on word similarity

- Words appearing in similar contexts generate similar contexts and vice versa
- Hence, mapped to similar representations in lower dimensional space
- Use $W = W^{(w)} + W^{(c)}$ as the low-dimensional representation

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

By using skipgram or CBOW we obtain two models mapping words into a low dimensional space corresponding to the two roles words can take, center words or context words. In practice, a final model is derived as the sum of the two matrices.

Alternative Formulation of the Problem

Consider as prediction problem

- Given a word, predict whether another word is a context word (skipgram)
- Given a context word, predict whether a word relates to this context (CBOW)

$$\theta = \underset{\theta}{\operatorname{argmax}} \prod_{w \epsilon V} \prod_{c \in C(w)} P_\theta(w|c) =$$

$$\underset{\theta}{\operatorname{argmax}} \sum_{w \epsilon V} \sum_{c \in C(w)} \log P_\theta(w|c)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The original formulation of the skipgram and CBOW models used a different approach. Instead of predicting the independent probabilities of occurrences of word-context pairs, it models the conditional probability of having a context word given a word (skipgram), respectively of having a word given a context word (CBOW). Using these conditional probabilities, a different optimization objective can be formulated, optimizing the observed conditional probabilities of word-context word occurrences.

Deriving Probabilities

Normalized scalar product?

- Produces values in [-1,1]
- $\sum_{i=1}^{m} P_\theta(w_i|c)$ does not add up to 1

Standard approach to convert a set of values into probabilities: **softmax function**

Values: $\boldsymbol{w}_i \cdot \boldsymbol{c}$ for $i = 1, \ldots, m$

$$P_\theta(w_i|c) = \frac{e^{\boldsymbol{w}_i \cdot \boldsymbol{c}}}{\sum_{j=1}^{m} e^{\boldsymbol{w}_j \cdot \boldsymbol{c}}}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For introducing a function that derives those conditional probabilities from the vector representation of words while using the scalar product, in addition to the problem that the values of the scalar product do not fall into the interval [-1, 1], we find the additional problem that the conditional probabilities for a given context word (CBOW) do not add up to 1.

To address this issue, there exists a standard approach to convert an (arbitrary) set of values into a probability distribution. First, applying the exponential function to the set of values (in our case the scalar products), produces positive values only, and second normalizing by the sum of the exponentials lets the values fall into the interval [0,1].

Illustration

Banking • Crisis = +3
Banking • Debt= +2
Banking • Monkey = -1

P(Banking, Crisis) = 0.72
P(Banking, Debt= 0.27
P(Banking, Monkey) = 0.01

```
from scipy.special import softmax
arr = [3,2,-1]
softmax(arr, axis=0)
```

```
array([0.72139918, 0.26538793, 0.01321289])
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Applying the softmax function to the scalar products, produces now a probability distribution for a context word, co-occurring with some word of the vocabulary.

Hierarchical Softmax

Issue: computing $P_\theta(w|c)$ is expensive

- In each iteration of the gradient descent, for computing the sum the algorithm iterates over the complete vocabulary

Solution: Hierarchical softmax

- Compute approximate softmax function using a tree over the vocabulary

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The issue with this variant of the loss function is that in the gradient descent the sums of the conditional probabilities $\sum_{i=1}^{m} P_\theta(w_i|c)$ have to be computed repeatedly which is prohibitively expensive. The solution to this problem is an approach called hierarchical softmax which performs an approximate computation of those sums using a tree-structure to organize the vocabulary.

1.3.4 Fasttext

Fasttext introduces the idea of using word n-grams (phrases) and **subword embeddings**

- Build embeddings for character n-grams
- Enable processing of unseen words

Example (in French): mangerai

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The word embedding models we have presented where based on the assumption that the vocabulary consists of the words found in the text corpus. Fasttext is a variant of word embeddings that has been developed by Facebook research. It incorporates two additional ideas:

- The use of word n-grams, respective phrases: often phrases carry a different meaning than their constituent words. Considering phrases (that are sufficiently frequent) can help to produce better representations of text and capture meaning that cannot be assigned to the constituents of the phrase. As an extreme example one might consider the phrase “The Who” which designates a rock band, whereas the constituent words have no meaning that would be related to music in any way.
- The use of character n-grams: many languages with a writing system based on an alphabet, e.g., Latin languages, have word variations with similar meanings resulting from grammatical rules. Often some variations might not occur in a training corpus. By using so-called subword embeddings the meaning identified for seen words, can so be transferred also to unseen works. This also works across languages, as often languages inherit expressions from foreign languages, or words are same or similar across different languages.

Subword Embeddings

For a given word create representations for all of its subwords S_w, including the word itself

- Add special characters at the start and end of the word
- Create all n-grams

Example: S_w for word "mangerai", n=3

- S_w = {_mangerai_, _ma, man, ang, nge, ger, era, rai, ai_}

Representation of w: $\boldsymbol{w} = \sum_{s \in S_w} \boldsymbol{s}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When using subword embeddings, the representation of a word is aggregated from the representation of its constituent n-grams, by adding up the n-gram representations.

Byte Pair Encoding (BPE)

Subword embeddings potentially generate large numbers of possible tokens, many of which are not useful

- Large vocabulary implies training the model becomes expensive

BPE allows to create smaller vocabularies, while capturing essential subword information

- Statistically analyze text for frequent character sequences
- Replace frequent sequences by tokens

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When using subword embeddings with character n-grams one faces the problem that the number of tokens, i.e., the size of the vocabulary, may become very large when all character n–grams occurring in the text are used. This in turn makes training the models very expensive. In addition, many of the tokens are rare and potentially not very useful. In order to address this problem, a method called byte pair encoding is used in order to obtain smaller vocabularies and select only frequent subwords. For this method, first the text needs to be analyzed to determine the frequent tokens.

Example

"Do Do! Do! Do! Do? Do! Done Done! Done? Done."

What are good tokens?

- All subwords of size n=2,3,4,5
  {Do, o!, o?, o., on, ne, e!, e?, e., Do!, Do?, ...}
- Better solution
  {Do, Do!, Done, !, ?, .}
  Tokenized text:
  [Do, Do!, Do!, Do!, Do, ?, Do!, ., Done, Done, !, Done, ?, Done, .]

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates the idea of BPE. Instead of blindly selecting all character n-grams, a carefully selected set of frequent n-grams is chosen and used to tokenize the text.

BPE Algorithm

Add word boundaries

"Do_Do!_Do!_Do!_Do?_Do!_Done_Done!_Done?_Done."

```
Initial Vocabulary V = all characters
while i < max_iterations
     determine a most frequent pair of tokens
     add pair to V as a new token
     update token frequencies
     i = i+1
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For running the BPW algorithm first a special token for word boundaries is added (in this example _).

The BPE algorithm proceeds iteratively. It determines in every iteration a most frequent pair of tokens and creates from this pair a new token. Then token frequencies are updated with the new set of tokens. The algorithm terminates after a predetermined number of tokens has been created.

In order to determine token frequencies without repeatedly scanning the text, the algorithm can initially scan the document collection to determine all n-gram frequencies (up to a given size).

BPE Example

Do_Do!_Do!_Do!_Do?_Do!_Done_Done!_Done?_Done.

V = {_, D, o, !, ?, n, e, .}

i = 1, most frequent pair: Do with frequency 9

V = {_, D, o, !, ?, n, e, ., v_0 = Do}

New text: v_0 _v_0!_v_0!_v_0!_v_0?_v_0 ne_v_0 ne!_v_0 ne?_v_0 ne.

i = 2: most frequent pair: !_ with frequency 5

V = {_, D, o, !, ?, n, e, ., v_0 = Do, v_1 = !_}

New text: v_0 _v_0 v_1 v_0 v_1 v_0 v_1 v_0?_v_0 ne_v_0 ne v_1 v_0 ne?_v_0 ne.

i = 3: most frequent pair: v_1 v_0 with frequency 4 (or ne with frequency 4)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we illustrate the first two steps of the algorithm. Having tokens that combine a character n-gram with a word boundary is important to identify character sequences that would typically occur at the end (or beginning) of a word and to distinguish them from character sequences that occur in the middle of a word.

For example, the sequence “ed” at the end of of an English word (like “worked”) implies a different meaning of this character sequences at the beginning of a word (like “education”) or in the middle of a word (like “medicine”)

Question

Fasttext speeds up learning by

1. Considering subwords of words
2. By selecting the most frequent phrases in the text as tokens
3. By selecting the most frequent subwords in the text as tokens
4. By pre-computing frequencies of n-grams

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

1.3.5 Glove

GLOVE is based on the following analysis

	x = solid	x = gas	x = water	x = random
$P(x\|\text{ice})$				
$P(x\|\text{steam})$				
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$				

1. Consider two terms, with similar, but related meaning: ice, steam
2. Determine the conditional probability $P(x|ice)$, $P(x|steam)$ of other terms x to co-occur within a window
3. Determine the ratio of these two conditional probabilities

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The skipgram and CBOW models are one variant among several similar models that have been proposed recently to create word representations that capture word semantics.

One popular model is GLOVE, which is based on the observation that ratios of probabilities of co-occurrence can be used to capture more accurately semantic relationships among terms.

Glove

Observation: ratio captures additional information

	x = solid	x = gas	x = water	x = random
$P(x\|\text{ice})$	large	small	large	small
$P(x\|\text{steam})$	small	large	large	small
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$	large	small	~1	~1

"solid" is related to "ice" but unrelated to "steam":
we expect a larger ratio of co-occurrence probabilities

"gas" is related to "steam" but unrelated to "ice":
we expect a smaller ratio of co-occurrence probabilities

"water" is related to both "ice" and "steam":
we expect a ratio of co-occurrence probabilities that is close to 1 (both large)

A random word that is unrelated to both "ice" and "steam":
we expect a ratio of co-occurrence probabilities that is close to 1 (both small)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In this example, the term solid is much more likely to co-occur with ice, than with water, and similarly steam co-occurs much more likely with gas than with ice. On the other hand, water co-occurs frequently with both. In the Glove paper it is stated: "Compared to the raw probabilities, the ratio is better able to distinguish relevant words (solid and gas) from irrelevant words (water and fashion) and it is also better able to discriminate between the two relevant words".

Global Co-occurrence Count

Word embeddings with global vectors (GloVe)

- Denote by x_{ik} the global co-occurrence count of words w_i and w_k in the vocabulary, where w_i is the center word and w_k is the context word
- Note: $x_{ik} = x_{ki}$
- Denote by x_i the total number of occurrences of word w_i

Then

$$P(w_k|w_i) = \frac{x_{ik}}{x_i}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to consider the property illustrated before, Glove uses the global co-occurrence counts among words within a window. In this way, it incorporates into the model a global statistics on the co-occurrence of words, different to skipgram/CBOW which iterate over the different occurrences while learning the parameters.

Using the global co-occurrence counts we can determine a co-occurrence probability.

Modeling Ratios

Find a function f such that for three words w_i, w_j, w_k

$$f_\theta(w_i, w_j, w_k) = \frac{P_\theta(w_k|w_i)}{P_\theta(w_k|w_j)}$$

If we choose

$$f_\theta(w_i, w_j, w_k) = e^{(\boldsymbol{w}_i - \boldsymbol{w}_j)\cdot \boldsymbol{w}_k} = \frac{e^{\boldsymbol{w}_i \cdot \boldsymbol{w}_k}}{e^{\boldsymbol{w}_j \cdot \boldsymbol{w}_k}}$$

we get

$$e^{\boldsymbol{w}_i \cdot \boldsymbol{w}_k} = P_\theta(w_k|w_i) = \frac{x_{ik}}{x_i}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to formalize the observation that we have described, we need to design a function that models ratios between co-occurrence probabilities derived from their vector representations. In Glove this function is chosen to be of the form of an exponential of the scalar product of the difference of the representation of two (context) words with the representation of a given word. With this function, it follows that the conditional probability of co-occurrence is modelled as the exponential of the scalar product of the representations of the two words.

Glove Loss Function

With word w_i and context words w_k

Assume

$$e^{\boldsymbol{w}_i \cdot \boldsymbol{w}_k} = P_\theta(w_k | w_i) = \frac{x_{ik}}{x_i}$$

taking the logarithm

$$\boldsymbol{w}_i \cdot \boldsymbol{w}_k = \log x_{ik} - \log x_i$$

Adding additional bias terms for w_i and w_k

$$\boldsymbol{w}_i \cdot \boldsymbol{w}_k + b_i + b_k \approx \log x_{ik}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Based on the function we have chosen to represent the conditional probability of word cooccurrence, we can derive a loss function. By taking the logarithm we obtain a linear relationship between the scalar product and the co-occurrence statistics. By adding so-called bias terms, which correspond to additional vector representations of the words, we obtain a relationship between the word embedding vectors and the co-occurrence probabilities. Note that by adding a bias term $\boldsymbol{b}_i$ the term $\log x_i$ is considered in this relationship. For maintaining symmetry bias terms are added for both words.

Glove Loss Function

Squared error loss with weights

$$J(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{m} h(x_{ij})(\boldsymbol{w}_i \cdot \boldsymbol{w}_j + b_i + b_j - \log x_{ij})^2$$

Weight function

$$h(x) = \min\left(\left(\frac{x}{x_{max}}\right)^{\beta}, 1\right)$$

With $\beta = 0.75$ and $x_{max} = 100$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Based on the relationship introduced, a loss function is derived. In the case of Glove, the squared error is used. In addition, a weighting function h is introduced to address two problems:

- If x_{ij} equals zero, the loss function would be ill-defined. The limit $\lim_{x \to 0} h(x) \log{}^2(x)$ is finite, making the function well-defined
- Small values of x_{ij} carry in general less information and should thus have lower weight.

Question

The most important difference between Glove and skipgram is

1. That Glove considers the complete context of a word
2. That Glove computes a global frequency for word-context pair occurrences
3. That Glove uses a squared error loss function
4. That Glove does not differentiate words and context words

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Glove: Nearest words to Frog

Nearest words to
frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

leptodactylidae

rana

eleutherodactylus

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A first property is that similar words are grouped together. Here is a nice example, produced with Glove, that illustrates the point. (The underlying data is from Wikipedia).

Word embeddings also capture syntactic relationships, like singular-plural or comparative and superlative, as shown here. This type of visualizations is obtained by projecting from the d-dimensional word embedding space (appropriately) into a 2-dimensional space.

Word Analogies: semantic relationships

man:woman :: king:?

+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Even more interestingly, word embeddings enable “computing” with relationships (this is called the word analogy task). Analogies translate into linear mappings! We will later exploit this property for extracting relationships from text.

Furthermore, word embeddings can capture semantic meaning in dimensions. In this example, data on nutrition has been analyzed and terms indicating qualities of foods are extracted. The word embedding clearly organizes it according to properties that are human understandable, e.g. natural vs. artificial ingredients.

Properties of Word Embeddings

1. Similar terms are clustered
2. Syntactic and semantic relationships encoded as linear mappings
3. Dimensions can capture meaning

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We summarise some of the main properties of word embeddings.

Use of Word Embedding Models

Document Search

- Use word embedding vectors as document representation

Thesaurus construction and taxonomy induction

- Search engine for semantically analogous / related terms

Document classification

- Use of word embedding vectors of document terms as features

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Word embeddings have beyond retrieval a wide range of applications.

Impact

- Latent semantic indexing

Indexing by **latent semantic analysis**
S Deerwester, ST Dumais, GW Furnas... - Journal of the ..., 1990 - search.proquest.com
Cited by 12027 Related articles All 87 versions Web of Science: 3525 Cite Save

+2000

Indexing by latent semantic analysis
S Deerwester, ST Dumais, GW Furnas... - Journal of the ..., 1990 - Wiley Online Library
A new method for automatic indexing and retrieval is described. The approach is to take
219,GrantRating (2).pdf
at higher-order structure in the association of terms with documents
") in order to improve the detection of relevant documents on the basis of ...
Cited by 13986 Related articles All 76 versions Web of Science: 4451

- Latent Dirichlet allocation

Latent dirichlet allocation
DM Blei, AY Ng, MI Jordan - Journal of machine Learning research, 2003 - jmlr.org
Cited by 17791 Related articles All 123 versions Web of Science: 5278 Cite Save

+8000

Latent dirichlet allocation
DM Blei, AY Ng, MI Jordan - Journal of machine Learning research, 2003 - jmlr.org
We describe latent Dirichlet allocation (LDA), a generative probabilistic model for collections of discrete data such as text corpora. LDA is a three-level hierarchical Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of ...
Cited by 25905 Related articles All 98 versions Web of Science: 8674

- Word embeddings

Distributed representations of words and phrases and their compositionality
T Mikolov, I Sutskever, K Chen, GS Corrado... - Advances in neural ..., 2013 - papers.nips.cc
Cited by 3370 Related articles All 23 versions Cite Save

+8000

Distributed representations of words and phrases and their compositionality
T Mikolov, I Sutskever, K Chen, GS Corrado... - Advances in neural ..., 2013 - papers.nips.cc
The recently introduced continuous Skip-gram model is an efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships. In this paper we present several improvements that make the Skip-gram model more expressive and enable it to learn higher quality vectors more rapidly. We show that by subsampling frequent words we obtain significant speedup, and also learn higher quality representations as measured by our tasks. We also introduce ...
Cited by 11619 Related articles All 37 versions

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

References

Relevant articles

- Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*. 2013.
- Goldberg, Yoav, and Omer Levy. "word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method." *arXiv preprint arXiv:1402.3722* (2014).
- Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.
- http://videolectures.net/deeplearning2015_manning_language_vectors/

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Part 2: Recommender systems

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Recommender System

Given a *user* model and a set of *items*, a recommender system is a function that helps to match users with items by *ranking* the items in order of decreasing *relevance*

Learning a ranking function

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Assume a user is interested in some type of items, like products, movies, books, news or jobs. A recommender system is a system that based on available data on these items proposes to the users a ranked list of items according to their preferences. Therefore recommendation can be understood as a learning problem, where the function to be learnt is the ranking of items.

The data that is available to produce such a ranking includes ratings of used of the items, demographic information on the users and characteristics of the items, including their content.

Recommender systems are widely used in many applications to provide value for the user and the provider.

For users recommender systems help, e.g., in the context of a product search,

– find things that are interesting

– narrow down the set of choices

– help explore the space of options

– Discover new things

For providers recommender systems help to

– increase trust and customer loyalty with personalized services

– increase sales, click rates, page views etc.

– identify opportunities for promotion and persuasion

– obtain more knowledge about customers

In the collaborative paradigm, the recommender system uses the user profile, primarily by the earlier ratings that are considered to represent the user preferences, and community data consisting of ratings of items by other users.

It is called collaborative because all the users participate to the recommendation by providing ratings to the items that they viewed or purchased.

In the content-based paradigm, the recommender system uses the profile of the items that the user rated in the past to return a recommendation list with the most relevant items and their scores. It is called content-based because the items are associated with a set of attributes that summarize their content.

Beyond the two basic approaches of collaborative and content-based recommender systems, there exist also hybrid ones, which combine collaborative and content-based recommendation, and knowledge-based ones, which exploit domain knowledge to determine how certain item features meet the user needs.

2.1 Collaborative Filtering

A widely used approach to generate recommendations

- Used by large e-commerce sites
- Applicable in many domains (e.g., books, movies ...)
- Well understood and studied

Approach (*Wisdom of the crowd*)

- Users give ratings to items
- Assumption: users with similar tastes in the past will have similar tastes in the future

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Collaborative recommender systems are based on collaborative filtering. In collaborative filtering, given a user and an item not yet rated by the user, the goal is to estimate the user rating for this item by looking at the ratings for the same item that were given in the past by similar users. Collaborative filtering requires a community of users that provide ratings and a way to assess user similarity.

Collaborative filtering is further distinguished into user-based collaborative filtering and item-based collaborative filtering. We will introduce these two approaches next.

User-based Collaborative Filtering

Basic technique:

Given a user x and an item i not rated by x, estimate the rating $r_x(i)$ by

1. Finding a set of users $N_U(x)$ who rated the same items as x (= neighbours of x) in the past <u>and</u> who have rated i
2. Aggregate the ratings of i provided by $N_U(x)$

Compute the ratings for all the items not rated by x and recommend the best-rated ones

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

User-based collaborative filtering is based on the idea that for an item for which a user's rating is not known, users with similar tastes are identified to estimate the rating. In order to do so the approach requires first a metric to compute similarity by users. After selecting the most similar users $N_U(x)$ it also requires a way to aggregated the ratings these users provided.

In detail user-based cllaborative filtering works as follows. First one considers all items that the user in question has already rated. Then other users are searched that have also rated these items, but also have rated the new item for which we do not know the rating of user x. In this way users with similar interests are selected.

From the ratings of those similar users of the new item then a rating of the item is estimated.

Similarity Between Users

Pearson correlation coefficient

$$sim(x,y) = \frac{\sum_{i=1}^{N}\left(r_x(i)-\overline{r_x}\right)\left(r_y(i)-\overline{r_y}\right)}{\sqrt{\sum_{i=1}^{N}\left(r_x(i)-\overline{r_x}\right)^2}\sqrt{\sum_{i=1}^{N}\left(r_y(i)-\overline{r_y}\right)^2}}$$

Cosine Similarity

$$sim(x,y) = \cos(\vartheta) = \frac{\sum_{i=1}^{N} r_x(i)\cdot r_y(i)}{\sqrt{\sum_{i=1}^{N} r_x(i)^2}\sqrt{\sum_{i=1}^{N} r_y(i)^2}}$$

x, y: users

N : number of items i rated by <u>both</u> x and y

$r_x(i)$: rating of user x of item i

$\bar{r}_x$: average ratings of user x

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to determine similarity of users, a similarity measure is needed that is based on the past ratings of the users of a common set of items. We can consider those ratings as a vector and therefore an obvious choice for a similarity measure is cosine similarity returning a similarity in the interval [0.1]. For recommender systems an alternative measure that is widely used is the Pearson correlation coefficient which which returns a value between -1 and 1. Note that the Pearson correlation can be understood as a cosine similarity of the rating vectors recentered around their mean. Therefore it can adjust for different biases of users in their rating scales, i.e., users that consistently rate higher or lower.

One drawback of the Pearson correlation coefficient is that it is not defined, when the variance of one of the user ratings is 0 (e.g., a user with all ratings 1). However, in general, the correlation coefficient works well in general domains.

Example

	Item 1	Item 2	Item 3	Item 4	Item 5
U	5	3	4	4	???
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

sim_{corr}(U, User1) = 0.85
sim_{corr} (U, User2) = 0.71
sim_{corr} (U, User3) = 0
sim_{corr} (U, User4) = -0.79

sim_{cos}(U, User1) = 0.97
sim_{cos}(U, User2) = 0.99
sim_{cos}(U, User3) = 0.89
sim_{cos} (U, User4) = 0.79

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an example of similarity computed using the two metrics. It is possible to see how the users are not ranked in the same way. For example, the most similar user to user U is User 1, when the Pearson correlation coefficient is used, or User 2, when the cosine similarity is used. In fact, user 1 seems to have a bias towards lower ratings, but has a similar rating pattern as user U.

Aggregate the Ratings

A common aggregation function is

$$r_x(i) = \bar{r}_x + \frac{\sum_{y \in N_U(x)} sim(x,y)(r_y(i) - \bar{r}_y)}{\sum_{y \in N_U(x)} |sim(x,y)|}$$

$N_U(x)$: neighbours of user x

i : item not rated by x

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The aggregation function calculates whether the neighbours' ratings for the unseen item *i* are higher or lower than their average. We can consider this term as the bias of the user *y* w.r.t. item *i*. Then the function combines the rating bias using the similarity as a weight, so that the most similar neighbours will have more importance. Finally, the aggregated neighbours' bias is added/subtracted from user's *x* average rating.

Example

	Item 1	Item 2	Item 3	Item 4	Item 5
U	5	3	4	4	???
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

Closest users to U: sim_{corr}(U, User1) = 0.85, sim_{corr} (U, User2) = 0.71

$$\bar{r}_U = 4$$

$$(r_{U1}(I5) - \bar{r}_{U1}) = 3 - \frac{12}{5} = \frac{3}{5}, (r_{U2}(I5) - \bar{r}_{U2}) = 5 - \frac{19}{5} = \frac{6}{5}$$

$$r_U(I5) = 4 + \frac{0.85 * \frac{3}{5} + 0.71 * \frac{6}{5}}{0.85 + 0.71} = 4.87 \approx 5$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we demonstrate in detail the calculation of an estimated rating with user-based collaborative filtering.

User-based Collaborative Filtering

Problems

- Cold start: users or items without ratings
- Scalability: large numbers of users
- Data dispersion: highly variable ratings, difficult to find similar users

Possible solution

- Item-based collaborative filtering

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A typical problem of user-based collaborative filtering is the cold-start problem. The recommendation for a new user that did not rate anything yet or for an item that was never rated by any user. Also, user-based collaborative filtering is problematic when there are millions of users and/or items, such as we find them on the typical Internet platforms like Amazon or Netflix. For example, for each user the most similar users have to be found doing nearest neighbour search from a large set of users. With large numbers of ratings, it becomes hard to find common ratings between users, due to the high dispersion of data.

Item-based Collaborative Filtering

The basic technique: use the similarity between items (and not users) to make predictions

- Given a user x and an item i not rated by x, estimate the rating $r_x(i)$ by

  1. Find a set of items $N_I(i)$ which are similar to i and that have been rated by x
  2. Aggregate the ratings of the items in $N_I(i)$ and use the aggregation as prediction of $r_x(i)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As opposed to user-based collaborative filtering, in item-based collaborative filtering the rating for an item for which a user's rating is not known, is derived from items that have similar ratings to the item in question, and not from users that have similar tastes as the user. Analogously, this requires a method to compute similarity among the ratings of items in order to determine a neighbourhood of similar items and a method for aggregating the ratings of the similar items.

The figure illustrates the approach. For obtaining an estimation of the rating of user x of an item i the user has not rated, first a set of items is identified that is similar, in the sense that it has received similar ratings as the item in question.

Similarity Between Items

	Item 1	Item 2	Item 3	Item 4	Item 5
U	5	3	4	4	???
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

sim_{corr}(item5, item1) = 0.97
sim_{corr}(item5, item2) = -0.48
sim_{corr}(item5, item3) = -0.43
sim_{corr}(item5, item4) = 0.58

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The attributes that define an item are the ratings of the users different than U. The similarity can be computed with the same functions used before for user-based collaborative filtering, Pearson correlation coefficient or cosine similarity. In this example, similarity is computed with the correlation coefficient, and it is possible to see how item 1 and 4 are the most similar to item 5. Thus, aggregating the ratings of these items given by user U is an estimate of the rating of user U for item 5.

Aggregate the Ratings

A common aggregation function is

$$r_x(i) = \frac{\sum_{j \in N_I(i)} sim(i,j) r_x(j)}{\sum_{j \in N_I(i)} |sim(i,j)|}$$

$N_I(i)$: neighbours of item i

j : item rated by x

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The aggregation function computes the prediction of an item i for a user x by computing the sum of the ratings given by the user on the items similar to i. Each rating is weighted by the corresponding similarity sim(i,j) between items i and j. Note that here bias in the ratings of other users is not taken into account.

Example

	Item 1	Item 2	Item 3	Item 4	Item 5
U	5	3	4	4	???
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

Closest items: sim_{corr}(item5, item1) = 0.97, sim_{corr}(item5, item4) = 0.58

$r_U(item5)$ = (0.97∗5 +0.58∗4)/(0.97 +0.58)=4.62 ≈ 5

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we demonstrate in detail the calculation of an estimated rating with item-based collaborative filtering.

Item-based Collaborative Filtering

Item-based collaborative filtering does not solve the cold-start problem but has better scalability

- Calculate all the pair-wise item similarities in advance
- Items similarities are more stable
- The neighbourhood $N_I(i)$ to be considered at runtime is small

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Although item-based collaborative filtering does not solve the cold-start problem, it is usually more suited to tackle big datasets. In fact, the item similarities can be computed in advance, given that the items are more stable than the users. In general, particularly in product recommendation systems, new items appear at a slower pace than new users. Furthermore, the size of the neighbourhood of a given item a is in general smaller than the neighbourhood of a user x, because $N_I(i)$ is composed of the other items rated by user x (e.g., tens of DVDs bought on Amazon), while $N_U(x)$ is composed by the other users that rated the same items as x (e.g., millions of users that watched The Godfather on Netflix).

Given 3 users with ratings…

u1:	1	3
u2:	2	4
u3:	1	4

A. $Sim_{corr}(u1, u2) > Sim_{corr}(u1, u3)$
B. $Sim_{corr}(u1, u2) = Sim_{corr}(u1, u3)$
C. $Sim_{corr}(u1, u2) < Sim_{corr}(u1, u3)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Item-based collaborative filtering addresses better the cold-start problem because …

A. usually there are fewer items than users
B. it uses ratings from items with similar content
C. item similarities can be pre-computed
D. none of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

2.2 Content-based Recommendation

Collaborative filtering uses only ratings, it does not exploit any other information about the items

However, the *content* of the item might provide some useful information

- E.g., recommend sci-fi novels to users who liked Asimov books in the past

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Both recommendation techniques we saw before are based on the ratings of the user exclusively, and exploit no information about the item. However, the content of the item might provide useful information for recommendation. Content-based recommendation takes into consideration the items that have been rated by the user and the content of all existing items, significantly reducing the scalability problems the recommendation is not relying on data from a large community of users.

Content-based Recommendation

The basic technique

- Given the items that have been rated by user x in the past

    1. Find the items that are similar to the past items
    2. Aggregate the ratings of the most similar items

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic technique for content-based recommendation needs to define

1 A way to formalize the item content

2 A similarity measure between items

3 An aggregation function for the ratings

Content-based Recommendation

Most methods originate from information retrieval to extract the content of an item

Given an item i and a term t (e.g. movie synopsis, book review), compute the tf-idf weights

$$w(t,i) = tf(t,i)idf(t) = \frac{freq(t,i)}{\max_{s \in V} freq(s,i)} \log \frac{N}{n(t)}$$

N : number of items to recommend

$n(t)$: number of items where term t appears

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A standard approach for content-based recommendation for items is the characterisation of the items in terms of tf-idf. Each item can be described by a set of terms that appear in their textual description description, with their associated weight, the tf-idf measure.

To compute the tf-idf measure measure the standard processing steps from information retrieval, including stopword elimination, and stemming, can be applied.

Similarity between Items

Cosine similarity

$$sim(i,j) = \cos(\theta) = \frac{\sum_{t \in V} w(t,i) w(t,j)}{\sqrt{\sum_{t \in V} w(t,i)^2} \sqrt{\sum_{t \in V} w(t,j)^2}}$$

i, j: items

V : vocabulary of all terms

$w(t,i)$: tf-idf of term t in item i

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The cosine similarity in content-based recommendation considers the items i and j as two |V|-dimensional vectors, where each dimension is the tf-idf measure of a specific term t.

Making Predictions

Given a set of items D that have been rated by the user, find the nearest neighbours $N_I(i)$ in D of a new item i

Use the ratings of the nearest neighbours to predict the rating of i with the aggregation function

$$r_x(i) = \frac{\sum_{j \in N_I(i)} sim(i,j) r_x(j)}{\sum_{j \in N_I(i)} |sim(i,j)|}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

After representing items by their tf-idf vectors, to estimate the rating of an item i not yet seen/rated by user x, the nearest neighbours of i in the subset of items that have been already rated by the user x are found and their ratings are aggregated. Note that this approach to predict the rating is analogous to the use of the kNN nearest-neighbour classification for document classification. In both cases, the nearest documents are used as predictors for the most likely label, respectively rating.

Content-based Recommendation

Problems

- Cold start problem for users with no ratings
- Content can be difficult or impossible to extract (multimedia)
- Tends to recommend “more of the same”

Scalable: tf-idf of items can be computed offline

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Content-based recommendation also suffers from the cold start problem as collaborative filtering, but only for the users that did not rate anything in the past. Another problem is that sometimes the information available to extract the content is limited (e.g., movie synopsis versus full plot) or impossible to process (e.g. a sound or a video). Finally, content-based recommendation tends to recommend more of the same, it does not surprise the user with new interesting items.

On the other hand, the approach is in general more scalable, because it does not rely on a community of users to provide recommendations. Furthermore, the tf-idf measure can be computed once a new item enters the system, and then an update of the pair-wise similarities with all the existing items can be performed.

For a user that has not done any ratings, which method can make a prediction?

A. User-based collaborative RS
B. Item-based collaborative RS
C. Content-based RS
D. None of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For an item that has not received any ratings, which method can make a prediction?

A. User-based collaborative RS
B. Item-based collaborative RS
C. Content-based RS
D. None of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

2.3 Matrix Factorization

	I1	I2	I3	I4
U1	3	1	-	-
U2	-	3	4	3
U3	3	-	1	-
U4	-	5	5	2

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Matrix factorization transforms the user-item rating matrix into a latent factor representation, where each user and item is described as a vector in a k-dimensional space, with a dimension k significantly smaller than the number of users or items. It can be understood as a combination of user-based and item-based collaborative filtering.

The factors describe different characteristics of users, some of which might be interpretable. Proximity in the latent space between items and users indicates that the users has a preference for the corresponding item. The preference can be computed as the cosine similarity between the vectors representing the item and users.

This approach is comparable to the use of singular value decomposition for document retrieval. However, a direct application of SVD is not possible because the user-item matrix is not complete, due to missing ratings. Therefore other approaches for determining the latent space representation are required.

Derive Latent Factors

Rating Matrix R with dimension $|U|\times|I|$

- Users U and Items I

Goal: Decompose R (approximatively)

$$R \approx P\times Q^t = \hat{R}$$

d latent factors (low dimension)

- P is a $|U|\times d$ matrix: user features
- Q is a $|I|\times d$ matrix: item features

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Since a algebraic approach for matrix factorization, as with SVD, is not possible, an alternative way is to produce the matrix factorization by approximation. For doing so, two factor matrices P and Q are introduced that represent the users and items in the latent space. The objective is to find these matrices, so that their produce is close, but not identical, to the original rating matrix R.

Using our approach to illustrate LSI, we can also ilustrate matrix factorization in the same way.

Computing Matrix Factorization

Problem: R has many undefined values (missing ratings)!

- SVD matrix decomposition (as used for LSI) not applicable

Formulate an optimization problem

$$\min_{p,q} \sum_{(i,j)known} \left(r_{ij} - \hat{r}_{ij}\right)^2, \hat{r}_{ij} = \sum_{k=1}^{d} p_{ik} q_{kj}$$

Apply SGD

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Since SVD can not be applied to a matrix with missing values, we formulate an optimization problem. Thus, to learn the factor vectors q_i and p_u , the optimization minimizes the squared error of the differences between the entries in the original rating matrix R, and the matrix $P \times Q^t = \hat{R}$, but only for those entries in R that are defined. Note that in $\hat{R}$ there can be non-zero entries in the positions where no ratings exist.

To solve the optimization standard SGD can be applied.

Approximation Error

For a given rating r_{ij}

$$e_{ij}^2 = \left(r_{ij} - \hat{r}_{ij}\right)^2 = \left(r_{ij} - \sum_{k=1}^{d} p_{ik} q_{kj}\right)^2$$

Minimizing error: compute gradient

$$\frac{\partial}{\partial p_{ik}} e_{ij}^2 = -2\left(r_{ij} - \hat{r}_{ij}\right) q_{kj} = -2 e_{ij} q_{kj}$$

$$\frac{\partial}{\partial q_{kj}} e_{ij}^2 = -2\left(r_{ij} - \hat{r}_{ij}\right) p_{ik} = -2 e_{ij} p_{ik}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To perform SGD the model is updated for each error term separately. Therefore, we can compute the gradients of the error terms independently, which results in simple linear terms.

Stochastic Gradient Descent

Update rule: for some random pair i, j

$$p_{ik} := p_{ik} - \alpha \frac{\partial}{\partial p_{ik}} e_{ij}^2 = p_{ik} + 2\alpha e_{ij} q_{kj}$$

$$q_{kj} := q_{kj} - \alpha \frac{\partial}{\partial q_{kj}} e_{ij}^2 = q_{kj} + 2\alpha e_{ij}\, p_{ik}$$

Repeat until error is small

Needs only to be computed for ratings r_{ij} that exist!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With the gradients we can then define the update rules. Since we perform SGD, we need only to update error values for which a corresponding rating exists. As a side effect, once the decomposition is computed, we obtain also estimates for the ratings for the other user-item pairs, for which not rating exists in the training data.

Regularization

In order to avoid overfitting

$$e_{ij}^2 = \left(r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj} \right)^2 + \lambda(\|P\|^2 + \|Q\|^2)$$

And then

$$p_{ik} := p_{ik} - \alpha \frac{\partial}{\partial p_{ik}} e_{ij}^2 = p_{ik} + 2\alpha\left(e_{ij} q_{kj} - \lambda p_{ik}\right)$$

$$q_{kj} := q_{kj} - \alpha \frac{\partial}{\partial q_{kj}} e_{ij}^2 = q_{kj} + 2\alpha\left(e_{ij}\, p_{ik} - \lambda q_{kj}\right)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to avoid over-fitting, as usual a regularization term is added, and the update rule is modified correspondingly. The regularization term assures that the parameters of the model are minimized.

Issues with Basic Matrix Factorization

Matrix P grows with number of users

- Problematic with large user populations, e.g., social networks
- Users modelled individually
  - Many models to manage
- Training becomes expensive
  - Large training matrix

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The main drawback of matrix factorization is that it does not scale with the number of users, which in real world scenarios can become very large (e.g., customers of an ecommerce or streaming service) and therefore the training can become expensive. This probalm is addressed by the next approach that we will introduce.

Question

How does matrix factorization address the issue of missing ratings?

A. It uses regularization of the rating matrix
B. It performs gradient descent only for existing ratings
C. It sets missing ratings to zero
D. It maps ratings into a lower-dimensional space

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Question

With matrix factorization one estimates ratings of unrated items

A. By retrieving the corresponding item from a user vector
B. By retrieving the corresponding item from an item vector
C. By computing the product of a user and an item vector
D. By looking up the rating in an approximation of the original rating matrix

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

2.4 SLIM, Sparse Linear Methods

Idea: model item-item relationship directly, without explicitly representing users

- Items with similar ratings have similar representation
- Compute their ratings as linear combination from a few samples -> sparse
- Avoids construction of large user-item matrix

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to overcome the issue of scaling with the user population, alternative models have been considered. SLIM is a representative of those.

It is based on the idea to a create a representation of items, such that items with similar ratings have a similar representation. Thus, it is comparable to the approach of item-based collaborative recommender algorithms, that we have introduced earlier. The difference is that instead of comparing items by their full rating vector, consisting of the ratings of the full user population, a sparse representation of item vectors is derived. This sparse representation that is encoded in a weight aggregation matrix W allows to recover the rating of an item from (a few) ratings from similar items.

Example

R

	Item 1	Item 2	Item 3	Item 4	Item 5
U	5	3	4	4	5
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

Optimal vector to recover original rating

Id 5
0
0
0
0
1

Sparse vector to recover original rating

w 5
0.6
0
0
0.4
0

Recovering approximate ratings for item 5 using w_5

U: $0.6*5 + 0.4*4 = 4.6$

User 1: $0.6*3 + 0.4*3 = 3$

User 2: $0.6*4 + 0.4*3 = 3.6$ ...

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In this example we illustrate the idea: In item-based collaborative filtering we reconstructed an unknown rating (e.g., item 5 of user U) by considering the ratings of the most similar users for item 5.

With sparse approximation, we try to find for each item a linear combination of other items that allow to reconstruct the ratings of the item for each user In our example we would use items 1 and 4 to reconstruct the rating of item 5. This vector used for reconstructing item 5 cannot contain a 1 on the diagonal, otherwise it would be trivial to reconstruct the rating of item 5.

Weight Aggregation Matrix

Approximate the Rating Matrix R with dimension $|U|\times|I|$ using a **sparse** weight aggregation matrix W

$$\hat{R} \approx RW$$

- W is a $|I|\times|I|$matrix
- $\mathrm{diag}(W) = 0$, otherwise, trivial solution with W as identity matrix

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To formalize the idea, we introduct weight aggregation matrix, which represents the aggregation vectors for all items. It has the dimension $|I|\times|I|$ since it maps the ratings of items into ratings of items, and it is designed to approximate the original rating matrix R. As an additional condition it has 0 on the diagonal, otherwise it would be trivial to reconstruct the original rating matrix.

What approximating the rating for a given user i, the item vector i (which contains the ratings of the user) is multiplied with the corresponding column form W.

Recommendation with SLIM

For all items of the user without rating, compute the estimated rating score

- Since the diagonal in W is zero, the score can be reconstructed from entries for other items
- Since W is sparse only a low number of other item ratings will be used

Recommendation is done by sorting the items of user i by the estimated rating and selecting the top-k items

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For performing a recommendation for a given user, for all unrated items of the user an estimated rating is computed by multiplying the users item rating vector with the sparse vectors from W corresponding to the unrated items. Since a user has in general rated only few items, and the vectors from are sparse, this is computationally more efficient than performing a full matrix multiplication. The final recommendation is obatined by selecting the top-k rated items.

Optimizing the Weight Aggregation Matrix

Impose constraints on W

- $\mathrm{diag}(W) = 0$: Ratings of an item needs to be reconstructed from other ratings
- $W \geq 0$: all entries are positive
- Minimize the norm of W (regularization)
  - Minimize $|W|_1$, minimizes number of positive weights (promotes sparsity, Lasso regression)
  - Minimize $|W|_2$, minimizes the size of weights (reduces model complexity, Ridge regression)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

It remains to be seen of how the aggregation matrix can be obtained. Apart from minimizing the distance from the actual weights in R and the reconstructed weights in RW, an important aspect is to promote sparsity of the aggregation matrix. For this, standard methods from machine learning are applied. Including into the loss function the 1-norm promotes the sparsity of the matrix and including the 2-norm minimizes the sizes of the weights.

Optimization Problem

Solve

$$\min_{W} \frac{1}{2}|R - RW|_2^2 + \frac{\beta}{2}|W|_2^2 + \lambda|W|_1$$

subject to $\mathrm{diag}(W) = 0$ and $W \geq 0$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As a result, this is the optimization problem to be solved in order to obtain the sparse matrix W.

Solving the Problem

Can be solved for each item separately (in parallel)

$$\min_{w_j} \frac{1}{2}\left|c_j - Rw_j\right|_2^2 + \frac{\beta}{2}\left|w_j\right|_2^2 + \lambda\left|w_j\right|_1$$

subject to $w_j \geq 0$ and $w_{j,j} = 0$

Standard methods for solving this problem exist (coordinate descent)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We observe that the optimization problem can be solved independently for all items, therefore the optimization can be performed in parallel. To solve the individual optimizations for items standard optimization techniques can be applied such as coordinate descent. In coordinate descent the function is iteratively optimized using gradient descent along the different coordinates.

Properties

- W directly represents relations among items
- Due to sparsity of matrix W top-k recommendations are less expensive to compute
  - scales with number of non-zero values
  - typically, 1% non-zero entries
- Training can be performed with a subset of users
- Users need not to be individually modelled, only their rating history needs to be stored

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The SLIM method allows to compute recommendations very efficiently, as the weight aggregation matrix is very sparse. In experiments typical values have shown the the matrix has only 1% of non-zero values. To further optimize the approach, the optimization for obtaining the aggregation matrix can be performed using only a subset users, if the user population is very large. No model of users is built with this approach, only their rating history needs to be recorded.

Question

Which of the following matrices is not sparse?

A. The matrix W

B. The matrix $\hat{R}$

C. The matrix RW

D. More than one of the above are sparse

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Question

If users rate about 1% of items, and W has sparsity 1%, how many multiplications are needed to recommend the top-k elements?

A. $|I| * |I| * 0.0001$

B. $|I| * |I| * k * 0.01$

C. $|U| * |I| * k * 0.0001$

D. $|U| * |I| * 0.0001$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

2.5 Evaluation of Recommender System

Standard error metrics used:

Root Mean Square Error (RMSE)

$$RMSE = \sqrt{\frac{1}{N}\sum_{r_{ij}}\left(r_{ij} - p_{ij}\right)^2}$$

N is number of available ratings r_{ij}

p_{ij} is the predicted ratings

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A simple standard metrics to evaluate globally the quality of a recommend system is RSME. It compares the known ratings to the predicted ratings, using l2-norm.

RMSE

RMSE corresponds to the optimization objective used in matrix factorization

RMSE can be evaluated

- Globally, averaging over all ratings
- per-user/per-item with averaging over all users/items

Better alternative: NDCG

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

RMSE can be evaluated using different strategies, either globally over all items, or first by computing the RMSE for items/users separately and then averaging the resulting values.

Question

Assume in top-1 retrieval, method 1 ranks (2, 3, 1) and method ranks 2 is (2, 1, 3)

A. RMSE(rec 1) < RMSE(rec 2) and DCG(rec 1) > DCG(rec 2)
B. RMSE(rec 1) = RMSE(rec 2) and DCG(rec 1) > DCG(rec 2)
C. RMSE(rec 1) < RMSE(rec 2) and DCG(rec 1) = DCG(rec 2)
D. RMSE(rec 1) = RMSE(rec 2) and DCG(rec 1) = DCG(rec 2)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Outlook: Bayesian Personalized Ranking

Ranking objective functions

- In practice recommender systems do not predict scores, but rankings

Instead of optimizing RMSE (as with matrix factorization) we can optimize the ranking for a user i by

- Sampling: a positive item j (which the user chose) and a negative item $\hat{j}$ (which the user never interacted with)
- Learning by maximizing the difference between their predicted scores: $\log \sigma(r_{ij} - r_{i\hat{j}})$
- In this way, the score is meaningless, only the order matters.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Finally, the absolute rating values are often not essential for recommendation. What counts are the items being proposed. Therefore, more recent approaches abandon RMSE as an optimization objective, and replace it with a metrics that essentially distinguishes only whether items are being selected by users or not. Bayesian Personalized Ranking is a representative of this class of approaches. We will encounter this method in more detail later in another context.

Summary

Recommender Systems evolved, similarly as IR, from basic models, to models based on optimizing an objective function

- Scaling to large numbers of users is a critical issue
- Cold-start problem is not solved by the more advanced models

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The evolution of models for recommender systems was similar to that from models for information retrieval. Whereas IR models have to deal with scaling to large number of documents, RS have to take into account scaling to large numbers of users.

One problem that remains open, also with the advanced models, is the cold start problem. It is fundamentally not possible to make predictions on future ratings from historical ratings, if no historical ratings are available. Only using additional contextual information this problem can be solved.

References

- Chapter 9 in mmds.org
- Recommender Systems Handbook, Springer 2015
- Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." *Computer* 42.8 (2009): 30-37.
- Ning, Xia, and George Karypis. "Slim: Sparse linear methods for top-n recommender systems." *2011 IEEE 11th International Conference on Data Mining*. IEEE, 2011.
- Rendle, Steffen, et al. "BPR: Bayesian personalized ranking from implicit feedback." Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. 2009.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

3.1 DOCUMENT CLASSIFICATION

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Document Classification

Task: Given a training set D of labelled documents, construct a **classifier** to decide for an unlabeled document which label (or class) it has

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Document classification is the task to assign a class label to documents from an existing finite set of available labels. It is a task of huge practical importance in the management od documents. Information retrieval can be understood as one example of document classification, where the task is to distinguish relevant from non-relevant documents. But there are many other use cases, such as spam filtering, sentiment analysis, topic detection.

Document Classifiers

Features

- Words of the documents
  - bag of words
  - document vector
- More detailed information on words
  - Phrases
  - Word fragments (subwords, character n-grams))
  - Grammatical features
- Any metadata known about the document and its author

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As for any classification problem, a central question is the choice of features used for classification. Like in information retrieval, the word features, which we called the full text, is an obvious choice. We can identify nevertheless many other features that can be exploited. Related to the text more detailed information about the words, including their grammatical features, can be used. Also, metadata on the documents, like the authorship or the history of usage during editing and reading can be considered.

Example

PromotionDesSciences @EPFL_SPS · May 1
More than 4000 visitors attended Scientastic the science festival of **EPFL**, organised in Valais for the first time. actu.epfl.ch/news/scientast…

Bag of words: {more, than, visitors, attend, …}

Phrases: {"science festival", "first time"}

Word fragments: {mor, ore, tha, han, vis, isi, ..}

Grammatical features:

More than 4000 visitors attended Scientastic the science festival of EPFL , organised in Valais for the first time

http://parts-of-speech.info/

Author: @EPFL_SPS

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we give a few examples of features that could be extracted from a simple tweet.

Challenge in Document Classification

The feature space is of very high dimension

- Vocabulary size
- All word bigrams
- All character trigrams
- etc.

Dealing with high dimensionality

- Feature selection, e.g., mutual information
- Dimensionality reduction, e.g., word embedding
- Classification algorithms that scale well

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Document classification is a special case of classification, a problem widely studied in machine learning. What makes the problem specific, is the very large feature space that can be generated from documents, e.g., large vocabulary sizes.

In order to address the issue of large feature spaces, in machine learning different approaches are pursued.

- Feature selection is used to identify among the available features those that have the largest distinctive power on deciding the class label. Many approaches, such as mutual information, are available from standard machine learning.
- Dimensionality reduction is an approach that we have already introduced in the context of information retrieval, both with LSI and word embeddings.
- Finally, having large number of features, suggest to also prefer classification algorithms that scale well with large numbers of features.

3.1.1 k-Nearest-Neighbors (kNN)

Simple approach: use vector space model

To classify a document d

- retrieve the k nearest neighbors in the training set according to the vector space model
- Choose the majority class label as the class of the document

Underlying assumptions

- documents in the same class form contiguous regions
- different classes do not overlap

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A standard method for document classification is rooted in ideas we have already introduced for information retrieval. It is assumed that there exists a training set of labelled documents. In the kNN approach, the top k labelled documents that are most similar to the document to be classified are retrieved. The predicted class label is then chosen to be the one that has the majority among the retrieved documents. Ideally, k is chosen to be odd to break ties. The parameter k can also be determined using standard approaches of parameter tuning with a validation sets.

kNN relies on the implicit assumption that documents in the same class form contiguous regions, i.e., are close to each other, and that the regions of documents from different classes are not overlapping.

From the perspective of computational complexity, kNN exhibits the following properties:

- Training requires to compute vector representations of documents. In the case of using tf-idf weights this implies tokenization of documents and computing the idf statistics for the document collection.
- Inferring the class label requires the search for the closest k neighbors. Using simple scanning of the training set, the cost of inference grows linearly in the size of the training set, and therefore can be expensive.

Variations of kNN

Probabilistic estimates

- If k large: use $\frac{|N_{C,k}(d)|}{k}$ to estimate $P(C|d)$, the probability that the document d has class C
- $N_{C,k}(d) \subseteq D$ are the documents of class C among the k nearest neighbors of document d

Weighting

$$score(C,d) = \sum_{d' \in N_{C,k}(d)} \frac{\vec{d}}{|\vec{d}|} \frac{\vec{d'}}{|\vec{d'}|}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Two variations of kNN can be used. With probabilistic estimates not a class label is returned, but a probability distribution of the class labels. This is meaningful as k grows larger. With weighting the neighboring documents of a class contribute according to their distance to the document to be labelled. Therefore, documents that are more remote have less influence on the decision of the label. In general, this method is more accurate than the basic method.

Rocchio classification

For each class C compute the centroid of all training samples D_C of documents in the class

$$\mu(C) = \frac{1}{|D_C|} \sum_{d \in D_C} \vec{d}$$

For a document assign the class label of the closest centroid

$$\underset{C}{\operatorname{argmin}} \, | \, \mu(C) - \vec{d} |$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

An alternative model using the vector representation of documents for classification is the so-called Rocchio classification. As the name suggests, it exploits the same ideas that are used for user relevance feedback with the Rocchio method. For each class label, the centroids of the document vectors o the documents in the training set are computed. These centroids are then used to decide the class label of document by selecting the closest centroid. This method assumes that classes are not only continguous in the vector space, but also convex. This assumption is many practical cases not valid.

Linear vs. Non-linear Classification

Rocchio: linear

- high bias
- low variance

1NN: non-linear

- low bias
- high variance

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One of the most important questions in machine learning is the tradeoff among bias and variance. In short, bias essentially captures whether a model makes systematic mistakes independent of the training set used. High bias is in general associated with an incapacity of the model to capture essential aspects of the domain, which again results from too simple models. Variance captures the stability of the model under different training sets. High variance in general is associated with a too large capacity of the model to capture non-essential aspects of the domain, i.e. noise, a property that is called over-fitting. Rocchio and kNN provide a nice illustration of this trade-off. Rocchio is a linear model and as such not capable to capture non-linear decision boundaries among classes as illustrated on the left. Therefore for highly convoluted class boundaries the model will always have a large error. On the other hand, kNN is non-linear and can perfectly adapt to the nonlinear decision boundaries in the training data, when using k=1. However, this also means the model represents noise and outliers in the training data, and is prone to very instable predictions as the training data changes. kNN can be undrestood as a way to moderate the effect by “smoothing” the prediction boundaries over larger regions.

3.1.2 Naïve Bayes Classifier

Approach: apply Bayes law

Feature: Bag of words model: all words and their counts in the document

Using Bayes law, the probability that document d has the class C is

$$P(C|d) \propto P(C) \prod_{w \in d} P(w|C)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Another frequently used method used in document classification is Naïve Bayes Classifier. Like kNN can be understood as the analogue of classification for vector space retrieval, Naïve Bayes can be understood as the analogue of classification for probabilistic retrieval. By applying Bayes law and making as in probabilistic retrieval an independence assumptions on the occurrence of words in documents, we can derive the probability of a document belonging to a class. The probabilities $P(C)$ and $P(w|C)$ are then derived from the available training data.

One problem in this formulation of the probability $P(C|D)$ is that for words that do not occur together with the class label in the training set, the probability of a document containing this word to have the class label becomes zero. This is an analogue problem we stated for the basic form of probabilistic retrieval, where documents not containing all query words would never be retrieved.

Naïve Bayes Classifier Method

Training

How characteristic is word w for class C?

$$P(w|C) = \frac{|w \in d, d \in C| + 1}{\sum_{w'} |w' \in d, d \in C| + 1}$$ Laplacian smoothing

How frequent is class C?

$$P(C) = \frac{|D_C|}{|D|}$$

Classification: Thus, the most probable class is the one with the largest probability

$$C_{NB} = \underset{C}{\operatorname{argmax}} \left(\log P(C) + \sum_{w \in d} \log P(w|C) \right)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For training a Naïve Bayes model we have to estimate the probabilities $P(w|C)$ and $P(C)$. $P(C)$ is straightforward to compute it is simply the probability of each class label to occur in the training set. For the probability $P(w|C)$ we apply Laplacian smoothing to avoid the problem of words not occurring for a class in the training set. Then the probability is estimated as the likelihood of word occurring in the documents of the class as compared to the overall number of occurrences of the word.

Classification is performed by selecting the class with the highest probability to occur. This is computed by applying the logarithm to the estimated probability, as the summation is numerically more stable than multiplication.

3.1.3 Classification Using Word Embeddings

Reminder

government debt problems turning into banking crises as has happened in

saying that Europe needs unified banking regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

Probability that word w_i occurs with context word c (softmax function)

$$P_\theta(w_i|c) = \frac{e^{\boldsymbol{w}_i \cdot \boldsymbol{c}}}{\sum_{j=1}^{m} e^{\boldsymbol{w}_j \cdot \boldsymbol{c}}}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Word embeddings provide a representation of documents that can not only be used for information retrieval, but also for document classification. They provide a low-dimensional feature vector that can be used for performing classification with supervised learning. We will now introduce one recent approach that has been proposed with the Fasttext approach. In fact, the initial application of FastText was document classification.

Classification Task

Word embeddings have been optimized for a prediction task

- Given a context word c, does word w occur?

Changing the task

- Given a text p, does it belong to class label C?
  - Word w → Class Label C
  - Context words c → Paragraph p

Fasttext uses word embeddings for classification

- Supervised learning

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Document classification is a task that is different from the task considered for constructing word embeddings in an unsupervised approach, as we have introduced it earlier. The task we had considered consisted of predicting the co-occurrence of words and context words. For document classification we have a training data set of labelled documents. Using that training data, we can modify the task as follows: instead of predicting for a given context word the corresponding word (or vice versa), we predict for a piece of text, a paragraph, the related class label. Therefore, the role of the context word will be taken over by the paragraph for which the label is predicted, and the prediction is on the class label and not on the center word.

Adapting the Model for Classification

Representation

- Given a text p, the representation of p is then $\boldsymbol{p} = \frac{1}{|p|}\sum_{w \in p} \boldsymbol{w}$
- A class label C has a representation $\boldsymbol{c}$, a vector of dimension d

Softmax function

$$P_\theta(C|p) = \frac{e^{\boldsymbol{p}\cdot\boldsymbol{c}}}{\sum_{c'} e^{\boldsymbol{p}\cdot\boldsymbol{c'}}}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Since the prediction is performed not for single words, but for texts, the text from paragraphs is aggregated by computing the average word embedding vector of all words in the paragraph. For each class label also a d-dimensional vector is generated. The predicted probability of a class label occurring with a paragraph is then computed using the softmax function. Note that different to the use of the softmax function for the word prediction task, the number of class labels is usually small, so that the computation of the softmax function does not create signficant computational cost.

Implementation of the Model

Training data encodes labels with text

__label1__, text1

__label2__, text2

…

Loss function

$$\underset{\theta}{\operatorname{argmax}} \sum_{C} \sum_{p} \log P_{\theta}(C|p)$$

Learning using SGD

- Number of class labels usually small

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For the implementation of the model the training data is presented as pairs of labels and text paragraphs. The simplest loss function is maximizing the logarithm of the joint probabilites of all class label-paragraph pairs. Learning can be performed using standard stochastic gradient descent. In practice, different variants of this model are implemented, e.g., using alternative loss functions.

This figure summarizes the different steps of the fasttext classification model. Note that the encoding of the class labels into d-dimensional vectors can be representated by a matrix P, with the vectors for each of the l class labels as rows. The graphical representation used in this figure is a common way to describe a model used for learning. One can understand it to depict graphically the composition of a complex function.

Figure: Analysis and Optimization of fastText Linear Text Classifier, Vladimir Zolotov and David Kung, https://arxiv.org/abs/1702.05531

For which document classifier the training cost is low and inference is expensive?

A. for none
B. for kNN
C. for NB
D. for fasttext

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

How many among the listed classifiers can be used to derive probability estimate of the class label?

1NN, kNN, Rocchio, NB, fasttext

A. 1

B. 2

C. 3

D. 4

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

3.1.4 Transformer Models

A new architecture for machine learning models

- Initially conceived for machine translation

Extending standard neural network architectures (NN)

- Uses basic NN and backpropagation (BP)
- Detailed understanding of BP not needed, we have seen simple examples of BP earlier

Employed by well-known models, such as BERT

- State-of-the-art results in many tasks in NLP, computer vision and speech
- Based on **self-attention**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Transformer models are a new type of models for creating text embedding, that have initially been developed for machine translation, and later been adapted for other tasks, such as document classification. They are based on an extension of neural networks, using a mechanism known as self-attention. Self-attention introduces into the model a way to capture contextual dependencies among words in a text, which has led to a significant improvement in the performance for many natural language processing tasks. One of the best-known models of this class is BERT.

Self-attention

Given input vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_s$ and output vectors $\boldsymbol{y}_1, \dots, \boldsymbol{y}_s$ of dimension d

- Input vectors are word embeddings

Self-attention is a mapping from the input to the output vectors

$$\boldsymbol{y}_i = \sum_j w_{ij} \boldsymbol{x}_j \,, \sum_j w_{ij} = 1$$

Each output vector is a weighted average of the input vectors

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In its simplest form, the self-attention mechanism is very straightforward. It transforms a set of input vectors, e.g., representing the words of a sentence that are learnt like with word embeddings, to a set of output vectors of the same dimension. The self-attention mechanisms consists of a linear mapping from the input vectors to the output vectors. One can consider this mapping as computing weighted averages of the input vectors.

Note that in the following we will use a notation different from before, where we denote word vectors as $\boldsymbol{x}_1, \dots, \boldsymbol{x}_s$ and not as $\boldsymbol{w}_1, \dots, \boldsymbol{w}_s$. s one can interpret as the length of a sentence consisting of words $w_1, \dots, w_s$. We will consistently denote vectors in bold.

Self-attention Weights

The weights w_{ij} are not parameters that are learnt, but a function of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$

- Different options for defining this function exist

Simplest case $w'_{ij} = \boldsymbol{x}_i \cdot \boldsymbol{x}_j = \boldsymbol{x}_i^t \boldsymbol{x}_j$

- normalization using softmax: $w_{ij} = \frac{e^{w'_{ij}}}{\sum_j e^{w'_{ij}}}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

However, the weights used in the linear mapping are not some arbitrary parameters that are learnt from training data but depend themselves on the input vectors. In the simplest case, this dependency would be given as the scalar product of two input vectors x_i and x_j producing the weight w_{ij}, followed by the application of a softmax function, such that the weights add up to 1 for one dimension. This implies that while learning the vector representation of words, a dependency among the different vectors in a sequence of words is introduced.

Intuition

Assume input vectors correspond to words in sentences

- Assume a relation among those words exists, e.g., word w_i is the subject related to a verb w_j
- Then the vectors of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ will be more similar and produce a larger weight, whereas the others will be dissimilar

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The self-attention mechanism is what enables to capture the contextual relationships among the words in a text. In the visualization the weights for one self-attention layer are shown. One can observe that the word “on” is most related to the words sat.

The visualization tool can be found on:
https://towardsdatascience.com/deconstructing-bert-part-2-visualizing-the-inner-workings-of-attention-60a16d86b5c1

Illustration

The output vector at position i are obtained by multiplying the input vector at position j with weights w_{ij} and adding up

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This figure graphically illustrates of how the output vectors are obtained from the input vectors. Note that the input vectors occur in three different ways, as input, as factor in the weight applied to itself, and as factor in the weight applied to another input vector.

Applied to Text

1. For each input token, create an embedding vector
   - Using embedding matrix as for word embeddings

2. Convert a sentence into a sequence of vectors

3. Learn the parameters of the embedding matrix by performing a task using the vectors generated by the self-attention mechanism

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When the self-attention mechanism is applied to text, the input vectors are word embeddings derived from the input token vectors (in 1-hot representation) using an embedding matrix, and a sentence is converted into a sequence of vectors. The sequence of vectors is then fed into the self-attention mechanism. The vectors resulting from the self-attention mechanism are then used to perform a task, i.e., are used to compute a loss function that is optimized by adapting the model parameters of the word embedding matrices using gradient descent.

Simple Text Classifier

Using basic self-attention, we can realize a simple text classifier

1. Map 1-hot vectors to embedding vectors
2. Apply self-attention
3. Average the outputs: $\boldsymbol{y} = \frac{1}{s}\sum_{i=1}^{s} \boldsymbol{y}_i$
4. Project to class labels:
   $\boldsymbol{c}' = C\boldsymbol{y}, C$ is a $l \times d$ matrix, l number of labels
5. Apply softmax: $c_i = \frac{e^{c'_i}}{\sum_i e^{c'_i}}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With the basic self-attention mechanism described, it would be already possible to construct a basic text classifier. For this, the output vectors generated from the self-attention mechanism are averaged, and then projected to a vector that has the dimension l of the label space (e.g. in case of binary classification to a vector of dimension 2). In order to interpret the outputs as probabilities, finally a softmax function is applied.

Question

The number of parameters of the fasttext classifier and the simple self-attention classifier

1. Are the same
2. Fasttext has more
3. Self-attention has more

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information Flow in Transformer

The self-attention mechanism is the only mechanism where different input vectors interact to produce the output vector

- All other operations are applied to the input vectors in isolation

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The input vectors interact among each other using the self-attention mechanism. In this way relations among different words in a sentence are captured. All other operations applied to vectors in a transformer network are performed independently.

More “Tricks”

Standard transformer models introduce many additional features

- Queries, keys, values
- Scaling
- Multiple heads
- Residual connections
- Layer normalization
- Multilayer perceptrons
- Position embedding

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We have shown how to construct a basic text classifier using a self-attention mechanism. In practice, many additional mechanisms are used to produce the transformer models used today in practice. The design of these mechanism is based on numerous theoretical and empirical insights gained in research. In the following we will introduce the main ideas, without being able to enter in all details.

Queries, Keys, Values

The representation $\boldsymbol{x}_i$ of a token plays three different roles:

- It is compared to every other vector to establish the weights for its own output $\boldsymbol{y}_i$ (query)
- It is compared to every other vector to establish the weights for the output of the jth vector $\boldsymbol{y}_j$ (key)
- It is used as input to the weighted sum to compute each output vector once the weights have been established (value)

To facilitate the learning task, different linear transformations are applied to $\boldsymbol{x}_i$ to represent each of the roles

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As observed for the basic self-attention mechanism, the input vectors play 3 different rules in the self-attention mechanism.

The better accommodate for these three roles, linear transformation are introduced that generate 3 different vectors for them, from the input vectors.

These linear transformations introduce additional parameters that have to be learnt.

Queries, Keys, Values

Linear transformations W_q, W_k, W_v
($d \times d$ matrices)

$$\boldsymbol{q}_i = W_q \boldsymbol{x}_i,\ \boldsymbol{k}_i = W_k \boldsymbol{x}_i,\ \boldsymbol{v}_i = W_v \boldsymbol{x}_i$$

$$w'_{ij} = \boldsymbol{q}_i^t \boldsymbol{k}_j$$

$$w_{ij} = \mathrm{softmax}(w'_{ij})$$

$$\boldsymbol{y}_i = \sum_j w_{ij} \boldsymbol{v}_j$$

Note: the linear transformations are independent of the vectors

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The linear transformations for query, key and value are part of the parameters that will be learnt, but are independent of the input vectors, i.e., only 3 constant linear mappings are learnt for a self-attention mechanism.

The figure illustrates of how the 3 linear mappings are applied to the input vectors before the self-attention mechanisms computes a weighted average of the inputs.

Scaling

The softmax function can be sensitive to values with a very large range

- slows down learning or causes it to stop altogether
- average value of the dot product grows with the embedding dimension d
- scaling avoids inputs to the softmax function from growing too large

$$w'_{ij} = \frac{\boldsymbol{q}_i^t \boldsymbol{k}_j}{\sqrt{d}}$$

Note: $\|(c, \dots, c)\|_2 = \sqrt{c^2 + \cdots + c^2} = \sqrt{d}c$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A second feature is scaling the vectors. This accounts for the fact that the softmax function becomes numerically unstable for values with a very large range. The problem is avoided by scaling the values, before feeding them into the softmax function. Scaling normalizes the output vectors to the same norm, independent of the dimension d chosen for the embedding vectors.

Multi-head Attention

A word can mean different things to different neighboring words

Example

3 different types of relationships

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In text multiple, different, relationships exist between different words. Capturing all these relationship in a single representation of the input word is difficult. Therefore, transformer networks employ multiple self-attention mechanisms in parallel, each of which can learn a different kind of relationship. This is called multi-head attention.

Multi-head Attention

Allow the model to learn different relationships separately

- Use t different attention mechanisms
- Each with different query, key, value matrices $W_q^m, W_k^m, W_v^m, m = 1, \ldots, t$

These are called **attention heads**

- Each attention head produces a different output vector

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When using multi-head attention, different query, key and value matrices are learnt for each attention head, allowing to represent different types of syntactic and semantic relationships.

Combining Multi-head Attention Outputs

Concatenate the different outputs $\boldsymbol{y}_{im}, m = 1, \ldots, t$ of multiple attention heads

Apply a linear transformation L to produce an output vector of dimension d

- L is a $td \times d$ matrix
- $\boldsymbol{y}_i = L(\boldsymbol{y}_{i1} \oplus \ldots \oplus \boldsymbol{y}_{it})$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When using multi-head attention, multiple output vectors are produced. These are combined at the output, by applying a linear transformation that reduces the dimension of the concatenated output vectors, with is t*d, back to the standard embedding dimension d. The $\oplus$ symbol denotes concatenation of vectors.

Transformers

Transformers use the self-attention mechanism as building block

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Using a multi-head attention mechanism allows now to establish what is called a transformer block. A transformer block takes the input vectors, applies the multi-head attention, and then performs further processing of the outputs. The additional processing steps are layer normalization and applying multi-layer perceptrons, a basic type of neural networks. We explain these steps next.

Layer Normalization

The outputs $\boldsymbol{y}_i$ of the self-attention are normalized, i.e., centered around the mean and scaled by the standard deviation

$$\overline{\boldsymbol{y}}_i = \frac{\boldsymbol{y}_i - E[\boldsymbol{y}_i]}{\sqrt{V[\boldsymbol{y}_i] + \varepsilon}} * \gamma + \beta$$

γ and β are parameters that are learnt for each layer separately

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Normalization addresses the problem called internal covariate shift. Internal covariate shift refers to the change of distribution of values occurring during training of a neural network, going from one layer to the next. As the network learns and the weights are updated, the distribution of outputs of a specific layer in the network changes. This forces the higher layers to adapt to the drift occurring in lower layers, which slows down learning. Normalizing the inputs to the neural networks, eliminates this problem.

The gamma/beta values used for normalization are learnt for different layers separately but are the same for each layer.

MLP – Multilayer Perceptron

An MLP is one or more linear transformations, each followed by an activation function (e.g., sigmoid, ReLu(x) = max(0,x), tanh)

- Given an input vector $\boldsymbol{x}$, a weight matrix W, and a bias vector $\boldsymbol{b}$, the output of a single layer of a MLP is computed as

$$\boldsymbol{y} = sigmoid(W\boldsymbol{x} + \boldsymbol{b})$$

- the activation function is applied to each component of the vector $W\boldsymbol{x} + \boldsymbol{b}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

MLPs are a simple form of neural networks. They apply first a linear transformation to the input vectors, using a weight matrix W and adding a bias vector b. To the components of the output vector an activation function is applied, to introduce non-linearity. In the original neural network models this function was a sigmoid or tanh function. In current models and standard transformer models it is the Relu function which has better numerical behavior. The Relu function is defined as relu(x) = max(0,x). The MLP is also often called Feed Forward Network (FFN).

MLP in Transformer Networks

To each normalized output vector $\boldsymbol{y}_i$ a MLP is applied **separately**

- Typically, the network has 1 hidden layer
- The dimensions of the vectors processed in the hidden layer should be larger than d
- After applying the MLP again layer normalization is applied

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To each normalized output vector $\boldsymbol{y}_i$ an MLP is applied separately. The MLPs applied to each position have the same parameters. They consist of two linear transformations with an activation in between. Therefore, they have one hidden layer.

While the linear transformations are the same across different positions, they use different parameters from layer to layer. The dimensions of the vectors processed in the MLP, i.e., at the hidden layer, should be larger than d.

Multi-Head Attention (MHA) module and the Feed Forward Network (FFN) module play different roles in the Transformer architecture: The MHA allows the model to jointly attend to information from different subspaces, while the latter increases the non-linearity of the model. In original BERT, the ratio of the number of parameter between the MHA and FFN is always 1:2. To keep that ratio, the inner-layer in the MLP has dimensionality dx4.

Residual Connections

The inputs to the self-attention block are added to its output

$$y := x + y$$

Residual connections

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The inputs to the self-attention block are added to its output through so-called residual connections. Similarly, also for the MLPs.

Residual connections (also called skip connections) are used to allow gradients during backpropagation to flow through a network directly, without passing through non-linear activation functions. Non-linear activation functions, by nature of being non-linear, cause the gradients to explode or vanish (depending on the weights). This results in the problem of vanishing gradients which make it more difficult to learn from the feedback obtained through backpropagation.

Position Embedding

So far, the transformer network does not consider what is the order of words

- The outputs are invariant to permutations of the inputs

Create a second vector of dimension d, that represents the position of the word: position embeddings

- For each position i, create a position vector $\boldsymbol{v}_i$
- To each word vector $\boldsymbol{x}_i$ append the position vector $\boldsymbol{v}_i$ corresponding to its position in the input sequence

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The way the transformer network is defined so far is insensitive to the order of the inputs. In other words, any permutation of the inputs would generate the same relations, which is naturally not how language works. Therefore, in addition the position of words in a sentence needs to be provided as input to the transformer network. The way this is achieved is by learning a position vector, which is the representation of an input position. Each word vector is concatenated with the position vector corresponding to the word's position within the sentence.

Input Length

For transformer networks a fixed length s of the input is assumed

- Maximal length of a sentence
- Position vectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_s$

For training, the transformer needs to see inputs of maximal length, otherwise it cannot learn the weights for higher positions!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to learn position vectors, the length of the sequences (sentences) needs to be fixed, or in other words a maximal sequence length needs to be specified. For learning the parameters this requires that the training data needs to have sequences that have this maximal length, otherwise the position vectors or the higher positions cannot be learnt.

Classification Layer

For text classification, the output of the transformer network is further processed

- Average all the outputs
- Project from dimension d to the dimension l, the number of labels
- Apply a softmax function

The loss function is obtained by comparing the processed output to the labels from the training data

This completes the classifier

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When using a transformer network for text classification, the output vectors are used to produce a prediction for the classification label. In order to do this, a standard approach is to average the output vectors, project them using a linear mapping the the dimension of the label space and apply a softmax function to obtain values that can be interpreted as probabilities.

The predicted values are in the training phase compared to the labels from the training data, and a loss function is calculated from the difference. This is exactly same approach we have taken with our first simple text classifier using a self-attention mechanism.

Using the loss function the parameters can then be learnt using backpropagation. The backpropagation is implemented in frameworks such as pytorch that allows to automatically derive the gradients for the complex function defined by the transformer network.

Putting it all together

The transformer blocks are combined to perform a task, e.g., classification

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This diagram illustrates the overall architecture of a typical transformer network. We see that multiple transformer blocks are applied in sequence (each of which contains a multi-head self-attention mechanism).

Visualization of Layers and Heads

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

From:

https://towardsdatascience.com/deconstructing-bert-part-2-visualizing-the-inner-workings-of-attention-60a16d86b5c1

Finetuning

In general transformer networks are not trained from scratch, but built on top of pretrained networks (**transfer learning**)

For a given task the network or part of it can be retrained

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In practice, given the very large number of parameters in transformer networks, they are usually not trained from scratch, in particular, when using them to build a document classifier. Instead, pretrained networks are used. For example, Huggingface is a platforms that provides a large number of pretrained models. The pretrained network is completed with a classification layer and retrained using the training data for classification. Retraining a network is called finetuning.

This approach has the advantage that general properties of natural language are already encoded in the network parameters and need not to be learnt from scratch every time. This training is in general hugely expensive.

There exist different variants of how fine-tuning is performed, depending whether all are only a part of the parameters of the pre-trained transformer network are modified during retraining.

Final Remarks

For the sake of time, we have been ignoring a number of aspects

- Some standard ways of training (e.g., masking)
- Use of special [CLS] token for sentence representation
- Application to standard NLP tasks (e.g, translation)
- Details of backpropagation in training (e.g., done by pytorch)
- Detailed motivation of technical choices in the architecture based on empirics in NN, e.g., that help to speed up convergence
- The use of Byte-Pair-Encoding tokenizers
- Different types of loss functions

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This introduction to transformer network aimed at highlighting the basic principles underlying their architecture. Many details related to their training, application, and the implementation of deep neural networks have not been discussed, as they are beyond the scope of this lecture.

Document Classification: Summary

Widely studied problem with many applications

- Spam filtering, Sentiment Analysis, Document Search
- Increasing interest
  - Powerful methods using very large corpuses
  - Information filtering on the Internet
- Significant improvements using transformer networks
  - From classifying using bag of words to word sequences
  - Better contextual understanding of language

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Document classification is an area of high interest, both because it is required in a growing number of application domains and because at the same time the classification methods are becoming increasingly powerful due to the availability of very large document corpuses, the growing computational power available and the recent developments around transformer networks.

References

Kowsari, Kamran, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. "Text classification algorithms: A survey." *Information* 10, no. 4 (2019): 150.

The introduction on transformers is based on this tutorial

http://peterbloem.nl/blog/transformers

Visualization tools can be found here

https://towardsdatascience.com/deconstructing-bert-part-2-visualizing-the-inner-workings-of-attention-60a16d86b5c1

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

3.2 LINK-BASED RANKING

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Web is a Hypertext

Web documents are connected through hyperlinks

1. **Anchor text** describes content of referred document
2. **Hyperlink** is a quality signal

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In addition to the textual content, Web documents contain also hyperlinks. A hyperlink can be exploited for information retrieval in two ways:

1. The link is embedded into some text that typically contains relevant information on the content of the document the link is pointing to. Thus, this text can complement the content of the referred document.
2. The link can also be considered as an endorsement of the referenced document by the author of the referring document. Thus, the link can be used as a signal for quality and importance of the referred document.

3.2.1 Indexing Anchor Text

Anchor text is loosely defined as the text surrounding a hyperlink

Example: "You can find cheap cars here."
Anchor text: "You can find cheap cars here"

Anchor text may contain a lot of additional content on the referred page

- It might be a better description of the page than the page content itself

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Anchor text corresponds to the text that is surrounding the link, and not only the text contained as part of the link tag (in the example, the text in the link tag would simply be "here".) The anchor text can contain valuable information on the referred page and thus be helpful in retrieval.

This example illustrates the use of anchor text in retrieval. Often, a home page is very visual and contains often little relevant text content. If we consider a home page, such as the EPFL home page, we probably find many pages pointing to the EPFL home page that very well characterize EPFL, such as pages mentioning topics related to research and technology transfer.

Assume that a malicious Internet user would create a fake EPFL home page. Then chances that such a page is referred by reputed organizations, such as SNF, is very low. On the other hand, pages listing spam pages might point to such a page and reveal its true character. These pages would probably also mention terminology related to spam pages or blacklists, and such text can give indications about the true character of the spam page.

In addition, links to the EPFL home page indicate a higher importance of the page, as compared to other less referenced pages, such as pages containing the EPFL regulations.

Indexing Anchor Text

Can sometimes lead to unexpected effects, e.g., easily spammable

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One of the risks of including anchor text is that it makes pages spammable. Malicious users could create spam pages that point to web pages and try to relate it to contents that serve their interests (e.g., higher the quality of preferred pages by adding links, lower the quality of the undesired page by attaching negative anchor text). That this is. Real phenomenon can be inferred from statistics on the in-degree distribution of Web pages that has been produced.

The figure shows a standard log-log representation of the in-degree vs. the frequency of pages. Normally this relationship should follow a power-law, which shows in a log-log representation as a linear relationship. In real Web data, we see that this power law is violated, and that certain levels of in-degrees are over-represented. This can be attributed to link spamming, which does create moderate numbers of additional links on Web pages.

This is of course only one example of spamming techniques, and Web search engines are in a continuous “battle” against this and other forms of spam.

Scoring of Anchor Text

Score anchor text with a weight depending on the authority of the anchor page's website

- E.g., if we were to assume that content from cnn.com or yahoo.com is authoritative, then trust (more) the anchor text from them

Score anchor text from other sites (domains) higher than text from the same site

- non-nepotistic scoring

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to fight link spamming, when considering anchor text, the text from pages with poor reputation can be given lower weight. We will later introduce methods of how to rank pages based in the hyperlink graph, which is one method to evaluate the reputation of a page.

In order to avoid self-promotion, another method to fight link spamming is to give lower weights to links within the same site (nepotism = promoting your own family members).

3.2.2 PageRank - Hyperlinks as Quality Signal

Bibliometry: analysis of citations in scientific publications

- Citation frequency: how important is a paper of author?
- Co-citation analysis: articles that co-cite the same articles are related
- Impact factor: Authority of sources, such as journals

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The use of links in order to evaluate the quality of information sources has a long tradition, specifically in science. The discipline of bibliometry is fully devoted to the problem of evaluating the quality of research through citation analysis. Different ideas can be exploited to that end:

- The frequency of citations to a paper, indicating how popular or visible it is
- Co-citation analysis in order to identify researchers working in related disciplines
- Analysis of the authority of sources of scientific publications, e.g., journals, publishers, conferences. This measure can then in turn be used to weight the relevance of publications.

All these ideas can also be exploited for any other document collections that have references, in particular, for Web document collections with hyperlinks.

When retrieving documents from the Web, the link structure bears important information on the relevance of documents. A document that is referred more often by other documents through hyperlinks, is likely to be of higher interest and therefore relevance. Therefore, a possibility to rank documents is considering the number of incoming links. Considering the number of incoming links allows to distinguish documents that otherwise would be ranked similarly when relying on text-based relevance ranking.

However, when doing this, also the importance of the link sources can be different. Therefore, not only counting then number of incoming links, but also weighing the links by the relevance of documents that contain these links can help to better assess the quality of a document. The same reasoning of course again applies then for evaluating the relevance of documents pointing to the source of the link and so forth.

Different to scientific publishing, in the Web references are not reliable and therefore simple link counting might not be appropriate. Since 1998 when search engines started to consider links for ranking the phenomenon of link spamming started. Link farms are groups of websites that are heavily linked to one another to boost their ranking.

Link-based Ranking: Idea

Imagine a user doing a **random walk** on Web pages:

- Start at a random page
- At each step, leave the current page along one of the links on that page, with same probability

"In the long run" each page has a long-term visit rate - use this as the page's score

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We introduce now an approach for link-based scoring that considers not only the absolute count of links, but also the quality of the link source. The basic idea is to consider a random walker that visits Web pages following the hyperlinks. At each page the random walker would select randomly among the hyperlinks of the page with uniform probability and move to the next page. When the random walker runs for a long time, it will visit every page with a given probability, which we can consider as a score for ranking the page. This score can be used to control the impact of the outgoing links of a page on the ranking of other pages.

One of the consequences of this model would be that pages that have few in-links, would be relatively infrequently visited. Since link farms and spam pages usually have not many links pointing to them, the expectation is that this approach could reduce their impact on ranking.

On the other hand, popular pages with many incoming links will have a higher impact on ranking, as they have a higher score.

Random Walker Model

$$P(p_i) = \sum_{p_j | p_j \to p_i} \frac{P(p_j)}{C(p_j)}$$

N is the number of Web pages

C(p) is the number of outgoing links of page *p*

$P(p_i)$ probability to visit page p_i, where page p_i is pointed to by pages p_1 to p_N = **relevance**

Result

- If a random walker visits a page more often it is more relevant
- takes into account the number of referrals AND the relevance of referrals

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We provide a formal description of the random walker model. The model is a Markov chain, a discrete-time stochastic process in which at each time-step a random choice is made.

We assign to each page a visiting probability $P(p_i)$. Then the probability that a page p_i is visited depends on the probabilities of the pages with a hyperlink to this page to be visited. For the source pages of the hyperlink the visiting probability is evenly split among all outgoing links. This formulation of the process results in a recursive equation, of which the solution is the steady-state of the process.

Transition Matrix for Random Walker

The definition of $P(p_i)$ can be reformulated as matrix equation

$$R_{ij} = \begin{cases} \frac{1}{C(p_j)}, if\ p_j \rightarrow p_i \\ 0, otherwise \end{cases}$$

$$\vec{p} = (P(p_1), \ldots, P(p_n))$$

$$\vec{p} = R.\vec{p}, \quad \|\vec{p}\|_1 = \sum_{i=1}^{n} p_i = 1$$

The vector of page relevance values is the Eigenvector of the matrix R for the largest Eigenvalue

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to determine the solution to the recursive equation on the probabilities of a random walker to visit a page, we define a transition probability matrix R, which captures the probability of transitioning from one page to another. We also require that the probabilities of visiting a page add up to 1. With this formulation of the problem, the long-term visiting probabilities become the Eigenvector of matrix R. More precisely, they are the Eigenvector with the largest Eigenvalue.

Example

Links from p_1

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

Links to p_1

Link Matrix

$$R = \begin{pmatrix} 0 & 0 & 1 \\ \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 1 & 0 \end{pmatrix}, \quad \vec{p} = \begin{pmatrix} \frac{2}{5} \\ \frac{1}{5} \\ \frac{2}{5} \end{pmatrix}$$

Ranking p_1, p_3 > p_2

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates the computation of the probabilities for visiting a specific Web page. The values $C(p_i)$ correspond to the transition probabilities. They can be derived from the link matrix. The link matrix is defined as $L_{ij}=1$ if there is a link from p_j to p_i. The link matrix is normalized by the outdegree, by dividing the values in the columns by the sum of the values found in the column, resulting in matrix R. The probability of a random walker visiting a node is then obtained from the Eigenvector of this matrix.

Modified Example

Links from p_1

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

Links to p_1

Link Matrix

$$R = \begin{pmatrix} 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 1 & 0 \end{pmatrix}, \quad \vec{p} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

No Ranking

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates a problem with the random walker as we have formulated, the existence of dead ends. We see that there exists a node p_3 that is a "sink of rank". Any random walk ends up in this sink, and therefore the other nodes do not receive any ranking weight. Consequently, also the rank of sink does not. Therefore, the only solution to the equation p=Rp is the zero vector.

Pure Random Walker Does Not Work

The web is full of dead-ends

- Random walk can get stuck in dead-ends
- Makes no sense to talk about long-term visit rates

Teleporting

- At a dead end, jump to a random web page
- At any non-dead end, jump to a random web page with some probability (e.g. 15%)
- Result: Now cannot get stuck locally, there is a long-term rate at which any page is visited

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A practical problem with the random walker is the fact that there exist Web pages that have no outgoing links. Thus, the random walker would get stuck. To address this problem, the concept of teleporting is introduced, where the random walker jumps to a randomly selected Web page with a given probability. If the random walker arrives at a dead end, it will then always jump to a randomly selected page.

Another problem are pages that have no incoming links: they would never be reached by the random walker, and the weight that they could provide to other pages would not be considered. This problem is also addressed by teleporting.

PageRank

Assumption

- random walker jumps with probability 1-q to an arbitrary node
- thus it can leave dead ends and nodes without incoming links are reached

$$P(p_i) = c\left(\frac{(1-q)}{N} + q \sum_{p_j | p_j \to p_i} \frac{P(p_j)}{C(p_j)}\right), c \le 1$$

$$\vec{p} = c((qR + (1-q)E)).\vec{p}, \quad E = \left[\frac{1}{N}\right]_{NxN}$$

$$\vec{p} = c(qR.\vec{p} + \frac{(1-q)}{N}\vec{e}), \quad \vec{e} = (1,...,1)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We give now the formal specification of the random walker with teleporting. At each step, the random walker makes a jump with a probability 1-q and any of the N pages is reached with the same probability. Therefore, an additional term is (1-q)/N is added to the probability for reaching a given page. Reformulating the equation for the probabilities in matrix form, results in adding a NxN Matrix E with all entries being 1/N. This is equivalent to saying that with probability 1/N transitions among any pairs of nodes (including transition from a node to itself) are performed. Since the vector p has norm 1, i.e., the sum of the components is exactly 1, E.p=e. Based on this property, an alternative formulation for the equation can be given. The method described is called PageRank and is used by Google for Web ranking. By modifying the values of the matrix E also a priori knowledge about the relative importance of pages can be added to the ranking algorithm.

Modified Example

$$R = \begin{pmatrix} 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 1 & 0 \end{pmatrix}, E = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{pmatrix}, q = 0.9$$

$$qR + (1-q)E = \begin{pmatrix} \frac{1}{30} & \frac{1}{30} & \frac{1}{30} \\ \frac{29}{60} & \frac{1}{30} & \frac{1}{30} \\ \frac{29}{60} & \frac{14}{15} & \frac{1}{30} \end{pmatrix} \quad \vec{p} = \begin{pmatrix} 0.091 \\ 0.203 \\ 0.705 \end{pmatrix}$$

Ranking $p_3 > p_2 > p_1$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With the modification of rank computation using a source of rank, we obtain for our example a non-trivial ranking which appears to match intuition about the relative importance of the pages in the graph well.

Practical Computation of PageRank

Iterative computation

$$\vec{p}_0 \leftarrow \vec{s}$$
$$while\, \delta > \varepsilon$$
$$\vec{p}_{i+1} \leftarrow qR \bullet \vec{p}_i$$
$$\vec{p}_{i+1} \leftarrow \vec{p}_{i+1} + \frac{(1-q)}{N}\vec{e}$$
$$\delta \leftarrow \left\| \vec{p}_{i+1} - \vec{p}_i \right\|_1$$

ε termination criterion

s arbitrary start vector, e.g., $s = \frac{\vec{e}}{N}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For the practical computation of the PageRank ranking an iterative approach can be used. The vector e is used to add a source of rank. It can uniformly distribute weights to all pages, but it could also incorporate pre-existing knowledge on the importance of pages and bias the ranking towards them. The vector can also be used as initial probability distribution.

Example: ETHZ Page Rank

Doc_ID	Rank_Value	URL
1	0.002536	http://www.ethz.ch/
146	0.002292	http://www.ethz.ch/r_amb/
10	0.000654	http://www.ethz.ch/default_de.asp
35	0.000511	http://www.rereth.ethz.ch/
376124	0.000503	http://computing.ee.ethz.ch/sepp/matlab-5.2-to/helpdesk.html
67378	0.000497	http://computing.ee.ethz.ch/sepp/
59887	0.000485	http://www.computing.ee.ethz.ch/sepp/
89307	0.000485	http://www.isg.inf.ethz.ch/docu/documents/java/jdk1.2.2ref/docs/api/overview-summary.html
216716	0.000485	http://www.isg.inf.ethz.ch/docu/documents/java/jdk1.2/api/overview-summary.html
147932	0.000484	http://isg.inf.ethz.ch/docu/documents/java/jdk1.2ref/docs/api/overview-summary.html
175544	0.000484	http://www.isg.inf.ethz.ch/docu/documents/java/jdk1.2ref/docs/api/overview-summary.html
186766	0.000478	http://isg.inf.ethz.ch/docu/documents/java/jdk1.2/api/overview-summary.html
228634	0.000477	http://isg.inf.ethz.ch/docu/documents/java/jdk1.2.1ref/docs/api/overview-summary.html
228421	0.000464	http://isg.inf.ethz.ch/docu/documents/java/jdk1.2.2ref/docs/api/overview-summary.html
3161	0.00045	http://www.ethz.ch/r_amb/reto_ambuehler.html
215673	0.000447	http://www.vision.ee.ethz.ch/computing/statlinks/sepp.sun/vxl-1.2b-mo/files.html
259672	0.000447	http://www.vision.ee.ethz.ch/computing/statlinks/sepp.sun/vxl-1.2b-mo/globals.html
259671	0.000447	http://www.vision.ee.ethz.ch/computing/statlinks/sepp.sun/vxl-1.2b-mo/functions.html
259670	0.000447	http://www.vision.ee.ethz.ch/computing/statlinks/sepp.sun/vxl-1.2b-mo/annotated.html
259669	0.000447	http://www.vision.ee.ethz.ch/computing/statlinks/sepp.sun/vxl-1.2b-mo/classes.html

Figure 1: Top 20 of ETH Zurich Web Documents

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

These are the top documents from the PageRank ranking of all Web pages at ETHZ (Data from 2001). It is interesting to see that documents related to Java documentation receive high ranking values. This is related to the fact that these documents have many internal cross-references.

Web Search

PageRank is part of the ranking method used by Google

- Compute the global PageRank for all Web pages
- Given a keyword-based query retrieve a ranked set of documents using standard text retrieval methods
- Merge the ranking with the result of PageRank to both achieve high precision (text retrieval) and high quality (PageRank)
- Google uses also many other methods to improve ranking
- Crawling the Web is a technical challenge

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

PageRank is used as one metrics to rank result documents in Google. At the basis Google uses text retrieval methods to retrieve relevant documents and then applies PageRank to create a more appropriate ranking. Google uses also many other methods to improve ranking, e.g., today largely based on personal information collected from users, like search history and pages visited. The details of the ranking methods are trade secrets of the Web search engine providers.

Building a Web Search engine requires to solve several additional problems, beyond providing a ranking system. Efficient Web crawling requires algorithms that can traverse the Web avoiding redundant accesses to pages and techniques for managing large link databases.

The relevance determined using the random walker model corresponds to

1. The number of steps a random walker needs to reach a page
2. The probability that the random walker visits the page in the long term
3. The number of incoming links a random walker can use to visit the page
4. The probability that the random walker will visit once the page

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Consider a random jump matrix with entries 1/3 in the first column and 0 otherwise. It means

1. A random walker can always leave node 1 even without outgoing edges
2. A random walker can always reach node 1, even without incoming edges
3. A random walker can always leave node 2, even without outgoing edges
4. none of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

3.2.3 Hyperlink-Induced Topic Search (HITS)

Key Idea: in response to a query, instead of an ordered list of pages, find **two** sets of inter-related pages:

- **Hub pages** are good lists of links on a subject
  e.g., "World top universities"
- **Authorative pages** are referred recurrently on good hubs on the subject
  e.g., "EPFL"

Best suited for "broad topic" understanding rather than for page-finding queries

- Understand common perception of quality

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic idea of HITS is to apply not a single measure for link-based relevance of a document, but to distinguish two different roles documents can play. Hub pages are pages that provide references to high quality pages, whereas authority pages are high quality pages. The method has been conceived for understand a larger topic in general and obtain an overview of the essential contents related to a given topic. It can nevertheless also be used as an alternative ranking model for Web search, that provides a more refined quality evaluation of Web pages.

Hub-Authority Ranking

Approach

- **Hubs** are pages that point to many/relevant authorities
- **Authorities** are pages that are pointed to by many/relevant hubs

page with large in-degree
e.g., EPFL

Hubs
e.g., QS World, THE , Shanghai Ranking

Authorities
e.g., EPFL, MIT, Stanford, ETHZ

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Hub-authority ranking is, like PageRank, based on a quantitative analysis of the link structure. Different to PageRank two different measures are considered. The number of incoming links as a measure for authority, and the number of links pointing to an authority as a measure for the quality of a hub. The example shows of how in this way authorative pages, such as university home pages, can be distinguished from hub pages, such as portal sites referencing universities.

Computing Hubs and Authorities

Repeat the following updates, for all *p*

$$H(p_i) = \sum_{p_j \in N | p_i \to p_j} A(p_j)$$

$$A(p_i) = \sum_{p_j \in N | p_j \to p_i} H(p_j)$$

Normalize values (scaling)

$$\sum_{p_j \in N} A(p_j)^2 = 1 \qquad \sum_{p_j \in N} H(p_j)^2 = 1$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As in PageRank the approach is to consider the ranking value of pages from which hyperlinks are emanating in the weighting of the influence the hyperlink has on the page it is pointing to. This results directly in a recursive formulation of the ranking values for hub and authority weights. Note that in the HITS method presented here you find subtle differences how those equations are formulated, as compared to PageRank:

1. The weights are not split among the outgoing links, but each link transfers the while hub or authority weight from the originating page
2. Since the weights are not split, the ranking values need to be normalized
3. The normalization uses L2 norm, and not L1 norm as for PageRank.

HITS Algorithm

$$n := |N|;\ (a_0, h_0) := \frac{1}{\sqrt{n}}\big((1, \dots, 1), (1, \dots, 1)\big); \mathrm{l} = 0$$

$$\begin{aligned}
&\textit{while } l < k \\
&\qquad l := l + 1 \\
&\qquad a_l := \left(\textstyle\sum_{p_i \to p_1} h_{l-1,i}, \dots, \sum_{p_i \to p_n} h_{l-1,i}\right) \\
&\qquad h_l := \left(\textstyle\sum_{p_1 \to p_i} a_{l,i}, \dots, \sum_{p_n \to p_i} a_{l,i}\right) \\
&\qquad (a_l, h_l) := \left(\frac{a_l}{|a_l|_2}, \frac{h_l}{|h_l|_2}\right)
\end{aligned}$$

In practice, k = 5 iterations sufficient to converge!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Similarly, as for PageRank, the equations can be solved using iteration. Here we show a possible realization of such an iterative computation, using uniformly distributed weights for initialization.

Convergence of HITS

$n \times n$ link matrix L^t

	1	2	3
1	0	1	0
2	1	1	1
3	1	0	0

Up to normalization

$h = L^t a, a = Lh$, thus
a is an Eigenvector of LL^t
h is an Eigenvector of $L^t L$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When formulating the HITS equations in matrix form, using the link matrix L, we see that the authority and hub weights correspond to the Eigenvectors of the matrices LL^t and $L^t L$ This shows that the iterative computation with normalization of the hub and authority values will converge to the principal Eigenvectors of the matrices LL^t and $L^t L$

When computing HITS, the initial values

1. Are set all to 1
2. Are set all to $\frac{1}{n}$
3. Are set all to $\frac{1}{\sqrt{n}}$
4. Are chosen randomly

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

If the first column of matrix L is (0,1,1,1) and all other entries are 0 then the authority values

1. $(0,1,1,1)$
2. $\left(0, 1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}\right)$
3. $\left(1, 1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}\right)$
4. $(1,0,0,0)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Practical Implementation

Apply HITS in the context of a query

- Given a query (e.g., "EPFL"), obtain all pages mentioning the query: call this the **root set** of pages.

Add page that either

- points to a page in the root set, or
- is pointed to by a page in the root set.

Use this set as **base set**

- Compute HITS on the base set

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One possible application of HITS is to compute the ranking on the complete Web Graph, as it is done with PageRank. Another way to use it (and this is how it was initially conceived), is to apply it in the context of a given query, to rerank the results by promoting results with high authority and hub values. In order to perform this operation, first all results for a query are retrieved (using a standard text retrieval model). Then the neighboring pages (either pointing to a result page, or referred by a result page) are added to the set of pages, which is then called the base set. HITS is then computed on the base set. This makes sense, as in this way we both consider referred pages and referring pages for the relevant documents, which helps to identify both hubs and authorities.

HITS Conclusions

Potential issues

- Mutually Reinforcing Affiliates: clusters of affiliated pages/sites can boost each others' scores
- Topic Drift: off-topic pages can cause off-topic "authorities" to be returned

Social Network Analysis

- PageRank and HITs are examples of Social Network (SN) Analysis algorithms
- SNs contain a lot of other interesting structure

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

HITS suffers from similar potential problems related to the manipulation of the link structure through link spamming as PageRank. In addition, when performing a broad topic search and computing a base set for analysis, topic drift may occur, e.g., through the introduction of off-topic hubs. This is a problem that is similar to the issues of topic drift in pseudo-relevance feedback that we have observed earlier.

Both, HITS and PageRank are examples of social network analysis algorithms. We will introduce later other types of algorithms for this purpose, aiming at community detection.

For efficient implementation, link-based ranking algorithms require an efficient representation of the Web graph. This is a topic that we will explore next.

3.3 GRAPH MINING

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Mining Graphs

Data Mining can be performed on different types of data

- Structured data: tables, graphs
- Unstructured data: text, images, sensor data

Graph data is increasingly important

- Social networks and Web
- Knowledge Graphs
- Scientific data on networks
- Graph structure can also be inferred from distance measures (e.g., for documents)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Data mining is applied for a wide range of both structured and unstructured data. It has traditionally evolved from analyzing large transactional databases, typically for business applications,. Structured data graph data is playing an increasingly important role in data mining, due to a growing number of graph data sources. One area where graph data play an obvious role is in social networks, where social interactions and relationships are modelled as graphs. This availability of graph data and the need to understand the structure of large graphs have been driving factors in the development and wide-spread application of graph mining algorithms.

It is also worthwhile to note that graph mining algorithms can be applied to graphs that are generated from other data types. For example, document similarity measures based on text embeddings could be used to generate graph structures for document collections.

Graphs and Clustering

Graphs often contain structure

– Clusters (also called communities, modules)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

It is widely observed that natural networks contain structure. This is true for social networks (e.g., social media platforms, citation networks), as well as for many natural networks as we find them in biology. Graph-based clustering aims at uncovering such hidden structures.

Social Network Analysis

Clusters (communities) in social networks (Twitter, Facebook) related to

- Interests
- Level of trust

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In social networks mining community structures is particularly popular. Consider the social network neighborhood of a particular social network user, i.e., all other social network accounts that are connected to the user, either through explicit relationships, such as a follower graph, or through interactions such as likes or retweets. Typically, the social neighborhood would decompose into different groups, depending on interests and social contexts. For example, the friends from high school would have a much higher propensity to connect to each other, than with members of the family of the user. Thus, they are likely to form a cluster. Similarly, other groups with shared interest or high mutual level of trust would form communities. Graph mining is a tool to detect these types of communities.

Many tasks can benefit from a reliable community detection algorithm. Online Social Networks rely on their underlying graph to recommend content, for example relevant jobs. Knowing to which community a user belongs can improve dramatically the quality of such recommendations.

Another typical use of community detection is to identify media influencers. Community detection can first be used to detect communities that share in social networks common interests or beliefs (e.g., in the discussion on climate change we might easily distinguish communities that are climate deniers and climate change believers), and then the main influencers of such communities could be identified.

Use of Community Structures: Social Science

Call patterns in Belgium cell phone network

– Two almost separate communities

V.D. Blondel et al, *J. Stat. Mech.* P10008 (2008).

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In 2008 Vincent Blondel and his students have started applying a new community detection algorithm to the call patterns of one of the largest cell phone operators in Belgium. It was designed to identify groups of individuals who regularly talk with *each other* on the phone, breaking the whole country into numerous small and not so small communities by placing individuals next to their friends, family members, colleagues, neighbors, whom they regularly called on their cell phone. The result was somewhat unexpected: it indicated that Belgium is broken into two huge communities, each consisting of many smaller circles of friends. Within each of these two groups the communities had multiple links to each other. Yet, these communities never talked with the communities in the other group (guess why?). Between these two large groups we find a third, much smaller group of individuals, apparently mediating between the two parts of Belgium.

Which of the following graph analysis techniques do you believe would be most appropriate to identify communities on a social graph?

A. Cliques
B. Random Walks
C. Shortest Paths
D. Association rules

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The intuition behind community detection is that the heavily linked components of the graph belong to the same community. Thus, a community is a set of nodes that has many connections among themselves, but few to other parts of the network. These communities form clusters in the network.

Types of Community Detection Algorithms

Hierarchical clustering

- iteratively identifies groups of nodes with high similarity

Two strategies

- **Agglomerative algorithms** merge nodes and communities with high similarity
- **Divisive algorithms** split communities by removing links that connect nodes with low similarity

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In general, community detection algorithms are taking a hierarchical approach. The idea is that in a community smaller sub-communities can be identified, till the network decomposes into individual nodes. In order to produce such a hierarchical clustering, two approaches are possible: either by starting from individual nodes, by merging them into communities, and recursively merge communities into larger communities till no new communities can be formed (agglomerative algorithms), or by decomposing the network into communities, and recursively decompose communities till only individual nodes are left (divisive algorithms).

In the following we will present one representative of each of the two categories of algorithms:

1. The Louvain Algorithm, an agglomerative algorithm
2. The Girvan-Newman algorithm, a divisive algorithm

3.3.1 Louvain Modularity Algorithm

Agglomorative Community Detection

- Based on a measure for community quality (**Modularity**)
- greedy optimization of modularity

Overall algorithm

- **first** small communities are found by optimizing modularity **locally** on all nodes
- then each small community is **grouped** into one new community node
- **Repeat** till no more new communities are formed

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The Louvain algorithm is essentially based on the use of a measure, modularity, that allows to assess the quality of a community clustering. The algorithm performs greedy optimization of this measure. The basic idea is straightforward: initially every node is considered as a community. The communities are traversed, and for each community it is tested whether by joining it to a neighboring community, the modularity of the clustering can be improved. This process is repeated till no new communities form anymore.

A short remark on terminology: in mathematics the usual terminology for graph nodes is “vertex”. We will in the following continue to use the term “node” for “vertex”, as it is custom in computer science.

Measuring Community Quality

Communities are sets of nodes with many mutual connections, and much fewer connections to the outside

Modularity measures this quality: the higher the better

$$\sum_{C \in \text{Communities}} (\#\text{edges within } C - \text{expected } \#\text{edges within } C)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to measure the quality of a community clustering, modularity compares the difference between the number of edges that occur within a community with the number of edges that would be expected in a random subset of nodes of the same size and randomly distributed edges.

Expected Number of Edges

Graph with unweighted edges

- m = total number of edges
- k_i = number of outgoing edges of node i (degree)

Observation: there exist $2m$ "edge ends"

distributes k_i edge ends uniformly to other nodes

$\frac{k_j}{2m-1}$ edges end up in node j

Probability of an edge of k_i ending up in node j: $k_i \frac{k_j}{2m-1} \approx \frac{k_i k_j}{2m}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We now determine the number of edges we expect between nodes with given degrees ki, if edges were purely randomly allocated.

How many edges would we observe if the connections on the graph were generated randomly? To answer this question, we reason as follows: select one of the nodes i with degree ki. What is the probability that this edge connects to another node j with weight kj? If there are a total of m edges in the network, there are 2m edge ends (since each edge ends in two nodes). If the edge ends are uniformly distributed and there are 2m-1 remaining edge ends in the graph, the probability to connect exactly to node j would be kj/2m-1. Applying the same argument to all outgoing edges of node i, node j will connect to node i with probability ki*(kj/2m-1). For large m we can replace 2m-1 by 2m.

Modularity

Modularity measure Q

- A_{ij} = effective number of edges between nodes i and j
- C_i, C_j = clusters of nodes i and j

Expected number of edges

$$Q = \frac{1}{2m}\sum_{i,j}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j)$$

Effective number of edges

Properties

- Q in [-1,1]
- 0.3-0.7 < Q means significant community structure

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given the expected number of edges we can now formulate the modularity measure as the total difference of number of expected and effective edges for all pairs of nodes from the same cluster. The delta function assures that only nodes belonging to the same cluster are considered, since it returns 1 when ci = cj.

Due to normalization the measure returns values between -1 and 1. In general, if modularity exceeds a certain threshold (0.3 to 0.7) the clustering of the network is considered to exhibit a good community structure.

Locally Optimizing Communities

What is the modularity gain by moving node i to the communities of any of its neighbors?

- Test all possibilities and choose the one that increases modularity the most, if it exists

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given the modularity measure, the next question is now how to use it to infer the communities. This is performed through local optimization. The algorithm sequentially traverses all nodes of the network, and for each node checks how the modularity can be increased maximally by having the node joining the node of a neighboring community. If such a neighboring node exists, the node will join the community.

Example

Initial modularity Q = 0

Start processing nodes in order

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We illustrate the algorithm for a simple example. Initially, the modularity is zero since all nodes belong to different communities. We start now to process the nodes in some given order, e.g., by increasing identifiers.

Example: Processing Node 1

Joining node 1 to node 2

- Q = 1/2*7 (1 – 2*2/2*7) = 1/14 * 10 /14 > 0

Joining node 1 to node 3

- Q = 1/2*7 (1 – 2*3/2*7) = 1/14 * 8/14 > 0

New modularity: 1/14 * 10 /14

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Node 1 can either join node 2 or 3, it's two neighbors. To decide which is better, we compute modularity after the join. We see that joining node 2 results in a higher modularity. We can interpret this as follows: since node 3 has more connections, it is more likely to be randomly connected to node 1, thus connecting to node 2 is less likely to be the result of a random connection.

Example: Processing Nodes 2 and 3

Joining node 2 to node 3 (leaving community of node 1)

- no improvement

Joining node 3 to community {1,2} (via node 1 or 2)

- Q = 1/14 (3 – 4/14 – 6/14 – 6/14) = 1/14 * 26 / 14

Joining node 3 to node 4

- Q = 1/14 10/14 + 1/14 (1 – 9/14) = 1/14 * 15/14

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Node 2 will not change the community, as the only alternative would be node 3. But for the same reasons node 1 did not join node 3, also node 2 does not. The next node is node 3: this node has two choices, either to join community {1,2}, or to join node 4. In the first case we obtain one larger community, and in the second case two smaller communities. Computation of modularity reveals that joining join community {1,2}, gives a higher modularity.

Example: Processing Nodes 4, 5 and 6

Joining node 4 to community {1,2,3} via 3

- Q = 1/14 (4 - 4/14 - 6/14 - 6/14 – 6/14 – 6/14 - 9/14) = 1/14 * 19/14

Joining node 4 to node 5

- Q = 1/14 * 26/14 + 1/14 * (1 – 6/14) = 1/14 * 34/14

Finally, also node 6 will join the second community

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For similar reasons as before, node 4 will join node 5 and finally node 6 will join nodes {4,5}. This completes the first iteration.

Example: Merging Nodes

Now that all nodes have been processed, we merge nodes of the same community in a single new node and restart processing

Will the two remaining nodes merge? Answer: yes

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the next iteration, the current community nodes are collapsed into new nodes representing the communities, and the algorithm is re-run with the new resulting graph. Obviously, now the two remaining nodes will merge into a single community, as the modularity will change from zero to a positive value. Then the algorithm terminates.

Modularity clustering will end up always with a single community at the top level?

A. true
B. Only for dense graphs
C. Only for connected graphs
D. never

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Modularity clustering will end up always with the same community structure?

A. true
B. Only for connected graphs
C. Only for cliques
D. false

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Modularity to Evaluate Community Quality

Modularity can also be used to evaluate the best level to cutoff of a hierarchical clustering

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Apart from detecting communities, the modularity measure can also be used to evaluate the quality of communities in a hierarchical clustering. This can be done independently of how the clustering has been constructed.

By evaluating modularity at each level of the hierarchy, an optimal modularity value can be determined. This optimal value indicates which are the highest quality communities, and lower levels in the tree hierarchy can be pruned.

Louvain Modularity - Discussion

Widely used in social network analysis and beyond

- Method to extract communities from very large networks very fast

Complexity: O(n log n)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Louvain modularity clustering is widely used method for social network clustering, mainly because of its good computational efficiency. It runs in O(n log n), which makes it applicable for very large networks, for example, for social networks resulting from large Internet platforms, such as social networking sites or messaging services.

3.3.2 Girvan-Newman Algorithm

Divisive Community Detection

- Based on a **betweenness measure** for edges, measuring how well they separate communities
- Decomposition of network by splitting along edges with highest separation capacity

Overall algorithm

- Repeat until no edges are left
  - Calculate betweenness of edges
  - Remove edges with highest betweenness
  - Resulting connected components are communities
- Results in hierarchical decomposition of network

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We now introduce a second algorithm for community detection, that belongs to the class of divisive algorithms. This algorithm uses he betweenness measures that quantifies the capacity of an edge to separate the network into distinct subnetworks. Note that while for an agglomerative algorithm we used a measure that quantifies how well communities are connected, for a divisive algorithm we use a measure that quantifies how well communities can be separated.

Using the betweenness measure, the algorithm will recursively split the network into smaller networks, till arriving at individual nodes. This will result in a hierarchical clustering.

Edge Betweenness Centrality

Edge betweenness centrality: fraction of number of shortest paths passing over the edge

$$betweenness(v) = \sum_{x,y} \frac{\sigma_{xy}(v)}{\sigma_{xy}}$$

where

σ_{xy}: number of shortest paths from x to y
$\sigma_{xy}(v)$: number of shortest paths from x to y passing through v

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Betweenness centrality is an indicator of a node's centrality in a network. It is equal to the number of shortest paths from all nodes to all other nodes that pass through that node. A node with high betweenness centrality has a large influence on the transfer of items through the network, under the assumption that item transfer follows the shortest paths. The concept finds wide application, including computer and social networks, biology, transport and scientific cooperation.

As an alternative measure, one could also consider *random-walk betweenness*. A pair of nodes x and y are chosen at random. A walker starts at x, following each adjacent link with equal probability until it reaches y. Random walk betweenness r_{xy} is the probability that the link $x \rightarrow y$ was crossed by the walker after averaging over all possible choices for the starting nodes x and y. Different to betweenness this measure does consider all paths between the two nodes, not only the shortest paths. However, this measure is expensive to compute.

Example

$$\sigma_{xy}(v)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In this example network we illustrate the value of $\sigma_{xy}(v)$ for some selected edges. For example, for the edge 1-2 there is one shortest path between 1 and 2 that traverses the edge 1-2, thus the value is 1. For 1-3 there are shortest paths from the remaining 12 nodes in the network (except node 2) to node 1 that must pass through this edge, thus the betweenness value is 12. For edge 3-7 we have shortest paths reaching both nodes 1 and 2, thus the betweenness value of that edge is significantly higher than of 1-3. For edge 7-8 we have 7 nodes in each of the two subnetworks on the left and on the right (including the nodes 7 and 8). Among each pair of nodes there exists exactly one shortest path. Therefore, the value of $\sigma_{xy}(v)$ is 49.

Betweenness can be considered as a concept that is dual to connectivity. This is illustrated by the two graphs that result from real phone call networks. On the left-hand side, we see the indication of the strengths of connections among the nodes. Communities are tightly connected by such links. On the right-hand side, we see the betweenness measure. Now the links that are connecting different communities have a high strength. This can be interpreted in different ways: on the one hand traffic from one community to another has to traverse these links, on the other hand if such links are cut, communities are separated.

Next, we illustrate the principle of the Girvan-Newman algorithm. It proceeds by recursively removing edges with highest betweenness from the network.

In Step 1 it removes one edge (in the middle) that had the highest betweenness value, resulting in two communities. Next the edges connected to nodes 7 and 8 are removed, and in the third and fourth step the network decomposes completely. By overlaying the communities that have resulted from each step we obtain the final hierarchical clustering.

As the graph structure changes in every step, the betweenness values need to be recomputed in every step. This results in the main cost of the algorithm. We will next explore the question of efficient computation of betweenness.

The algorithm has been applied in many contexts, due to the computational cost mostly on smaller graphs resulting from social science studies.

In fact, the origin of the Girvan-Newman algorithm can be traced back to a specific social science study. In this study a network of 34 karate club members was studied by the sociologist Wayne Zachary Links capture interactions between the club members outside the club. The white and gray nodes denote the two communities that resulted after the club decided to split following a feud between the group's president and the coach. The split between the members closely follows the boundaries of the two communities. In the community graph we can identify one outlier, node number 9, where the algorithm wrongly assigns the member at the time of conflict. Node 9 was completing a four-year quest to obtain a black belt, which he could only do with the instructor, node 34.

This example has been widely used as a benchmark to test community detection algorithms.

We describe now the approach for computing the betweenness values. It is based on a breadth-first search (BFS) starting at every node in the graph. For a given node, e.g., node A, the other nodes are arranged in levels according to increasing distance from the node.

Computing Betweenness – Path Counting

Count the number of shortest paths from A to all other nodes of the network

$$\#shortestpaths(A - X) = \sum_{P\ parent\ of\ X} \#shortestpaths(A - P)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In a first phase, we count the number of shortest paths that are leading to each node, starting from node A. To do so, the algorithm can at each level reuse the data that has been computed at the previous level. The number of shortest paths leading to a given node corresponds to the sum of number of shortest paths leading to each of its parent needs. For example, the shortest paths leading to node F, are all shortest paths leading to nodes B and C. Note that in this way paths that are not shortest paths are ignored, like the path A-B-C-F.

Computing Betweenness – Edge Flow

Edge Flow

- 1 unit of flow from A to each node
- Flow to be distributed evenly over all paths
- Sum of the flows from all nodes though an edge equals the betweenness value

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to compute the betweenness values for edges, we have to aggregate the contributions of all paths between arbitrary pairs of nodes. To do this the following model is adopted. From each node a flow is emanating to each other node of the graph, which has a total volume of 1. This flow is evenly distributed among all shortest paths, that lead from the source node to the target node. Note that these different shortest paths may be overlapping, i.e., have common edges. Therefore edges that are part of different shortest paths will receive contributions from all the shortest paths passing through them. Finally, the flows will be aggregated for the flows emanating from all nodes, ot obtin the betweenness value of the edge.

Here we illustrate the second phase of the betweenness computation, computing the flow values related to one node, in the example node A. For each node one must consider the flow that is arriving at the node, plus the flows that are passing through the nodes to another target node. For computing the aggregate edge flows, the algorithm will start at the farthest node, in the example node K, which has no shortest paths emanating. Therefore, the only flow it receives is the one assigned to itself, i.e., a flow of 1. Since the node can be reached through different shortest paths, which are represented by the incoming edges from the nodes of the next higher level, this flow is distributed proportionally to the number of shortest paths leading to the parent nodes. In the case of node K, there exist 3 shortest paths leading to its two parent nodes. Therefore, the flows are evenly distributed. This results in flows of ½ for the incoming edges of node K.

For the nodes at the higher levels, the flows destined for the node are summed up with the flows destined for its descendant nodes that have been computed before. For example, node I has an outgoing flow of ½ plus a flow of 1 destined for itself. Therefore, it has to distribute a total flow of 3/2 over its incoming edges. This distribution must be done proportionally to the number of shortest paths arriving at its parent nodes. Since 2 shortest paths arrive at node F and 1 shortest path arrives at node G, the flow of 3/2 is split at a ratio of 2:1 among the two incoming edges of node I. The analogous reasoning is applied to the other nodes.

In general, each node distributes tribute a weight of $1 + \sum_{C\ child\ of\ X} edge_weight(X - C)$ over the incoming edges, i.e., the flow of 1 arriving at the node plus the sum of the flows on the outgoing edges. The distribution is done according to the ratios of the shortest paths arriving at the parent nodes.

Algorithm for Computing Betweenness

1. Build one BFS structure for each node
2. Determine edge flow values for each edge using the previous procedure
3. Sum up the flow values of each edge in all BFS structures to obtain betweenness value
   - Flows are computed between each pairs of nodes → final values divided by 2

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Once the edge flows have been computed for all nodes, the resulting values are summed up. Since for each shortest path two flows have been generated, corresponding to the traversal of the path in the two directions, the final aggregate flow is divided by 2.

$\sigma_{xy}(v)$ of edge 3-4 is …

A. 16
B. 12
C. 9
D. 4

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When computing path counts for node 1 with BFS, the count at 6 is …

A. 1

B. 2

C. 3

D. 4

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Girvan-Newman Discussion

Classical method

- Works for smaller networks

Complexity

- Computation of betweenness for one link: $O(n^2)$
- Computation of betweenness for all links: $O(L\ n^2)$
- Sparse matrix: $O(n^3)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The Girvan-Newman algorithm is the classical algorithm for community detection. Its major drawback is its scalability. The flow computation for one link has quadratic cost in the number of nodes, since it is computed for each pair of nodes. If we assume sparse networks, where the number of links is of the same order as the number of nodes, the total cost is cubic. This was also one of the motivations the inspired the development of the modularity-based community detection algorithm.

References

The slides are loosely based also on:

- http://barabasilab.neu.edu/courses/phys5116/

Papers

- Blondel, Vincent D., et al. "Fast unfolding of communities in large networks." *Journal of statistical mechanics: theory and experiment* 2008.10 (2008): P10008
- Girvan, Michelle, and Mark EJ Newman. "Community structure in social and biological networks." *Proceedings of the national academy of sciences* 99.12 (2002): 7821-7826.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Part 4: Information Extraction

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.1 NAMED ENTITY RECOGNITION

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Knowledge Graphs (Google 2012)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One model for representing ontologies that has gained large popularity recently are knowledge graphs. They are based on a graph-based representations of basic first order logic constructs, entities, which are elements from given domains, relationships, which are binary relations and attributes which are relations among entities and values from a standard data structure.

RDF – Resource Description Framework (W3C 2004)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In fact, long before knowledge graph have become popular, a similar model has been developed by the W3C, the Resource Description Framework. It has the same basic concepts as knowledge grpahs, together with some additional modeling primitives. We will provide an overview of RDF in the following.

Entity-Relationship Model

The Entity Relationship Model: Toward a Unified View of Data, Peter Pin-Shan Chen, **1976**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The approach of using concepts from first order logic for modeling reaches however far back in history. In the area of database management systems, the entity-relationship model has been established as a standard approach for data modeling already in 1976. The main difference is in the intended use of the model at the time. It was used to provide a conceptual model of an application domain and then to derive in a mostly automated way a database schema (in other words data structures) in the data model supported by the target database management system. This model typically is the relational data model.

Populating Knowledge Bases

Manual creation of knowledge bases is expensive

Can we produce them automatically?

Idea: Extract knowledge from documents

Challenge: Knowledge is encoded in natural language

Objectives

- Automated or accelerated creation of knowledge bases
- Support for structured search on documents

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Traditionally knowledge bases are created manually, either by experts (e.g., WordNet) or by crowd-sourcing (e.g., WikiData). This is expensive. In the case of WordNet, it took tens of years to construct the knowledge base, in the case of WikiData (resp. WikiPedia) we all know about the notorious difficulty to finance this endeavor. Therefore, an interesting question is whether such knowledge bases could not be automatically constructed.

For automatic construction we can exploit data that is digitally available, e.g., all documents accessible on the Web. These documents encode massive human knowledge in natural language. The challenge is to extract such knowledge by analyzing natural language text, which is not an easy problem.

The results would, however, be immensely useful. First, we could create massive knowledge bases in a nearly automated way, and furthermore these knowledge bases could be used to annotate documents, for supporting more expressive and precise searches and analyses.

Information Extraction

From text to knowledge

The New York Times

MARTIN LUTHER KING IS SLAIN IN MEMPHIS; A WHITE IS SUSPECTED; JOHNSON URGES CALM

Who are the entities?

What are their attributes?

How are they related?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For investigating knowledge extraction, more commonly called information extraction, from textual content, we can consider the different constituents of a knowledge graph separately: entities, attributes and relationships. We will now introduce methods for extracting entities and then for establishing relationships among entities and with attributes.

4.1.1 Key Phrase Extraction

Idea: key phrase extraction is "the automatic selection of important and topical phrases from the body of a document" (Turney, 2000)

- Document summarization, search and indexing
- Document classification and opinion mining

EPFL is one of the two **Swiss Federal Institutes of Technology**. With the status of a **national school** since 1969, the **young engineering school** has grown in many dimensions, to the extent of becoming one of the most **famous European institutions** of **science and technology**. Like its **sister institution** in **Zurich**, **ETHZ**, it has three **core missions**: **training**, **research** and **technology transfer**. Associated with several specialised **research institutes**, the two **Ecoles Polytechniques** (**Institutes of Technology**) form the **EPF domain** , which is directly dependent on the **Federal Department of Economic Affairs, Education and Research (EAER)**.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A first type of information extraction method is key phrase extraction. Key phrase extraction aims at identifying words and phrases that are typical for a document and characterize important concepts that occur in a document. Key phrase extraction has been developed to support document summarization, where the key phrase gives an overview of the key concepts of a document. Key phrase extraction supports also document search and indexing, where key phrases are used to index documents. Moreover, key phrases can also provide useful features for document classification, i.e., key phrase extraction can be considered as a feature selection method. In the example text we see the possible outcome of key phrase extraction, with all identified key phrases marked in bold.

Keyphrase Extraction Methods

Approach: generate candidate phrases and rank them

Candidate phrases

- Remove stopwords
- Use word n-grams
- Consider part-of-speech tags (POS)

Baseline ranking approach

- rank candidate phrases of the document according to their tf-idf value

Advanced approaches

- Use of many structural, syntactic features of the documents
- Use of external resources, such as Wikipedia, Wordnet
- Use of transformer models

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic approach to key phrase extraction uses principles well known from information retrieval. As in information retrieval stopwords are excluded. Key phrase candidates can be all word n-grams in the remaining text. Furthermore, part-of-speech tags can be used to further select the candidates, e.g., for excluding all verb phrases.

In order to assess whether a candidate phrase is characteristic for a document, tf-idf ranking is a possible approach. As in IR a candidate phrase is considered as relevant for a document if it is at the same time frequent and specific. Apart from that, many heuristics have been developed to refine this approach, considering additional features of the document. For example, a phrase in the title or a header could be considered as more relevant. External knowledge bases could be used to check whether a phrase corresponds to a commonly known concept. Recently, also transformer models have been applied for the task of key phrase extraction.

Use of Keyphrase Extraction

Creation of domain-specific thesaurus and taxonomy

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Key phrase extraction can be used as an initial step to create a domain-specific thesaurus or taxonomy. In the example, we see a thesaurus that has been constructed for the food domain, based on key phrase extraction. For the phrase “junk food” many synonyms and near-synonyms have been identified. In practice a human expert would not be able to extract reliably all different types of mentions of such a concept.

As a result, this allows to increase recall for retrieval of documents that refer to this concept.

Use of Keyphrase Extraction

Document classification and search

Evolution of Structure in the Intergalactic Medium and the Nature of the Ly-alpha Forest ver. 3

HongGuang Bi, Arthur F. Davidsen

Astrophysics

We have performed a detailed statistical study of the evolution of structure in a photoionized intergalactic medium (IGM) using analytical simulations to extend the calculation into the mildly non-linear density regime found to prevail at z = 3. Our work is based on a simple fundamental conjecture: that the probability distribution function of the density of baryonic diffuse matter in the universe is described by a lognormal (LN) random field. The LN field has several attractive features and follows plausibly from the assumption of initial linear Gaussian density and velocity fluctuations at arbitrarily early times. Starting with a suitably normalized power spectrum of primordial fluc- tuations in a universe dominated by cold dark matter (CDM), we compute the behavior of the baryonic matter, which moves slowly toward minima in the dark matter potential on scales larger than the Jeans length. We have computed two models that succeed in matching observations. One is a non-standard CDM model with Omega=1, h=0.5 and \Gamma=0.3, and the other is a low density flat model with a cosmological constant(LCDM), with Omega=0.4, Omega_Lambda=0.6 and h=.65. In both models, the variance of the density distribution function grows with time, reaching unity at about z=4, where the simulation yields spectra that closely resemble the Ly-alpha forest absorption seen in the spectra of high z quasars. The calculations also successfully predict the observed properties of the Ly-alpha forest clouds and their evolution from z=4 down to at least z=2, assuming a constant intensity for the metagalactic UV background over this redshift range. However, in our model the forest is not due to discrete clouds, but rather to fluctuations in a continuous intergalactic medium. (This is an abreviated abstract; the complete abstract is included with the manuscript.)

Intergalactic medium | Lambda-CDM model | Opacity | Cold dark matter | Mean mass density | Quasar | Filling fraction | Dark matter | Peculiar velocity | Jeans length | Ultraviolet background | Voigt profile | Ionization | Lyman-alpha forest | Random Field | Neutral hydrogen gas | Absorption line | Intensity | Cold-plus-hot dark matter | Gunn-Peterson effect | Confinement | Line of sight | Intercloud medium | Hydrodynamical simulations | Dark matter model | Cosmological constant | Proximity effect | Velocity fluctuations | Statistics | Hydrostatics | Cosmological model | Photoionized Intergalactic Medium | Line thermal broadening | Absorption feature | Zeldovich approximation | Curve of growth | Photoionization | Autocorrelation | Hopkins Ultraviolet Telescope | Lyman Limit System | Cosmic Background Explorer | Fine structure | Hot dark matter | Big bang nucleosynthesis | Density contrast | Expansion of the Universe | Diffuse gas | Cooling | HI column density | Intergalactic gas | Light curve | Halo model | Numerical simulation | Magnet | Deuterium Abundance | Phase space caustic | Graph | Gaussian noise | Equivalent width | Two-point correlation function | Shock wave | Cluster of galaxies | Jeans mass | Primordial fluctuations | N-body simulation | Line spread function | Infall velocity | IGM temperature | Neutral hydrogen absorber | Speed of sound | Intergalactic clouds | Galactic disks | Cosmological parameters | Collapsing clouds | Time Series | Recombination rate | Collisional ionization | Cross-correlation | Hubble parameter | Large scale structure | Hubble Space Telescope | Hierarchical clustering | Exponential function | HIRES spectrometer | Signal to noise ratio | Mass distribution | Density parameter | Critical density | Cosmological redshift | Spectral resolution | Spectral line | Fluid dynamics | Cosmic microwave background | Fast Fourier transform | Sunyaev-Zel'dovich effect | Gravitationally lensed quasars | The early Universe | Hydrogen atom | Matter power spectrum | Simulations | Redshift | Mass | Fluctuation | Mathematics (under construction) | Velocity | Field | Temperature | Potential | Universe | Baryons | Gas | Picture | Pressure | Optical depth | Units | Amplitude | Probability density function | Theory | Measurement | Resolution | Geometry | Droplet | Wavelength | Dispersion | Object | Order of magnitude | Polynomial | Ion | Atom | Particles | Metals | Resonance | Probability | Frequency | Electron | Cross section | Materials

sciencewise.info

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an example of using key phrase extraction for scientific documents in physics. Many highly specific concepts are identified automatically and allow a scientist to more precisely filter and search the documents. You can try this out on your own at sciencewise.info.

4.1.2 Named Entity Recognition (NER)

Task: Find and classify names of people, organizations, places, brands etc. that are mentioned in documents

organization organization

EPFL is one of the two **Swiss Federal Institutes of Technology**. With the status of a national school since 1969, the young engine georegion has grown in many dimensions, to the extent of becoming c city ost famous **European** institutions of science and technology. Like its sister institution in **Zurich**, **ETHZ**, it has three core missions: training, research and technology transfer. Associated with several specialised research institutes, the two **Ecoles Polytechniques** (**Institutes of Technology**) form the **EPF domain** , which is directly dependent on the **Federal Department of Economic Affairs, Education and Research (EAER)**.

EPFL is located in **Lausanne** in **Switzerland**, on the shores of the largest lake in **Europe**, **Lake Geneva** and at the foot of the **Alps** and **Mont-Blanc**. Its main campus brings together over 11,000 persons, students, researchers and staff in the same magical place.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Named entity recognition is a more specific task than key phrase extraction. In NER the objective is to identify phrases that are names of specific types of entities, such as people, organizations or places. This, again, is very useful for document classification and search, but also a steppingstone to extract more complex knowledge, in particular statements relating different entities, as we will see later.

Named Entity Recognition (NER)

Uses of NER

- Named entities can be indexed, linked, etc.
- Sentiment can be attributed to companies or products
- Information extraction can use named entities as anchors

Commercial tools available

- Reuters' OpenCalais, AlchemyAPI (now IBM)
- Python libraries: NLTK NER, Spacy

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

NER has many commercial applications, e.g., for marketing or studying public perception, by linking volume of communication, sentiment and popularity to specific entities, such as products, companies or organizations. Thus, there exist manycommercial tools that offer this type of service.

NER as Sequence Labelling Task

Sequence of tags, indicating whether a word is inside (I) or outside of an entity (O)

The occurrences of entities (can be) typed

EPFL is located in **Lausanne** in **Switzerland** , next to **Lake Geneva**

I	O	O	O	I	O	I	O	O	O	I	I
ORG				GEO		GEO				GEO	GEO

A classification problem!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic task of NER is to detect whether a word belongs to an entity name or not. Furthermore, when an entity name is detected, it can be classified according to the type of the entity, e.g., an organisation (ORG), a location (GEO), a person etc.

When analyzing a text, NER is thus a classification problem, where for each word it needs to be decided whether it is inside or outside of an entity name. More detailed classifications, whether a word is the beginning or end of entity name, can also be performed. Note that in this context also punctuation marks are considered as words, as they may carry important information on the presence of an entity.

Given that NER can be considered as a classification problem, we must decide on two questions. First, which are the input features for the classifier, and second, which is the classification algorithm to be used. As for the input features, typically the neighborhood of a word is considered. In this neighborhood we find other words, which can be used as directly as features and from which several derived features can be produced. The classifier classifies words while reading the words in the sequence they appear in the document. Therefore, one special kind of feature that is used in NER are the labels that have been produced by the classifier for words preceding the word to be classified. Even though in principle any classification algorithm could be applied, e.g., Naïve Bayes, specific sequence-oriented classifiers (HMM, CRF) can have better performance.

Features used in NER

EPFL is located in **Lausanne** in **Switzerland** , next to **Lake Geneva**

Features of “Lausanne”:

Word and neighboring words:	Lausanne, in
Part-of-speech tags (POS):	POS(Lausanne) = NN
Prefixes and Suffixes:	prefix(Lausanne, 3) = Lau
Word shape:	WS(Lausanne) = Xxxxxxxx
Short wordshape:	SWS(Lausanne) = Xx

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we see a list of typical features that are used in named entity recognition. Some of them are quite specific to the task. For example, part-of-speech tags can be helpful as they allow to distinguish noun phrases (NN) which are typical for entities. Pre- and suffixes are another interesting feature. For example, words ending in “land” would often be locations. Learning this fact can help to generalize the classification to new terms that would contain such a suffix. For entities, in particular in English, also the word shape is an important feature, as usually proper names start with capital letters, or acronyms consist of capital letters only.

Exploiting Context

When deciding the entity type exclusively on local context, important information may be missed

- The release of *Harry Potter and the Philosopher's Stone* in 2001 was **Watson's** debut screen performance.
- Although the system is primarily an IBM effort, **Watson's** development involved faculty and graduate students

Idea: consider a model that takes into the account the sequential structure of language and exploits sentence context

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

If only features derived from the local context from the immediate neighborhood of a word to be classified are used, important context information can be missed. This motivated the use of classification models that exploit the larger context of a word in the text, like the complete sentence in which the word occurs.

Generative Probabilistic Model

Sequence of words (known): $W = (w_1, w_2, w_3, \dots, w_n)$

Sequence of labels (unknown): $E = (e_1, e_2, e_3, \dots, e_n)$

Assume the text is produced by a probabilistic process:

$$P(E, W)$$

Find the most probable model

$$\underset{E}{\operatorname{argmax}}\, P(E|W)$$

Bayes Law

$$\underset{E}{\operatorname{argmax}}\, P(E|W) = \underset{E}{\operatorname{argmax}}\, P(E)P(W|E)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We introduce now a classification approach that has been used for NER and exploits the sequential nature of natural language. The approach belongs to the class of generative probabilistic models, like the one we have introduced for information retrieval.

The basic model assumes that there exists a (unknown) probability distribution $P(E, W)$ that correlates sequences of words with the corresponding sequences of entity labels. The classification task is then to identify for a given sequence of words, the most probable sequence of labels. Using Bayes law, we can reformulate this, by decomposing the conditional probability $P(E|W)$ into the product of two probability distributions. $P(E)$ is a model describing of the probability of different labels to occur, and $P(W|E)$ is a model describing of how words correlate with labels.

Approximation

Label transition probabilities (bigram model)

$$P(E) = P(e_1, \dots, e_n) \approx \prod_{i=2,\dots,n} P_E(e_i | e_{i-1})$$

Word emission probabilities

$$P(W|E) \approx \prod_{i=1,\dots,n} P_W(w_i | e_i)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As it is not possible to estimate the complete probability distribution functions $P(E)$ and $P(E|W)$, we approximate them by making independence assumptions. We assume that the probability of a label to occur, depends only on the previous label. This corresponds to a bigram model, generalizing the unigram model we have introduced in probabilistic information retrieval. For a word we assume that its probability of occurrence depends only on the label it received. Thus, the two probability functions decompose into products of simpler functions that we can estimate.

Hidden Markov Model (HMM)

Graphical representation of the approximate probabilistic model

Maximum Likelihood Estimation

$P_E(I \mid O) = 2 / 4$, $P_W(\text{in} \mid O) = 2 / 5$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We can represent approximate model of the probability distribution graphically as a Markov Model, where we indicate which probabilistic variables depend on which others. More precisely, it is a Hidden Markov Model (HMM). The hidden labels E are unknown, and their probabilities need to be estimated from the words that can be observed.

This approach for estimating probabilities of hidden variables with HMMs can be applied to other sequence labelling tasks. For example, it can be used to learn part-of-speech tags and the types of the entities.

Learning the Model

To learn the conditional probabilities from a document collection using Maximum Likelihood Estimation requires only counting

e.g., $P_E(I|O) = 2/4$, $PW\ (in|O) = 2/5$

Smoothing: Unseen words might only accidentally miss in the training data of length n :

$$P_{WS}(w_i|e_i) = \lambda\, P_W(w_i|e_i) + (1-\lambda)\frac{1}{n}$$

For labels no smoothing is needed, as all labels occur in the training data

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given a document collection we can estimate the probabilities $P_E(e_i|e_{i-1})$ and $P_W(w_i|e_i)$. As in probabilistic information retrieval we use maximum likelihood estimation. For example, for estimating the probability $P_E(e_i|O)$ we count the total number of occurrences of O, and then compute the ratio between the cases where the preceding label is e_i with the total number of occurrences of O.

As in probabilistic information retrieval, we need to consider the issue of sparsity of words in the training set. It might be the case that a specific word does not occur together with a given label in the training data, whereas this word still might be related to the label in general. Therefore, smoothing is applied, where the smoothing parameter depends on the size of the training data. The more data is available, the less the likelihood that a word-label pair that is likely to occur is not found in the training data.

For estimating the probability of labels to occur, no smoothing is required, as the number of labels is very small and thus all pairs of labels combinations are likely to occur in the training data.

Using the Model

For a given sequence of words W find the most likely values for the labels E

$$\underset{E}{\operatorname{argmax}}\, P(E|W)$$

Brute force search: compute for all possible sequences $E = (e_1, e_2, e_3, \dots, e_n)$ the probability $P(E|W)$ and then take the maximum

Complexity $O(2^n)$ → unfeasible for longer sequences

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Once the parameters of the HMM model have been derived from the training data, they can be used to estimate the most likely values of labels for an unknown sequence of words. One possibility is to apply brute-force search by computing the probability of each possible label sequence using the model. For longer sequences this becomes computationally intractable as the number of possible sequences grows exponentially.

Observation

$$\underset{E}{\operatorname{argmax}}\, P(E|W)$$

$$= \underset{E}{\operatorname{argmax}} \prod_{i=2,\ldots,n} P_E(e_i|e_{i-1}) \prod_{i=1,\ldots,n} P_W(w_i|e_i)$$

$$= \underset{E}{\operatorname{argmax}}\, P_E(e_n|e_{n-1})\, P_W(w_n|e_n)$$

$$\underbrace{\underset{E}{\operatorname{argmax}} \prod_{i=2,\ldots,n-1} P_E(e_i|e_{i-1}) \prod_{i=1,\ldots,n-1} P_W(w_i|e_i)}_{\text{Independent of the choice of } e_n}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We made an independence assumption on the probabilities of the elements of a label sequence, where a label depends only on its predecessor in the sequence. We can exploit this independence assumption to simplify the computation of the sequence probability. The choice of the the last label in the sequence that maximizes the overall probability is independent of the choices of the other labels in the sequence maximizing the overall probability. Using this property simplifies the computation of the sequence probability significantly. Note tha

Viterbi Algorithm

Let $\pi(k, v)$ be the maximum probability a sequence of length k can achieve with last label v

Then

$$\pi(k, v) = \max_{u} \pi(k-1, u)\ P_E(v|u)\ P_W(w_k|v)$$

$$\pi(0,*) = 1$$

This is a dynamic programming algorithm
→ Viterbi algorithm

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Using the independence assumption described before, we can iteratively compute the sequence of labels that maximizes the probability, using in each step the label sequence found so far. The maximum probability $\pi(k, v)$ that a label sequence of length k can achieve, if the last label is v can be computed from the maximum probabilities known for shorter sequences and the parameters of the probabilistic model.

This algorithm is a simple version of Viterbi's algorithm. In its general form a random variable can depend on several earlier random variables in the earlier sequence, resulting in a dynamic programming algorithm.

An HMM model would not be an appropriate approach to identify

A. Named Entities
B. Part-of-Speech tags
C. Concepts
D. Word n-grams

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Which statement is correct?

A. The Viterbi algorithm works because words are independent in a sentence
B. The Viterbi algorithm works because it is applied to an HMM model that makes an independence assumption on the word dependencies in sentences
C. The Viterbi algorithm works because it makes an independence assumption on the word dependencies in sentences
D. The Viterbi algorithm works because it is applied to an HMM model that captures independence of words in a sentence

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.1.3 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 entity 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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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.

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

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

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

$$Contributors_e = \{(m', \operatorname*{argmax}_c PPR_{(c,\, m')}(e), m' \neq m)\}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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.

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.1.4 Entity Matching with GPT

GPT can be prompted to directly decide whether to entities are the same

- the challenge is to design good prompts

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Considerations in Prompt Design

General vs. domain-specific

- “are these entities the same?” vs. “are these products the same”

Simple vs. complex

- ”match” vs. “refer to the same real-world entity”

Free vs. forced answer

- “answers with ‘yes’ or ‘no” vs. no instruction

Adding entity attributes

- Adding price information or not

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Sample evaluation results

Prompt	P	R	F1	Δ F1	cost (¢) per pair
general-complex-free-T	49.50	**100.00**	66.23	-	0.11
general-simple-free-T	70.00	98.00	81.67	15.44	0.10
general-complex-forced-T	63.29	**100.00**	77.52	11.29	0.14
general-simple-forced-T	75.38	98.00	85.22	18.99	0.13
general-simple-forced-BT	79.66	94.00	**86.24**	20.01	0.13
general-simple-forced-BTP	71.43	70.00	70.70	4.47	0.13
domain-complex-free-T	71.01	98.00	82.35	16.12	0.11
domain-simple-free-T	61.25	98.00	75.38	9.15	0.10
domain-complex-forced-T	71.01	98.00	82.35	16.12	0.14
domain-simple-forced-T	74.24	98.00	84.48	18.25	0.13
domain-simple-forced-BT	76.19	96.00	84.96	18.73	0.13
domain-simple-forced-BTP	54.54	84.00	66.14	-0.09	0.13
Narayan-complex-T	85.42	82.00	83.67	17.44	0.10
Narayan-simple-T	**92.86**	78.00	84.78	18.55	0.10

Findings on what helps
- domain-specific wording
- simpler wording
- forcing an answer

Findings on what hurts
- price information or products (lack of normalization)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In-context Learning

Providing sample training data

Matches:

Product 1: 'Title: SANDISK EXTREME PRO SDHC 32GB 300MB/S UHS-II U3' Product 2: 'Title: Sandisk SDXC card Extreme Pro UHS-II, 32gb, 300mbps'

Product 1: 'Title: Dymo 53718 Black On Yellow - 24mm' Product 2: 'Title: Dymo 24mm Black On Yellow D1 Tape (53718)'

Product 1: 'Title: DS-7216HQHI-K1 Hikvision 16 cs. TurboHD DVR' Product 2: 'Title: Hikvision DS-7216HQHI-K1 Turbo HD DVR'

Product 1: 'Title: APCRBC133APC Replacement Battery Cartridge #133' Product 2: 'Title: APC RBC133 Replacement Battery Cartridge'

Product 1: 'Title: Gigabyte NVIDIA GeForce GTX 1650 4GB D6 OC Turing Graphics Card' Product 2: 'Title: GigaByte GeForce GTX 1650 D6 Windforce OC 4G'

Non-matches:

Product 1: 'Title: RAM Corsair ValueSelect DDR4 2133MHz 1x8Go' Product 2: 'Title: CORSAIR New 8gb (1x8gb) Ddr4 2666mhz Vengeance CMK8GX4M1A2666C16'

Product 1: 'Title: Evolis Zenius/Primacy Black Monochrome Ribbon 2000 image RCT023NAA' Product 2: 'Title: Zebra 800015-101 Black Monochrome Ribbon - 1000 Prints'

Product 1: 'Title: 128GB Pendrive SanDisk Extreme PRO SSD USB 3.1 420MB/s' Product 2: 'Title: SanDisk 128GB Extreme Pro SDXC 150MB/s V-30 UHS-1 U3 Memory Card'

Product 1: 'Title: Samsung SSD 970 EVO Plus 250GB' Product 2: 'Title: Samsung 970 EVO SSD M.2 2280 - 500GB'

Product 1: 'Title: Akumulator APC Replacement Battery Cartridge #110' Product 2: 'Title: APC - Replacement Battery Cartridge #117'

Do the following two product descriptions refer to the same real-world product? Answer with 'Yes' if they do and 'No' if they do not.

Product 1: 'Title: MEMORIA SD SANDISK ULTRA SDHC SDXC UHS I CARD 100 MBS SDSDUNR 256G GN6IN'

Product 2: 'Title: SanDisk Extreme Pro (256GB) SDXC Memory Card'

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Evaluation Results

Selection heuristic	Shots	P	R	F1	Δ F1	Cost (¢) per pair	Cost increase	Cost increase per Δ F1
ChatGPT-zeroshot	0	71.01	**98.00**	82.35	-	0.14	-	-
ChatGPT-random	6	78.33	94.00	85.45	3.10	0.77	450%	145%
	10	79.66	94.00	86.24	3.89	1.13	707%	182%
	20	78.95	90.00	84.11	1.76	2.07	1379%	783%
ChatGPT-handpicked	6	76.19	96.00	84.86	2.51	0.72	414%	165%
	10	80.00	96.00	87.27	4.92	1.00	614%	125%
	20	79.66	94.00	86.24	3.89	2.03	1350%	347%
ChatGPT-related	6	80.36	90.00	84.91	2.56	0.68	386%	151%
	10	**89.58**	86.00	87.76	5.41	1.05	650%	120%
	20	88.46	92.00	**90.20**	7.85	1.97	1307%	167%
GPT3.5-handpicked	10	61.97	88.00	72.72	-9.63	10.54	7429%	771%
	20	61.43	86.00	71.67	-10.68	19.71	13979%	1309%
GPT3.5-related	10	67.69	88.00	76.52	-5.83	10.04	7071%	1213%
	20	61.43	86.00	71.67	-10.68	20.34	14429%	1351%

Findings
- in-context learning helps
- (lexcially) related terms work better than human selected
- cost increases significantly

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Lexically related = high Jaccard similarity

Providing Matching Rules

Task Desc.	Your task is to decide if two product descriptions match. The following rules regarding product features need to be observed:
Rules	1. The brand of matching products must be the same if available 2. Model names of matching products must be the same if available 3. Model numbers of matching products must be the same if available 4. Additional features of matching products must be the same if available
Task Desc.	Do the following two product descriptions match? Answer with 'Yes' if they do and 'No' if they do not.
Task Input	Product 1: 'Title: DYMO D1 – Roll (1.9cm x 7m)' Product 2: 'Title: DYMO D1 Tape 12mm x 7m'

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Evaluation Results

Prompt	Shots	P	R	F1	Δ F1	Cost (¢) per pair	Cost increase	Cost increase per Δ F1
ChatGPT-zeroshot	0	71.01	**98.00**	82.35	-	0.14	-	-
ChatGPT-zeroshot with rules	0	80.33	**98.00**	88.29	5.94	0.28	100%	17%
ChatGPT-related	6	80.36	90.00	84.91	2.56	0.68	386%	151%
	10	89.58	86.00	87.76	5.41	1.05	650%	120%
	20	88.46	92.00	**90.20**	7.85	1.97	1307%	167%
ChatGPT-related with rules	6	90.70	78.00	83.87	1.52	0.79	464%	305%
	10	90.91	80.00	85.11	2.76	1.17	736%	267%
	20	**91.11**	82.00	86.32	3.97	2.09	1393%	351%

Findings
- rules with in-context learning increases precision at cost of recall
Note that rules do not require training data

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

References

Lecture partially based on

- Dan Jurafsky and James H. Martin, Speech and Language Processing (3rd ed. Draft), Chapter 21 https://web.stanford.edu/~jurafsky/slp3/
- Jay Pujara and Sameer Singh, Mining Knowledge Graphs from Text, Tutorial, https://kgtutorial.github.io

References

- Mintz, Mike, et al. "Distant supervision for relation extraction without labeled data." *ACL 2009*.
- Riedel, S., Yao, L., McCallum, A., & Marlin, B. M. Relation extraction with matrix factorization and universal schemas. *ACL 2013*.
- Ralph Peeters and Christian Bizer, “Using ChatGPT for Entity Matching”, ADBIS 2023

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.2 KNOWLEDGE REPRESENTATION

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.2.1 Representation of Semantics

Information retrieval and data mining aim at extracting and using semantics that is implicitly represented in documents and data

Alternatively, semantics can, and is, also explicitly represented and manipulated ☞ knowledge representation

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Historically speaking, the first information systems were based on explicit representation of the semantic models used. In business information systems, the data relevant to businesses was and is till today explicitely modelled in a structured data model, e.g. the relational data modelled. The data of the model is manipulated either manually, e.g. through manual entry, or processed by algorithms, e.g. for accounting. Only at a later stage with the advent of more computing power the use of implicitly represented models, as found in text documents, became feasible. The methods we have covered so far in this lecture fall in this later class. Interestingly, as algorithms become increasingly powerful in extracting information from unstructured content, the explicit representation of the extracted models and their use for further inference is now gaining in interest and importance.

Database Schemas

Schemas define data structures for databases

- Relational database schemas, XML Schemas

Agreement on data structures

- Well understood meaning of data values
- Data consistency, e.g., integrity constraints

Optimizing query evaluation and data storage

- e.g., indexing, query optimization

STUDENTID	NAME	COURSE
1234	John	DIS
3456	Bob	physics
2345	Ann	DIS

Schema

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The probably most widely used approach for explicitly representing semantic models is through database schemas in database management systems. Schemas in database management systems play two important roles. First, they define data structures that capture that application semantics. They label data values with names (e.g., attribute names) that have well-understood semantics, or of which the semantics is provided by additional documentation. Second, by providing well-defined data structures and additionally imposing integrity constraints, database schemas additionally assure that data stored in the database remains structurally and semantically coherent. Exploiting knowledge on data structures also enables query and storage optimizations.

However, in many contexts of information management, e.g., on the Web, schemas are not always readily available, and imposing the use of schemas might be too rigid.

Data on the Web

Example: Searching biological data

- Based on textual content of Web pages (e.g. Google)

Searching for data on "anglerfish"

- Results will be precise

This seems easy, but the same for "leech"

- Organism leech
- Software: LeechFTP
- Rock band: Leech
- Authors: Leech
- Protein sequences: …MNTS**LEECH**MPKGD…

Search for "257" …

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When publishing information on the web, no schemas are used, even if the published data could follow a regular data structure. Consequently, when searching the Web, search is content-based, e.g., using keyword queries.

This may work well for very characteristic search terms, such as illustrated for the example “anglerfish”, but might become problematic, when the search term has an ambiguous meaning, as illustrated for the example “leech”. Since a text search the type of the information that is searched cannot be specified – in our example we would need to specify that we meant the “leech” is an organism - the search algorithm cannot disambiguate the different potential meanings of the search terms. In the extreme case, if we would try to search for a numerical value as a search term, the results become essentially useless.

This suggest that the use of schema information for Web search could have some merit.

Using HTML to Structure Data

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A typical approach to handle data in the Web is to publish documents with a regular structured layout, e.g., using HTML tables. In this way a context is provided that allows to correctly interpret data values. When inspecting the source code of a typical HTML page publishing structured data, we recognize that it might be difficult to automatically process such a page and to correctly interpret the data contained in it. Such processing might be very involved and error-prone, even more so as in the context of editing Web page layout no mechanism assures that the intended page structure is correctly maintained. The problem is that a document layout language, such as HTML, never has been conceived to specify semantics of data.

Web Scraping

Encoding data semantics through layout is a bad idea

- Generations of "Web scrapers" experience this

FOR E-COMMERCE DATA SCIENTISTS: LESSONS LEARNED SCRAPING 100 BILLION PRODUCTS PAGES

July 02, 2018 Ian Kerins 11 Comments

Challenge #1 - Sloppy and Always Changing Website Formats

It might be obvious and it might not be the sexiest challenge, but sloppy and always changing website formats is by far the biggest challenge you will face when extracting data at scale. Not necessarily because of the complexity of the task, but the time and resources you will spend dealing with it.

https://www.zyte.com/blog/price-intelligence-web-scraping-at-scale-100-billion-products/

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The widely adopted approach of publishing data through HTML documents has given rise to a whole industry of web scraping. It is of interest to gather data from the Web, for example, for comparison of prices from ecommerce sites. However, experience shows that this is a complex and time-consuming task. It is also a never-ending task as formats of Web sites can change at any time and there exist no standards on how the data is represented in HTML.

Application-specific Markup

Limitations of HTML

- Structure of data expressed as layout :
  `<tr><td> leech </td><tr>`
- Semantics of data hard to analyse and difficult to share
- No schemas, no constraints

Application-specific markup (tags)

Making the meaning of data explicit

- SwissProt: `<Species> leech </Species>`
- EMBLChange: `<Organism> leech </Organism>`

Embedding of schema information into the data

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Therefore, an alternative approach for specifying the meaning of data contained in Web documents, has been conceived. As for HTML, the approach is based on the idea of using tags, i.e., markup text that associates parts of a document with a name. However, instead of using tags to provide layout instructions (e.g., a table format), tags are used to provide domain-specific meaning to data contained in a document. For example, markup can be used to specify that a specific name (such as leech) denotes an organism. This approach of using markup is widely used in different contexts. For example, email formats contain different standard markups to indicate the meaning of different parts of a mail. In the Web, the use of markup has found wide-spread adoption through the XML format. XML generalizes the markup approach from HTML to allow applications to specify their application-specific tags and to embed them into documents. If you are not familiar with XML, it would be helpful to refresh your knowledge on XML and schemas in XML.

4.2.2 Semi-structured Data

Data that embeds schema information (through tags, markup etc.) to explain the meaning of data values and to relate different data values (e.g. hierarchically) = semi-structured data

No predefined data structure, no schema required!

Examples of semi-structured data

- Email
- Microformats (e.g. hCard)
- JSON
- XML (Extensible Markup Language)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Data that contains embedded schema information, such as tags or markup, is generally called semi-structured data. Semi-structured data allows to indicate the meaning of data values, as in structured data with schemas, without necessarily requiring predefined schemas. This combines flexibility of modelling with the ability to specify meaning. The fact that semi-structured data does not require schemas, does not imply that we cannot define schemas for semi-structured data. XML is an example of this. We can have XML structured documents without schema, so-called well-formed XML documents, but define in addition schemas for XML (document type definitions, so-called XML DTDs). Documents that follow such a schema are then called valid XML documents.

XML – a Document Markup Language

```
<?xml version="1.0"?>
<purchaseOrder orderDate="1999-10-20">
  <shipTo country="US">
    <name>Alice Smith</name>
    <street>123 Maple Street</street>
    <city>Mill Valley</city>
    <state>CA</state>
    <zip>90952</zip>
  </shipTo>
  <billTo country="US">
    <name>Robert Smith</name>
    <street>8 Oak Avenue</street>
    <city>Old Town</city>
    <state>PA</state>
    <zip>95819</zip>
  </billTo>
  <comment>Hurry, my lawn is going wild!
    </comment>
  <items>
    <item partNum="872-AA">
      <productName>Lawnmower
      </productName>
      <quantity>1</quantity>
      <price>148.95</price>
      <comment>Confirm this is electric
      </comment>
    </item>
    <item partNum="926-AA">
      <productName>BabyMonitor
      </productName>
      <quantity>1</quantity>
      <price>39.98</price>
      <shipDate>1999-05-21</shipDate>
    </item>
  </items>
</purchaseOrder>
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Originally XML has been conceived as a document model. In this example of an XML document, we can clearly see the document nature of XML. An XML document consists of hierarchically nested tags which enclose textual content.

XML- a Semi-structured Data Model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

However, the same XML document can also be viewed as structured data. An XML parser would typically generate a hierarchical data structure as illustrated. This explains why we can understand XML also as a data model, more precisely a semi-structured data model, as it embeds schema information directly into the data.

Schema-less data

Benefits of schema-less data

Increased flexibility

- e.g., dynamically adding or dropping structural elements such as attributes

Self-contained data

- e.g., context (schema information) directly encoded into data (markup)

Drawbacks

Loss of consistency

Certain optimizations not feasible

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Semi-structured data can be schema-less. This has significant advantages for applications that do not know the structural elements in advance. For example, often it is helpful if new types of attributes can be added in a database as application requirements evolve. Another advantage is that by embedding schema information into the data, data can be interpreted without requiring additional context information in the form of database schemas. This can be useful in applications where data is exchanged.

Having no schema has however also its drawbacks as the additional flexibility also can be at the detriment of data consistency. Applications might, for example, not be able to properly process attributes that have been arbitrarily added. Also many optimizations that rely on the availability of a schema, and on the assumption that the schema is stable, cannot be applied.

Serialization: Data and Documents

Document = medium for exchange of information

Communication

Information system 1

Information system 2

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The duality of XML, being a document model and a data model at the same time makes it particularly suitable for applications that exchange data over communication networks. For every data collection or database that is represented in XML there exists a canonical encoding into a text (document). The encoding is called serialization. Since documents are sequences of symbols, they can be naturally exchanged over communication networks. There exist many other similar models for serializing data into documents. Recently JSON has become widely used for exchanging data among applications and has to a certain extent substituted XML in this role in many contexts. However, JSON is less suited for applications that enrich natural language text with structured data.

Semi-structured data …

A. is always schema-less
B. always embeds schema information into the data
C. must always be hierarchically structured
D. can never be indexed

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Why is XML a document model and not just a general data model?

A. It supports application-specific markup
B. It supports domain-specific schemas
C. It has a serialized representation
D. It uses HTML tags

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Serialised vs. Semi-structured Data

Many of the semi-structured data models have been conceived to structure text documents (XML) or serialise data (JSON)

However, there exist semi-structured data models that are not based on a serialized representation

→ knowledge graphs

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Serialisibility and semi-structured data are concept that are strongly correlated but not the same. Many of the semi-structured data models, like XML and JSON, have been introduced to either enrich documents with data semantics, or to encode data in a serialized format. However, it is perfectly possible to define a semi data model, that does not necessarily aim at a serialized representation. Knowledge graphs are an example for this - though they of course also can be and are serialized. Knowledge graphs have been conceived to provide a flexible way to model data and capture semantics. We will next review the development of this concept.

4.2.3 The Semantic Web

User-defined markup (schemas): provides possibility to share interpretation of data across various applications

Different databases – Different schemas

- SwissProt: Find `<Species> leech </Species>`
- EMBLChange: Find `<Organism> leech </Organism>`

Problem: Semantic Heterogeneity → Semantic Web

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

XML provides a framework to encode data on the Web in a standardized fashion, to deal with irregular Web data structures and to exchange Web data among information systems. Thus, it is an approach to address syntactic heterogeneity on the Web.

The markup used in XML can be interpreted by applications. This enables automated processing of Web data. However, for properly interpreting the markup, semantic heterogeneity needs to be addressed as well. The same concepts can ^represented in different application contexts differently, e.g., by using different terms to denote the same meaning. This is a major obstacle to enable interoperability on the Web and indeed in industry this is considered as one of the major problems in the use of information technology. Already in 2010, Mike Brodie at the time at GTE, estimated that the total cost of semantic integration world-wide amounts to 1 trillion$ / year. Very likely, this figure has in the meantime increased.

The Vision of W3C: Semantic Web

The *Semantic Web* is an extension of the current Web in which *information is given well-defined meaning*, better enabling computers and people to work in cooperation. The mix of content on the Web has been shifting from exclusively human-oriented content to more and more data content. The Semantic Web brings to the Web the idea of having data defined and linked in a way that it can be used for more effective *discovery, automation, integration, and reuse across various applications*. For the Web to reach its full potential, it must evolve into a Semantic Web, providing a universally accessible platform that allows data to be shared and processed by automated tools as well as by people. The Semantic Web is an initiative of the World Wide Web Consortium (W3C), with the goal of extending the current Web to facilitate Web automation, universally accessible content, and the 'Web of Trust'.

(http://www.w3.org/2001/sw/Activity)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to address the problem of semantic interoperability and thus enable automated processing of Web data, the W3C initiated the “Semantic Web” initiative. The Semantic Web is a framework that builds on Web technology, including the XML framework and extends it with technologies that facilitate semantic interoperability.

Three Ways to Overcome Semantic Heterogeneity

1. Standardization: agree on common user-defined markup (schemas)
   - great if no pre-existing applications
   - great if power player enforces it
2. Translation: create mappings among different schemas and databases
   - requires human interpretation and reasoning
   - mappings can be difficult, expensive to establish
3. Annotation: create relationships to agreed upon conceptualizations
   - requires human interpretation and reasoning
   - annotation can be difficult, expensive to establish
   - reasoning over the conceptualization can provide added value

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

There exist three basic approaches to tackle the problem of automating semantic processing of the data on the Web:

1. Standardization avoids the problem of semantic heterogeneity a priori. This approach is used when there exists already (historically) a wide agreement on the structure of relevant data and their interpretation. For example, terminology in the financial industry is largely standardized, and therefore it is not a major problem to produce agreed upon formal specifications of such terminology and corresponding schemas to represent the data. This is even more the case as there exist in this type of business environment typically strong players that can enforce the standards.
2. Translation or mapping between different schemas and databases is the second possibility to establish semantic interoperability. This is the approach that has been extensively adopted for data integration problems in relatively small and controlled domains, such as inside businesses and organizations. The requirements in these domains are typically changing not too quickly, thus much effort can be invested into developing the necessary mappings in order to properly map data from one representation into another, or to map data from multiple representations into one common global representation.
3. The third possibility is slightly different from the second: instead of engineering mappings between heterogeneous schemas for each integration problem, one first agrees on a common conceptualization of the domain, covering relevant aspects for a large class of applications. This conceptualization is normally called an ontology and is assumed to be application independent. Since ontologies have a formal representation, they are machine-processable. Once such an ontology is in place, existing information system can relate their structural elements for expressing certain concepts (e.g., element names) to concepts from the ontology. This then ideally enables other applications to properly interpret the contents of the information system. In addition, ontologies in the general case should include reasoning capabilities, which would permit to use not only hard-coded, or pre-canned knowledge, e.g., in form of explicitly relating concepts from an information system to the ontology, but also to derive new knowledge by combining existing knowledge.

This differentiation among different approaches to semantic interoperability has been standardized: according to ISO 14258 (Concepts and rules for enterprise models), there are three basic ways to relate entities together:

Integrated approach. There exists a common format for all models. This format must be as detailed as the models themselves. The common format is not necessarily a standard but must be agreed upon by all parties in order to elaborate models and build systems. (-> Standardization)

Federated approach. There is no common format. In order to establish interoperability parties must accommodate on the fly. Federated approach implies that no partner imposes their models, languages and methods of work. This means they must share a common ontology. (-> Translation)

Unified approach. There exists a common format but only at a meta-level. This meta-model is not an executable entity as in the case of the integrated approach but provides a way for semantic equivalence to allow mapping between models. (-> Annotation)

Mapping: Three Approaches

1. Standardization
   - Mapping through standards

2. Mapping
   - Direct mapping

3. Ontologies
   - Mediated mapping

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Conceptually there exist three principal approaches of how to address semantic heterogeneity. A first approach is to map all the models to one common global model. This approach is taken with standardization. For example, EDIFACT (https://www.unece.org/cefact/edifact/) is an international standard that models all concepts that are commonly used in business and trade. For exchanging information, the information systems map their data to EDIFACT and can thus share their data with other information systems.

A second approach consists of constructing directly a mapping among two information systems, without using any additional, shared knowledge in form of standards or ontologies

A third approach is to relate the model of an information system to a common model, frequently called ontology, and use this mapping to construct a direct mapping among the different models used in the information systems. . For example, in the financial domain there exists a standard reference ontology called FIBO (https://fib-dm.com/finance-ontology-transform-data-model/)

Direct Schema Mapping

Assume all data represented in canonical data model (e.g. relational)

- detect correspondences (schema matching)
- resolve conflicts
- integrate schemas (schema mapping)

Mappings are frequently expressed as queries (e.g. SQL Query)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Creating direct mappings among information systems has been widely applied for mapping data that is stored in relational, object-oriented and XML databases, where the model is represented as a database schema. The problem is known as the **schema mapping** problem. Given two schemas of databases, first schema elements are identified that are likely to correspond to each other, i.e., that are likely representing the same real-world aspect. This can be done by using many types of analysis using structural and content features of the database schema and the database. Tools for supporting these steps are called schema mapping tools. Once this step is performed some conflicting correspondences may occur, e.g., mapping some concept in one schema to two different ones in the other. These need to be resolved typically through decisions taken by humans. Once the correspondences are consistent, mappings can be automatically derived. The mappings are typically expressed as queries of the data manipulation language.

Schema Matching Tools

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This screenshot shows a schema matching tool that maps between different XML schemas. The lines indicate which attributes might be related to each other. One can easily see a lot of conflicts where the same attribute corresponds to multiple attributes in the other schema.

Matching with LLMs

It is possible to generate schema transformation code with GPT

	name	description	incident_type	receiver_name	receiver_country	receiver_region	receiver_category	receiver_category_subcode	initiator_nam
0	Cyberwar Azerbaijan-Armenia 2000/Azerbaijani A...	Azerbaijani Hackers took down and defaced webp...	Disruption	[['Not available']]	['Armenia']	[['ASIA', 'CENTAS', 'CSTO']]	[['State institutions / political system', 'Me...	NaN	['N available
1	Cyberwar Azerbaijan-Armenia 2000/Armenian Coun...	In response to the previous attack Armenian ha...	Disruption	NaN	['Azerbaijan']	[['ASIA', 'CENTAS']]	[['Social groups',		

Code properly working on first run!

	victim	damage	attack	attacker	affected	time	fi
0	Cyberwar Azerbaijan-Armenia 2000/Azerbaijani A...	operational disruption	Disruption	Not available	None	Unknown	
1	Cyberwar Azerbaijan-Armenia 2000/Armenian Coun...	operational disruption	Disruption	Liazor	None	Unknown	
2	Taiwan Election Hack 2000	operational disruption	Disruption	Unknown	None	Unknown	None

None	State institutions / political system	Taiwan	Chinese hackers succeeded in attacking several...

I have two datasets describing cyber-attacks.

The first is structured as follows (attribute names and 3 samples)

... give here the head of the table ...

The second has the following format

... give here the head of the table ...

Provide a python code that transforms the first format into the second format.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This was the code generated for this case

```
def transform_to_json_format(record):
return { "victim": record.get("name", "Unknown"),
"damage": "operational disruption", # Assuming
operational disruption for all "attack":
record.get("incident_type", "Unknown"), "attacker":
record["initiator_name"].strip("[]'") if
isinstance(record.get("initiator_name"), str) else
"Unknown", "affected": "None", # Default value
"time": record.get("start_date", "Unknown")[:7],
"financial loss": "None", # Default value "sector":
record.get("receiver_category",
"Unknown").split(',')[0].strip("[]'"), "country":
record["receiver_country"].strip("[]'") if
isinstance(record.get("receiver_country"), str)
else "Unknown", "summary":
record.get("description", "No description
available"), "doc_id": "None" # Default value }
```

This simple example illustrates of how ontologies can help to increase semantic interoperability and how new knowledge can be generated by inference. Take the earlier example of biological databases. These can use different schemas to represent the same or related facts. For example, System 1 uses the term Organism to denote an organism, and System 2 uses the term Species to do the same. Two annotators, who share the same ontology, now inspect the document and each of them associates the elements with terms taken from the ontology. So, Annotator 1 decides that the element Organism corresponds to information related to an organism, whereas annotator 2 concludes by analyzing the content that the element is related to information about a fish and annotates correspondingly. The annotation can be perfoermed by adding an is-instance-of relationship. At this point, the facts that both annotators used the same ontology, and that reasoning is possible in this ontology come into play. Since in the ontology a Fish is a sub-concept of Organism (a fact represented formally by an ISA relationship in the ontology) an automated processing tool (e.g., for searching for information) can exploit this relationship and correctly identify for a request for information on Organisms in both databases the related elements.

Ontologies

Ontologies are an explicit specification of a conceptualization of the real world (Gruber, 1993)

Ideally

- different information systems agree on the same ontology
- relate their model/schema/data elements to the ontology
- mapping can be constructed via the ontology

Requires agreement on the conceptualization of the real world!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Thus, ontologies provide a “proxy representation” for the real world, to which the interpretation of data in information systems refers to and thus provide an interpretation of the data in an information system that can be automatically processed. In order to realize this approach, different technical challenges need to be addressed.

Creating Ontologies

Different approaches to create ontologies

1. Ontology engineering
   - Manual effort
   - Tools for editing and checking consistency
2. Automatic generation of ontologies
   - From large document collections or existing structured data sources

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to make ontologies available, they need to be constructure. This is the practice of ontology engineering. Ontology engineering is largely a manual process, that can be supported by tools for checking consistency and facilitating editing. Ontology are often not only constructed once but need to be maintained to stay up to date with developments in a given domain. For example, important entities, like organizations, can change, technology evolves etc.

More recently, also automated methods based on information extraction from large document collections are becoming available to that end. We will see later examples of such methods using statistical and machine-learning approaches.

Modeling and Encoding of Ontologies

Issues

- Modeling primitives and their semantics
  - what does an arrow mean?
  - what does "instance-of" mean?
  - what does ISA mean?
- Standardized encodings of the model
  - Into document language (XML, HTML)
  - Enriching document content with semantic markup

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For representing ontologies, a mathematical model for the constructs used in modelling the application domain is required, as well as a data model to represent such a model in a data management system. The choice of the formal mathematical is important, as the model should at the same time be expressive to capture a wide range of real-world aspects, but at the same time understandable by the user, have a well-defined semantics and relevant reasoning capabilities.

Finally, the models should also provide a serialized representations, e.g., by encoding them in standard document languages such as XML or HTML. Such encodings can at the same time be used to enrich text documents with semantic annotations.

Model Requirements for Ontologies

	HTML	XML	RDF	OWL
Simplicity	+	+	+	+
Exchangeability	+	+	+	+
Non-intrusive annotation	+	-	+	+
Domain-specific vocabularies	-	+	+	+
Modeling primitives	-	-	+	+
Reasoning capabilities	-	-	-	+

"separate structure from presentation"

"separate interpretation from structure"

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We have identified so far different models for representing information in the Web, HTML, XML, and RDF. They all serve specific purposes and exhibit characteristic properties. This table summarizes those properties and highlights interesting “separation of concerns”. RDF and OWL, an extension of RDF to provide more reasoning capabilities separate the concern of reasoning at a semantic level from structuring of data. XML has been serving to separate the concern of structure of data, from providing layout information to documents. The table highlights related design objectives.

1. Simplicity: the success of the Web was always founded on the principle of simplicity of concepts to encourage wide-spread use. Early version of ontology models, such as CyC, where partly also not widely adopted due to their complexity. Similarly, XML was developed as a simplification of a predecessor standard, called SGML, which was more complex.
2. Exchangeability: Since the Web is a communication environment, any kind of data that is processed must be easily exchangeable. This is what motivated the use of XML as a data representation format in the first place and should apply for ontology models as well.
3. Non-intrusive annotation: machine-processable knowledge required for the interpretation of data will be associated with the data typically a-posteriori. Also there does not always exist a unique interpretation for the same data. Therefore, any attempt to encode the knowledge required for interpretation directly into the data is not practical. This excludes, for example, the approach to use XML elements for annotation.
4. Domain-specific vocabularies: an ontology model must provide a mechanism that permits to introduce vocabulary or terminology that is specific to a domain, in other words the possibility to specify schemas for different domains.
5. Modeling primitives: since an ontology model will be used in many different contexts, they have to offer a sufficiently rich set of possibilities to model complex structures and relationships. There exists a rich experience in modeling (e.g., from data modeling in databases with the entity-relationship models) that can be applied to ontology models.
6. Reasoning Capabilities: even simple forms of reasoning within the ontology layer can make the interpretation of the data much more powerful and thus automated processing of data in the Semantic Web.

W3C Web architecture

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This figure depicts the architecture adopted by the W3C. It standardizes different layers of information processing. The lowest layer is concerned with identification and encoding as basic building blocks for building data models. XML together with the namespace mechanism is the data modeling framework. RDF and its extension OWL enable semantic interpretation and exchange. The levels of proof and trust are still rather hypothetical.

An ontology …

A. helps to separate layout issues from the structural representation of data

B. defines a common syntactic framework to represent standardized domain models

C. can be used as a mediation framework for integrating semantically heterogeneous databases

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.2.4 RDF - Resource Description Framework

How to produce a serialized representation of a graph?

How to describe complex entities?

How to incorporate schema information into a knowledge graph?

How to reason about the knowledge graph?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

RDF has been designed by the W3C with a number of different questions in mind. We will now provide a step-wise introduction into the modeling framework.

RDF Model

RDF (Instances)

- *Statements* about *Resources* (addressable by an URI) and literals (XML data)
- Statements are of the form: subject property object
- Like simple natural language sentences
- RDF statements are themselves resources (reasoning about RDF)
- *Properties* define relationships to other resources or atomic values

RDF-Schema

- Data model to specify schemas for RDF instances
- Which properties are applicable for which objects with which subjects
- Defines "grammar" and "vocabulary" for semantic domains (ontology language)

Relation RDF vs. RDFS comparable to well-formed vs. valid XML

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

RDF is a standard supported by the W3C (http://www.w3.org/RDF/) to represent metadata. It is a graph-oriented data model used to annotate XML documents. RDF consists of two parts: a language for representing metadata instances (RDF), which allows to annotate Web resources with statements. The Web resources are addressed by Universal Resource Identifiers (URI), of which URL's are the most important example. Thus, any Web document or part of it can be annotated. The second part of the RDF standard is a language for specifying schemas for RDF Instances. This language enables specification of the vocabulary and grammar that is used for forming statements for annotation. Since RDF statements can be created also without using RDF schemas, RDF is a semi-structured data model, similar to XML.

The RDF model is like the entity-relationship (ER) model. Entities correspond to resources and relationships correspond to properties. The main difference is that RDF requires that relationships are directed : the resource from which the (directed) relationship emerges is assigned a property with a value to which the relationship points. This reflects the intention to use RDF to associated metadata (the value) with data (the source of the relationship). Sample RDF applications include PICS (annotating documents with information on the suitability of the content for certain groups, e.g., like the movie rating system) and Dublin Core (annotating documents with basic bibliographic information.

It is important to understand that the syntax of RDF and its encoding into XML is well-specified, but that the semantics RDF is only specified in a « semi-formal » manner. This introduces certain ambiguities for applications interpreting RDF statements.

RDF Statements Example

Form 1:
Statement

http://www.doc.ch/
created by
John Smith

Subject	a resource
Predicate	a property
Object	a resource or value

Form 2: Directed Graph

```
<rdf:RDF>
  <rdf:Description about="http://www.doc.ch/">
    <dc:Creator> John Smith</dc:Creator>
  </rdf:Description>
</rdf:RDF>
```

Form 3: XML Document

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic constituent of RDF is an RDF statement. We can view RDF statements in three different ways: (1) like natural language sentences, where the subject is a URI (uniform resource identifier) and the object can be either a URI or a String; (2) as directed graphs, where the subject and object are represented as nodes (an ellipsis is used to represent resources identified by a URI and rectangles to represent literals as atomic XML values) and the predicate is represented as directed link , or (3) as XML documents where the RDF statement is encoded into XML format. The graph representation is suitable for visual presentation and reasoning, whereas the XML representation is used for exchange and storage.

RDF Syntax

Many syntactic varieties possible

Basic form

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For encoding RDF statements into XML there exist several syntactic variations, which can make the reading of RDF documents sometimes rather difficult. Here we summarize the different variants. The basic pattern of encoding is as follows: the subject is represented by an XML element called rdf:Description. This element is the root of the document fragment representing the RDF statement. In the content of the element, one finds one or more predicates, represented by XML elements, e.g., s:Creator. The content of this XML element in turn represents the object of the RDF statement. If the object is not a literal, one can alternatively represent the object as an attribute of the predicate element. Also, both the predicate and the object can be encoded into the element representing the statement, as it is shown for s:Date predicate.

Typing Resources

Resources can be associated with a type by using **rdf:type** (a special property)

```
<rdf:RDF>
  <rdf:Description about="http://www.doc.ch/">
    <rdf:type>Document</rdf:type>
    <dc:Creator> John Smith </dc:Creator>
  </rdf:Description>
</rdf:RDF>
```

abbreviated syntax

```
<rdf:RDF>
  <dc:Document about="http://www.doc.ch/">
      <dc:Creator> John Smith </dc:Creator>
  </dc:Document>
</rdf:RDF>
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

RDF allows us to categorize resources into different classes (typing). For that purpose, one associates with the resource that should be categorized another resource, that represents the category, using a special RDF property rdf:type. We will see later, when introducing RDF schema, what are possible uses of the type statement. With respect to encoding, the type property can either be represented explicitly like any other property, or one can use a special abbreviated syntax, where the name of the type becomes the element name of the element representing the statement.

RDF Complex Values

Use an intermediate resource

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Associating a subject with a property whose value is a simple object is the simplest form of a statement. In general, the value of a property (the object of the statement) may be a complex statement itself, representing a complex data structure. In order to represent such complex object values a new intermediate resource is created - in the example Person://1234/1 - to which different properties are associated. Note that this is different of directly associating these properties with the subject of the overall statement, http://www.doc.ch/ in the example (why?).

In the XML encoding such a complex object value can be represented by directly inlining the complex object into the content of the statement.

RDF Containers

Containers

- Bag (unordered)
- Seq (ordered)
- Alt (alternatives)

Quantifiers

- about: John is author of the talk (consisting of many slides)
- aboutEach: John is author of each slide of the talk

```
<rdf:RDF>
  <rdf:Description about="/talks/7.002">
    <s:slides>
      <rdf:Seq>
        <rdf:_1 resource="/slides/42"/>
        <rdf:_2 resource="/slides/73"/>
      </rdf:Seq>
    </s:slides>
  </rdf:Description>
</rdf:RDF>
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Statements can be made not only using single resources, but as well using collections of resources. For that purpose, RDF provides the container concept. Containers are special resources of one of three container types that are specified in the RDF standard. A container resource is then associated with a set of other resources. By creating a statement using the container object as "object", one can express statements made about the set of objects. More precisely, one can specify whether the statement is a statement of the set of objects "as a whole" or a statement that applies to each element of the set individually. What the implications of this distinction are is not further specified in the RDF standard, which is an example where the semantics of RDF is not clearly specified.

There exist three different types of containers: bags which are unordered multi-sets (= sets with multiple occurrences of the same resources), sequences which are ordered sets (i.e., lists) of resources and alternatives which is a single resource that is to be chosen out of a given set. The property labels can be used to impose an order on the elements of the set, by using labels _1, _2 etc. If the order is irrelevant one can use rdf:li as element name, instead of rdf:_1, rdf:_2 etc.

Creating New Resources

New RDF resources are created by using **rdf:ID** (a special property)

```
<rdf:RDF>
  <rdf:Description rdf:ID="12345">
    <rdf:type>Document</rdf:type>
    <dc:Title>Harry Potter</dc:Title>
  </rdf:Description>
</rdf:RDF>
```

```
<rdf:RDF>
  <rdf:Description rdf:about="#12345">
    <rdf:type>Document</rdf:type>
    <dc:Title>Harry Potter</dc:Title>
  </rdf:Description>
</rdf:RDF>
```

Alternative syntax

```
<rdf:RDF>
  <dc:Document rdf:ID="12345">
     <dc:Title>Harry Potter</dc:Title>
  </dc:Document>
</rdf:RDF>
```

Abbreviated syntax

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

RDF resources are not necessarily Web resources identified by URIs but can also be instantiated within RDF statements. In other words, *new resources* can be defined as part of RDF statements. A resource is in general anything that can be *identified* on the Web. As a consequence, also a newly created resource requires an identifier. For this purpose, RDF provides the rdf:ID attribute to introduce the identifier of the resource. This attribute replaces rdf:about attribute when the statement is about a new resource instead about an existing resource. Using the identifier, this new resource can then be referred to in other RDF statements.

A basic statement in RDF would be expressed in the relational data model by a table …

A. with one attribute
B. with two attributes
C. with three attributes
D. cannot be expressed in the relational data model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The type statement in RDF would be expressed in the relational data model by a table …

A. with one attribute
B. with two attributes
C. with three attributes
D. cannot be expressed in the relational data model

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In RDF everything is a resource. In particular, RDF statements are considered as resources and therefore it should be possible to make statements about them. From the viewpoint of the Semantic Web this is in fact extremely important. Since statements do not express some necessarily agreed upon facts, but often different interpretations of data, it is important to provide the possibility of annotating annotations (such as illustrated in the simple example). This allows to comment on annotations, to agree or dispute them etc..

When considering the graph representation of RDF, it is not immediately clear of how to treat a statement as a resource, since a statement consists of three structural elements, the subject, the object, and the predicate. But we can apply the same "trick" as we did already for complex objects and collections. We introduce a new resource which serves as representation for the statement. This resource has as properties all the constituents that make up the statement it represents. This process is called *reification*. It is straightforward to connect the resource representing the statement it to it's subject and object, as they are both resources themselves. Also, the type of the object can be determined through a property rdf:type pointing to the special resource rdf:statement representing the category of statements. For the predicate, a specific new resource representing the predicate is required. We will see later, when introducing RDF schema, that this new resource is indeed part of a schema for RDF statements, expressing that statements using these properties are valid in the given context. The reified RDF statement is connected to its constituents through the special RDF properties rdf:object, rdf:subject and rdf:property. By reifying a statement, one creates a new resource, which can be anonymous, if no identifier is associated with it using rdf:ID, or can be referenced if it has such an identifier.

RDF Reification - Syntax

The statement has an anonymous resource as subject, namely the reified statement which is fully characterized by its properties!

```
<rdf:RDF>
 <rdf:Description about="#triple123">
  <rdf:subject resource="http://www.doc.ch"/>
  <rdf:predicate resource="http://description.org/schema#Creator"/>
  <rdf:object>John Smith</rdf:object>
  <rdf:type resource="http://www.w3.org/TR/WD-rdf-syntax#Statement"/>
  <dc:Creator>Anne Gold</dc:Creator>
 </rdf:Description>
</rdf:RDF>
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The XML encoding of reified statements follows the principles that have been introduced before. The reified statement is represented as a complex, anonymous object.

RDF Schema - Classification

RDF resources

- anything that can be described

RDF classes

- Categories for subjects and objects
- Classes allow to associate a type with an RDF instance
- Different classes can be in a subClass relationship (be included in each other)
- Classes are themselves RDF instances

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Finally, we introduce RDF Schema. RDF Schema provides two basic mechanisms.

1. Categorization RDF resources, into *classes*.
2. Constraints on the possible use of properties, in the form of constraints expressing which classes can participate as subjects and objects in statements using a specific property.

First, we describe the classification mechanism. Classes are represented themselves as resources, which are of type rdfs:Class, a special class in the RDFS specification. The type property rdf:type is used, as in RDF, to indicate the type of a *class resource*. If an application resource is of a specific type, then the resource is connected to the corresponding class resource via the rdf:type property. In the illustration two examples of such a case are included: the resource with ID „John Smith" belongs to the class (or is of type) Person and the resource "Married" is of type MaritalStatus, which is a resource representing a specific predicate type. Between different classes a subclass relationship can be specified, by using the attribute rdfs:subClassOf. The intended semantics is that any resource belonging to the subclass also belongs to the superclass (containment relationship).

An interesting aspect of RDFS is the modeling of the RDF and RDFS concepts within RDFS itself. This is done by introducing a meta-class level that models the modeling constructs and reflects the paradigm that in RDF everything is a resource, including the concepts introduced by RDF and RDFS. In particular, classes are sets of resources, and thus the rdfs:Class resource is a subClass of rdfs:Resource. Application classes, such as "Animal", are sets of resources, and thus also subclass of rdfs:Resource. On the other hand, the type of a resource is indicated by the rdf:type property. Since rdfs:Resource is a class, it is connected via the rdf:Type property to the rdfs:Class resource. This produces the cyclic structure that we can observe within the RDF meta-class schema. In this figure only a small fragment of the RDFS meta-class schema is shown.

RDF Schema - Properties

RDF properties: connect resources

- The RDF instance must have the properties that are declared for the class
- rdfs:domain: classes of which the instances may have a property
- rdfs:range: classes of which the instances may be the value of a property

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The second important concept of RDFS is the possibility to constrain the usage of RDF properties that connect resources. As everything in RDF, RDF properties are resources themselves. Thus, they are represented in an RDF schema as resources. For example, maritalStatus is a resource representing a property. In RDF schema it is now possible to constrain the usage of properties as follows: by connecting the property resource through the property rdfs:domain to a class resource, one specifies that the subject when using this property must originate from that class, i.e., be of the type of this class. Similarly for the object, the range can be constrained using rdfs:range. Ranges can also be of atomic type, in that case one connects the property resource to (predefined) resources representing the data type of the atomic type. In the example above this is the atomic type STRING.

The RDFS model bears a lot of similarities with object-oriented model. However, a fundamental difference is that properties (attributes in OO terminology) are defined independently of classes.

RDF Schema - Syntax

```
<rdf:RDF xml:lang="en"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdfs:Class rdf:ID="Person">
        <rdfs:subClassOf rdf:resource="http://www.w3.org/classes#Animal"/>
</rdfs:Class>
<rdf:Property ID="maritalStatus">
        <rdfs:range rdf:resource="#MaritalStatus"/>
        <rdfs:domain rdf:resource="#Person"/>
</rdf:Property>
<rdf:Property ID="name">
        <rdfs:range rdf:resource="http://www.w3.org/classes#String"/>
        <rdfs:domain rdf:resource="#Person"/>
</rdf:Property>
<rdf:Property ID="age">
        <rdfs:range rdf:resource="http://www.w3.org/classes#Integer"/>
        <rdfs:domain rdf:resource="#Person"/>
</rdf:Property>
<rdfs:Class rdf:ID="MaritalStatus"/>
<MaritalStatus rdf:ID="Married"/>
<MaritalStatus rdf:ID="Divorced"/>
<MaritalStatus rdf:ID="Single"/>
<MaritalStatus rdf:ID="Widowed"/>
</rdf:RDF>
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Since RDF schemas are expressed as RDF statements they can be encoded into XML along the same principles that we have introduced for RDF statements earlier. This example shows the complete encoding of the RDF schema we have used in our example before. Throughout the schema the abbreviated syntax for statements is used, replacing the element name "Description" by the corresponding class name of the subject of the description. Note of how using the ID attribute, in the RDF schema new resources are introduced. They can be referred to from other statements using the newly introduced identifier prefixed with #. The specification of the class with ID "MaritalStatus" includes the specification of the complete extension of the class, enumerating all possible values the members of this class can take, i.e., all possible predicate names that predicates of type MaritalStatus can take. Strictly speaking, the statements creating the instances of class are not part of the schema level but part of the instance level of RDF.

Which is true?

A. Reification is used to produce a more compact representation of complex RDF statements
B. Reification requires to make a statement the subject of another statement
C. Reified statements always make a statement about another statement

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Which of the following are part of the RDF schema language?

A. The « type » statement for RDF resources?
B. The « domain » statement for RDF properties?
C. The « subject » statement for RDF statements?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.2.5 Semantic Web Resources

Examples of Popular Ontologies and Knowledge Bases

- WordNet
- WikiData
- Google Knowledge Graph
- Schema.org
- Linked Open Data

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the following we provide an overview of the common resources and standards used today in the Semantic Web.

WordNet

English dictionary with semantic relationships

Synonymy. Words that have similar meanings, e.g., happy and glad.

Antonymy. The opposite of synonymy, e.g., happy and sad.

Nouns only

Hypernymy. Hierarchical relationship between words, e.g., furniture is a hypernym of chair since every chair is a piece of furniture.

Hyponymy. Opposite of hypernymy. Dog is a hyponym of canine since every dog is a canine.

Meronymy. Part-whole relationship. For example, paper is a meronym of book, since paper is a part of a book.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

WordNet is a quite old project conducted at Princeton University. The objective of the project is to provide a complete dictionary of the English language, including all semantic relationships among words. In the meantime, similar projects have been started also for other languages.

WordNet Example

http://wordnet.princeton.edu/

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The data of WordNet is publicly available and can be accessed online. It has been used in a variety of projects related to problems sich as query expansion in retrieval, word sense disambiguation or or text classification.

WikiData

Community project to create an open database of structured data

- Data curation model like Wikipedia
- Intended to support Wikipedia (InfoBoxes)
- 84 million entities (2020)
- Multi-lingual
- Both API access and full databases dumps (JSON, RDF)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

WikiData is a compagnion project of WikiPedia. It aims at creating a universal knowledge graph and is based on the same community approach as WikiPedia. It is used by WikiPedia to provide the InfoBoxes.

Example WikiData

www.wikidata.org

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Semantic Web - 51

The WikiData project uses (a part of) the RDF model to represent the facts. One interesting aspects is that it correlates the representation of the terms describing an entity or concept across languages.

Google Knowledge Graph

Google's internal knowledge base to support the search engine

- Populated from FreeBase and internal data
- Enriched with the support of **schema.org**
- Accessible through API

https://www.google.com/intl/es419/insidesearch/features/search/knowledge.html

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Google has popularized knowledge graphs by the development of its knowledge graph. The project actually started from an earlier community effort called FreeBase. It was then enriched with internal data from Google. It uses schema.org as the underlying data model.

Schema.org

Collaborative, community activity to create, maintain, and promote schemas for structured data on the Internet

- Sponsoring companies: Google, Microsoft, Yahoo and Yandex
- Two type hierarchies: textual property values, things that they describe
- Core vocabulary currently consists of 642 Types, 992 Properties, and 219 Enumeration values
- Used by other knowledge bases, e.g., Google Knowledge Graph API, Dbpedia, etc.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Schema.org is a generic data model, or knowledge base schema, that has been adopted and promoted by the large Internet platforms. It consists of two hierarchies, one for descibing types of attributes and one for describing types of entites.

Example

Core plus extension vocabularies

- Thing
  - Action
    - AchieveAction
      - LoseAction
      - TieAction
      - WinAction
    - AssessAction
      - ChooseAction
        - VoteAction
      - IgnoreAction
      - ReactAction
        - AgreeAction
        - DisagreeAction
        - DislikeAction
        - EndorseAction
        - LikeAction
        - WantAction
      - ReviewAction

Action

Thing > Action

An action performed by a direct agent and indirect participants upon a direct object. Optionally happens at a location with the help of an inanimate instrument. The execution of the action may produce a result. Specific action sub-type documentation specifies the exact expectation of each argument/role.

See also blog post and Actions overview document.

Usage: Between 100 and 1000 domains

[more...]

Property	Expected Type	Description
Properties from Action		
actionStatus	ActionStatusType	Indicates the current disposition of the Action.
agent	Organization or Person	The direct performer or driver of the action (animate or inanimate). e.g. "John" wrote a book.
endTime	DateTime	The endTime of something. For a reserved event or service (e.g. FoodEstablishmentReservation), the time that it is expected to end. For actions that span a period of time, when the action was performed. e.g. John wrote a book from January to "December". Note that Event uses startDate/endDate instead of startTime/endTime, even when describing dates with times. This situation may be clarified in future revisions.
error	Thing	For failed actions, more information on the cause of the failure.
instrument	Thing	The object that helped the agent perform the action. e.g. John wrote a book with "a pen".
location	Place or Text or PostalAddress	The location of for example where the event is happening, an organization is located, or where an action takes place.
object	Thing	The object upon the action is carried out, whose state is kept intact or changed. Also known as the semantic roles patient, affected or undergoer (which change their state) or theme (which doesn't). e.g. John read "a book".
participant	Organization or Person	Other co-agents that participated in the action indirectly. e.g. John wrote a book with "Steve".
result	Thing	The result produced in the action. e.g. John wrote "a book".
startTime	DateTime	The startTime of something. For a reserved event or service (e.g. FoodEstablishmentReservation), the time that it is expected to start. For actions that span a period of time, when the action was performed. e.g. John wrote a book from "January" to December. Note that Event uses startDate/endDate instead of startTime/endTime, even when describing dates with times. This situation may be clarified in future revisions.
target	EntryPoint	Indicates a target EntryPoint for an Action.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is a sample of schema.org. It is easy to see that the model covers very generic concepts that can be applied to a wide range of applications.

Encoding

Different encodings can be used, JSON, RDFa, Microdata

RDFa: microformat for embedding RDF into HTML

```
<div vocab="http://schema.org/" typeof="Person">
  <span property="name">Jane Doe</span>
  <img src="janedoe.jpg" property="image" alt="Photo of Jane Joe"/>
  <span property="jobTitle">Professor</span>
  <div property="address" typeof="PostalAddress">
    <span property="streetAddress">
      20341 Whitworth Institute
      405 N. Whitworth
    </span>
    <span property="addressLocality">Seattle</span>,
    <span property="addressRegion">WA</span>
    <span property="postalCode">98052</span>
  </div>
  <span property="telephone">(425) 123-4567</span>
  <a href="mailto:jane-doe@xyz.edu" property="email">
    jane-doe@xyz.edu</a>
  Jane's home page:
  <a href="http://www.janedoe.com" property="url">janedoe.com</a>
  Graduate students:
  <a href="http://www.xyz.edu/students/alicejones.html" property="colleague">
    Alice Jones</a>
  <a href="http://www.xyz.edu/students/bobsmith.html" property="colleague">
    Bob Smith</a>
</div>
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Schema.org can be encoded in any of the existing data exchange standards. One specific type of encoding is RDFa. It is used to embed RDF statements into HTML documents in a standardized approach.

Linked Open Data

Repository of open data and knowledge bases and tools

https://www.w3.org/wiki/DataSetRDFDumps

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Given the rapid adoption of knowledge graphs and the growing number of knowledge bases published, Linked Open Data is an effort to provide an common entry point to a large number of knowledge graph. An interesting aspect in this effort is the fact that knowledge bases often have references among each other. Instead of having isolated knowledge bases, this results in a network of knowledge graphs that is captured and display on the Linked Open Data Website.

Linked Open Data Example

DataSetRDFDumps

Linked Data Sets (i.e., with Dereferenceable URIs) available as RDF Dumps

- Please provide the URL for the directory containing the RDF dump files.
- Please try to have one directory or tarball per dump set -- such that we can retrieve and load the entire URL contents, to have a restored snapshot.
- Please include a Publisher/Maintainer URI, for use in constructing attribution triples.

Project	Data Exposed	Size of Dump and Data Set	Archive URL	Publisher / Maintainer URI
Addgene	Addgene catalog (tab delimited file)	1.1 MB	tab-delimited file	
Allen Brain Atlas	Science Commons extract from ABA Web site, on or shortly before 26 Feb 2007	51 MB	dump file	
Airport Data	SPARQL	754,585		Rob Styles
BAMS	BAMS	5.6 MB	bams-from-swanson-98-4-23-07.owl	
BBC John Peel sessions from DBtune.org	holding data released during Hackday, 2007	277,000 triples	peel.tar.gz	[URI?]
BBOP	All OBO ontologies	36 MB	obo-all.tgz	[URI]
BBOP	selected OBO ontologies, downloaded ~21 April 2007, augmented with inferred relations	2.6 MB	obo-in-owl.tgz	[URI]
Billion Triples Challenge Dataset 2008	various dumps	1 billion triples	download page	
Billion Triples Challenge Dataset 2009	crawled Web data	1.14 billion triples	download page	
Billion Triples Challenge Dataset 2010	crawled Web data	3.2 billion triples	download page	
Bio2RDF	various bio- and gene- related datasets	10+ billion triples	download page	
Bitzi	collaborative file describing service	330,026 discrete files, 270MB uncompressed	dump directory	
British Geological Survey (BGS) OpenGeoscience	1:625 000 Geology of the UK (DigMapGB), BGS geochronology and chronostratigraphy, BGS Lexicon of Named Rock Units	Approx 840,000 (19 MB compressed) N-Triples	data_bgs_ac_uk_ALL.zip	British Geological Survey (BGS)
	Chef Moz	290344 restaurants - 104856 reviews - 59243 links to reviews - 2402 editors	size?	URL?
Data-gov Wild	Datasets containing RDF data converted from datasets published at http://data.gov (and other sources). The datasets are clustered by dc:subject, e.g. government budget, environmental statistics, housing and population statistics, medical cost, energy consumption, public library statistics, and labor statistics.	5+ billion triples	dump directory	Tetherless World Constellation
DBpedia	Data set containing extracted data from Wikipedia. About 2.6 million concepts described by 247 million triples, including abstracts in 14 different languages	247 million triples	dump directory	

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

LoD provides a common point of access to obtain a large number of public knowledge bases.

Which of the following is NOT an (instance-level) ontology?

A. Wordnet
B. WikiData
C. Schema.org
D. Google Knowledge Graph

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

References

Relevant articles

- Grigoris Antoniou, Frank van Harmelen, Semantic Web Primer, MIT Press, 2nd edition, 2004.
- Jeen Broekstra, Michel C. A. Klein, Stefan Decker, Dieter Fensel, Frank van Harmelen, Ian Horrocks: Enabling knowledge representation on the Web by extending RDF schema. WWW 2001: 467-478
- Stefan Decker, Sergey Melnik, Frank van Harmelen, Dieter Fensel, Michel C. A. Klein, Jeen Broekstra, Michael Erdmann, Ian Horrocks: The Semantic Web: The Roles of XML and RDF. IEEE Internet Computing 4(5): 63-74 (2000)

WebSites

- XML, XPath, Xquery, RDF: http://www.w3.org/
- OWL: http://www.w3.org/TR/2004/REC-owl-features-20040210/#s1.1

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.3 INFORMATION EXTRACTION

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Populating Knowledge Bases

Manual creation of knowledge bases is expensive

Can we produce them automatically?

Idea: Extract knowledge from documents

Challenge: Knowledge is encoded in natural language

Objectives

- Automated or accelerated creation of knowledge bases
- Support for structured search on documents

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Traditionally knowledge bases are created manually, either by experts (e.g., WordNet) or by crowd-sourcing (e.g., WikiData). This is expensive. In the case of WordNet, it took tens of years to construct the knowledge base, in the case of WikiData (resp. WikiPedia) we all know about the notorious difficulty to finance this endeavor. Therefore, an interesting question is whether such knowledge bases could not be automatically constructed.

For automatic construction we can exploit data that is digitally available, e.g., all documents accessible on the Web. These documents encode massive human knowledge in natural language. The challenge is to extract such knowledge by analyzing natural language text, which is not an easy problem.

The results would, however, be immensely useful. First, we could create massive knowledge bases in a nearly automated way, and furthermore these knowledge bases could be used to annotate documents, for supporting more expressive and precise searches and analyses.

Information Extraction

From text to knowledge graphs

The New York Times

MARTIN LUTHER KING IS SLAIN IN MEMPHIS; A WHITE IS SUSPECTED; JOHNSON URGES CALM

Who are the entities?

What are their attributes?

How are they related?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For investigating knowledge extraction, more commonly called information extraction, from textual content, we can consider the different constituents of a knowledge graph separately: entities, attributes and relationships. We will now introduce methods for extracting entities and then for establishing relationships among entities and with attributes.

4.3.3 Information Extraction (IE)

Task: Extract statements from text
→ creation of knowledge graphs

EPFL is one of the two **Swiss Federal Institutes of Technology.** With the status of a national school since 1969, the young engineering school has grown in many dimensions, to the extent of becoming one of the most famous **European** institutions of science and technology. Like its sister institution in **Zurich, ETHZ**, it has three core missions: training, research and technology transfer. Associated with several specialised research institutes, the two **Ecoles Polytechniques** (**Institutes of Technology**) form the **EPF domain** , which is directly dependent on the **Federal Department of Economic Affairs, Education and Research (EAER)**.

EPFL is located in **Lausanne** in **Switzerland**, on the shores of the largest lake in **Europe**, **Lake Geneva** and at the foot of the **Alps** and **Mont-Blanc**. Its main campus brings together over 11,000 persons, students, researchers and staff in the same magical place.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Taking the analysis of documents one step further, we now consider the extraction of statements from natural language text, as is illustrated in the example. Statements connect entities through relationships. For example, the first statement expresses that EPFL is part of a larger organization, the Swiss Federal Institutes of Technology. The notion of statement we use here corresponds exactly to the notion of statement we introduced earlier with RDF.

Sample Statements

EPFL is one of the two **Swiss Federal Institutes of Technology**

EPFL - IS-A - Swiss Federal Institute of Technology

its sister institution in **Zurich, ETHZ**

EPFL - RELATED-TO - ETHZ

EPF domain , which is directly dependent on the **Federal Department of Economic Affairs**

EPF Domain - DEPENDS-ON - FDEA

EPFL is located in **Lausanne**

EPFL – LOCATED-IN - Lausanne

Lake Geneva and at the foot of the **Alps**

EPFL – LOCATED-IN - Alps ? Lake Geneva – LOCATED-IN – Alps?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Looking more closely at some of the statements we can extract from the text, we make the following observations. First, statements are always anchored in two entities, thus they link two entities by a relationship. Second, the relationships can carry different meanings, which in natural language are typically expressed in verbs. Third, the extraction of statements can be ambiguous. In the last example, when looking at the original text, the implied meaning is that EPFL is close to the alps. When looking only at the local context of ”Lake Geneva” and “Alps”, one might make also the incorrect inference that the statement is about Lake Geneva located in the alps. Accidentally this is not a wrong statement, but not the one that is intended in the text. So, statement extraction can be a tricky task due to the ambiguity and complexity of human language.

Typed Statements

EPFL – PART-OF – Swiss Federal Institute of Technology

Type: ORG – PART-OF – ORG

EPFL – RELATED-TO – ETHZ

Type: ORG – RELATED-TO – ORG

EPF Domain – PART-OF – FDEA

Type: ORG – PART-OF – ORG

EPFL – LOCATED-IN - Lausanne

Type: ORG – LOCATED-IN – LOC

EPFL – LOCATED-IN - Alps ? Lake Geneva – LOCATED-IN – Alps?

Type: ORG – LOCATED-IN – LOC

Type: LOC – LOCATED-IN – LOC

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With NER we can extract entities of given types. Using entity types, we can also introduce statements of given types. The types in a statement can concern its three components, the subject, the object and the predicate. This allows to exclude statements that are meaningless, like a location being part of a person. It also helps in the task of detecting statements, since only entity pairs and predicates that match the type constraints need to be considered.

Note that the same entity types can be related by predicates of different types. For example, two universities could be related by relationships “perform joint research” and “exchange students”. When using a model for knowledge graphs, such as RDF, the types of the statements would be specified in the schema, e.g., RDF Schema.

Statement Schema

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an example of a possible schema that constrains the type of statements to be considered.

Approaches to Information Extraction

- Hand-written patterns
- Supervised machine learning
- Bootstrapping
- Distant supervision
- Matrix Factorization

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information extraction is a central problem in interpreting and using text data, and has therefore attracted a lot of interest. Many different methods have been developed, of which we will now introduce some of the most important examples. Note that even in the recent time transformer models are successfully applied to this task as well, the earlier models have still their interest as they are often computationally less expensive and incorporate relevant ideas on how to perform information extraction that are also of interest for developing mode complex models.

4.3.3.1 Hand-Written Patterns

Early approach from **Hearst (1992)**

- "Agar is a substance prepared from a mixture of **red algae, such as Gelidium**, for laboratory or industrial use"

Patterns to detect IS-A relationships:

```
"Y such as X ((, X)* (, and|or) X)"
"such Y as X"
"X or other Y"
"X and other Y"
"Y including X"
"Y, especially X"
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

An early approach to information extraction is based on the observation that in natural language a relationship is often expressed in a regular fashion. This observation has been exploited to extract specific relationships, such as ISA, by using regular expression patterns. Hearst was one of the first to introduce this approach, and the method is still being used till today. The performance of the method depends on the quality and diversity of patterns used.

Web isa Database

Large scale extraction of IS-A relationships from web documents

	PreTerm	Term	PostTerm	PrecClass	Class	PostClass	Frequency
1	darth	vader		star wars	character		167
2	darth	vader			character		83
3	darth	vader			villain		43
4	darth	vader			none		41
5	darth	vader			dad		34
6	darth	vader		iconic	character		34
7	darth	vader			dad	like any otherexcept	29
8	darth	vader			great		21
9	darth	vader		good	father		21

http://webdatacommons.org/isadb/

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Web is a DB is an example a large-scale effort to extract IS-A relationships from Web data using Hearst patterns. The patterns are more complex than the ones initially introduced by Hearst, but the basic idea is the same.

More General Hand-Written Patterns

Idea: relations often hold between specific entity types

- located-in (ORGANIZATION, LOCATION)
- founded (PERSON, ORGANIZATION)
- cures (DRUG, DISEASE)

First, perform Named Entities Recognition

Use typed pattern: ORG is located in LOC, LOC

EPFL is located in **Lausanne**, **Switzerland**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The idea of Hearst that has been successfully applied for detecting ISA relationships, can be extended to patterns of a more general type, for extracting statements for other types of relationships. This approach exploits the fact that certain relationships can only hold among certain types of entities. Thus, in a first step a named entity recognition is performed, and subsequently the patterns are searched for which the matching types of entities can be found.

Summary Hand-Written Patterns

Advantages

- Rules tend to be high-precision
- Can be tailored to specific domains

Disadvantages

- Human patterns are often low-recall
- A lot of effort to think of all possible patterns

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Hand-written patterns are in general a very reliable method for information extraction. However, it suffers from low recall, and it is very difficult to conceive all possible patterns by a human expert. Therefore, more automated methods are often preferrable.

4.3.3.2 Supervised Learning for IE

Approach: train a classifier on labeled data

Creating a training set

- Choose relevant named entities and their relations
- Manually label relations among entities (positive examples)

http://mars.cs.utu.fi/BioInfer/

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A supervised learning approach for information extraction requires training data. Producing such a training set is labor-intensive. For example, in the domain of life sciences major efforts have been undertaken. In the example shown, more than 2000 sentences have been manually annotated, by identifying both the entities and their relationships.

Classifiers for Information Extraction

Two-step approach

- A **filtering classifier** (e.g., Naïve Bayes), to detect whether a relation exists among the entities
- A **relation-specific classifier** detecting the relation label

Training the classifiers

- Extract named entities in the document corpus using NER
- Detect pairs of entities, e.g., in the same sentence
- Use unlabeled entity pairs as negative examples

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Provided manually annotated training data, classifiers for information extraction can be trained. The standard approach is to train two types of classifiers, one that detects whether a relationship exists among two entities, and a second one that determines the type of the relationship. The use of a filtering classifier for detecting relationships can speed up the classification task and allows to use of different features for the two tasks.

For training the classifiers, one first extracts all entity pairs using NER that occur in the same context, for example, in the same sentence. If the pairs are not annotated, they can be taken as negative examples. Then the classifier is trained using features extracted from the context of the occurrences of the entities.

Features Used in Information Extraction

EPFL is located in **Lausanne** in **Switzerland**, next to **Lake Geneva**

Features for mention (M1, M2) = (Lausanne, Lake Geneva):

BOW and bigrams in the sentence:	is, located, in, located in, Lausanne, in, next to …
BOW and bigrams in between the mentions:	in, Switzerland, next, to, next to
Headwords* of mentions, their concatenation:	Lausanne, Geneva, Lausanne-Geneva
Words in positions:	M1-1: in, M1+1: in, M2-1: to
Types of entities	LOC, LOC
Stemmed version of the words	
Syntactic features	
…	

*headword = entry in a dictionary

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Diverse text features can be used for training a classifier for information extraction. After detecting entity pairs using NER, the so-called mentions, the features for relation extraction can use features of the both entities as well as of the text surrounding them and in between.

When we have identified two occurrences of named entities in a sentence, also called two mentions (M1, M2), then we can identify the following features related to the two mentions:

- The bag of words and bigrams found within the whole sentence
- Separate from that the BOW and bigrams in between the mentions
- The headwords, which are words that are found in a standard dictionary
- Words in specific position with respect to the mentions
- Stemmed versions of all the words above
- The type of entities used
- Syntactic features, extracted with part-of-speech analysis

Syntactic Features

Parse Tree

```
(S (NP EPFL)
        (VP is
                (VP located
                        (PP in
                                (NP Lausanne))
                        (PP in (NP Switzerland ,
                                (PP next to
                                        (NP Lake Geneva)))))))
```

Features:

Sequence between entities: PP NP PP NP

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The syntactic features, or part-of-speech tags can be exploited in various ways. In the simplest case only the sequence of POS tags in between the mentions is used. More complex features can be constructed as well, e.g., the navigation path between the mentions in the parse tree. For trying out POS tagging you can try: https://www.link.cs.cmu.edu/link/submit-sentence-4.html

Which is true?

A. Hand-written patterns are in general more precise than classifiers
B. Hand-written patterns cannot exploit syntactic features
C. Supervised classifiers do not require any human input
D. Supervised classifiers can only detect typed statements

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.3.3.3 Bootstrapping

No training data, but a few high-precision patterns

Approach:

- Find entity pairs that match the pattern
- Find sentences containing those entity pairs
- Generalize the entities in those sentences
- Generate new patterns

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One of the big problems with supervised approaches to information extraction is the scarcity of training data. One way to avoid the use of training data for information extraction is called bootstrapping. The basic idea is that on a few high-precision patterns, such as the Hearst patterns, one generalizes these patterns by analyzing a large text collection.

The approach is to first find entity pairs using the high precision patterns. Then using those entity pairs sentences containing the same entity pairs are searched. The assumption is that with large likelihood these sentences express the same type of relationship, just in a different syntactic representation. Thus, such sentences can be considered as text templates for expressing the relationship from which new patterns for detecting the relationship can be derived.

This figure illustrates the different steps of how new patterns are detected and applied to find new relationships. P1 is a known pattern for detecting the relationship. It is used to detect matching entity pairs. Then, in a second step, such pairs are searched. The text associated with these pairs can then be used to infer new pattern P2.

Example

Pattern: **LOC** is located in **LOC**

- Mumbai **is located in** India
- Adelaide **is located in** Southern Australia
- Sriharikota **is located in** Nellore

Search for entity pairs (Mumbai, India)

- **Mumbai** is **India's** top destination
- **Mumbai** hotels, **India**

New patterns

- **LOC** is **LOC**'s top destination
- **LOC** hotels, **LOC**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates the approach. We start with a simple, but precise pattern, LOC is located in LOC, which we use to find occurrences of entity pairs. Then we search for those entity pairs and find other sentences mentioning them. These are then generalized to patterns by replacing the entity occurrences by their types.

Problem: Semantic Drift

Example: The pattern

LOC hotels, **LOC**

matches also

… Geneva hotels, Lausanne hotels …

➔ Geneva is located in Lausanne?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A potential problem when using this approach is that the new patterns that are found are potentially misleading and may create inaccurate results. The example shows an instance of such a problem.

Confidence

Assume we have a confirmed set of pairs of mentions M

- A new pattern should also match many of those

$Hits_p$ = number of pairs in M that a new pattern matches

$Finds_p$ = total number of pairs that a new pattern matches

Confidence that a new pattern finds many relevant mentions

$$Conf(p) = \frac{Hits_p}{Finds_p} \log(Finds_p)$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to contain the problem of inferring too many inaccurate patterns, a confidence metric can be used. With this metric the confidence into a new pattern increases, when the fraction of matched entity pairs that are already confirmed to be correct is large. Otherwise said, if a new patterns frequently matches, but only few of the matched entity pairs are known to be in the intended relation, the confidence in the new pattern will be low.

Confidence in Statements

A statement s may match a set of patterns P

- Each pattern with a given confidence

Assuming confidence is a probability

Probability that the statement s is correct

$$Conf(s) = 1 - \prod_{p \in P} (1 - Conf(p))$$

Only statements with sufficient confidence are used to infer new patterns

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When inferring new patterns with bootstrapping, statements that have been spotted in the text involving the relevant entities are considered as new templates. In order to avoid semantic drift, one can be selective about which statements to consider for creating new patterns, by considering confidence of the rules that match the specific statement. This can be done by using the confidence of patterns and considering it like probabilities. The product of the values of $(1 - Conf(p))$ can then be considered like the probability of all patterns being wrong, and therefore the complement being the probability of at least one pattern being right.

4.3.3.4 Distant Supervision

No Training Data? **Idea**: use existing knowledge bases to collect training data for building a classifier

- Combines advantages of bootstrapping with supervised learning

Example: learning PLACE-OF-BIRTH

WikiData has many positive examples!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Another approach to address the problem of lacking training data is to use knowledge bases that contain large number of acts, such as WikiData. The facts recorded in such knowledge bases can be used to spot corresponding statements in text. Such statements can then be used as text patterns for fact representation, that can be provided as training data to a classification algorithm for information extraction.

Linking Text to Knowledge Bases

Using Entity extraction, entity mentions in text can be linked to corresponding mentions in a knowledge base

John was born in **Liverpool**, to **Julia** and **Alfred Lennon**

Entities		Text Features			Relation from knowledge base	
Entity 1	Entity 2	PER was born in LOC	PER was born to PER	PER and PER	Birthplace(X,Y)	Married(X,Y)
John Lennon	Liverpool	x			?	
John Lennon	Julia Lennon		x			
John Lennon	Alfred Lennon		x			
Julia Lennon	Alfred Lennon			x		?
Barack Obama	Hawaii	x			x	
Barack Obama	Michelle Obama			x		x

Barack Obama was born in **Hawaii. Barack** and **Michelle Obama …**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In this figure we illustrate of how entity pairs identified in text can be linked to the same entity pairs found in a knowledge base. This allows to infer the meaning of syntactic patterns (e.g., ”was born in”) and relate such syntactic patterns to the corresponding relationship (e.g., birthplace). In the example, the system can first learn from the knowledge base text patterns for known facts (about Barack Obama), and then use those patterns to infer new facts from text (about John Lennon). The matrix shown in this figure we call the entity-pairs/relation matrix.

The figure illustrates the steps in the distant supervision method. First, it starts by doing NER over a large document collection, such as WikiPedia. This produces a set of sentences s that contain entity pairs (e1, e2). Having those entity pairs, it searches in the knowledge base, in this example WikiData, whether relations among the entity pairs exist. For those entity pairs, for which a relation is confirmed in the knowledge base, it can generate an instance for the training data. The training data consists of the entity pair, the type of relations and features extracted from the text. The result is a training data set that can be used to train a classifier to predict relations for new documents. The classifier extracts from a new document the entity pairs and feeds them together with the text features into the classifier to obtain the type of relationship.

Features for Distant Supervision

Use conjunctions of standard IE features as sentence features

- Match only if all individual features match
- High precision, but low recall features!
- Feasible, since training set is large

Complex features resemble to templates used in rule-based approaches

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With distant supervision it is possible to obtain many training samples. Therefore, it is possible to use a large number of features for classification. One way to obtain a large number of features is to consider each conjunction if individual text features as a separate feature. Such features resemble then to patterns that are used in rule-based approaches. They have typically low recall, but higher precision.

Example

Example sentences

- **Mumbai** hotels, **India**
- **Geneva** hotels**, Lausanne** hotels

Individual features

F1: M1=LOC, F2: M2=LOC, F3: between={hotels} , F4: after={}
F1: M1=LOC, F2: M2=LOC, F3: between={hotels} , F4: after={Switzerland}

Complex Features

CF1: M1=ORG and M2=LOC and between={hotels} and after={}
CF2: M1=ORG and M2=LOC and between={is, located, in} and after={hotels}

the two sentences have two different, unrelated complex features

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we illustrate the use of complex features. Assume we have a set of basic features, like those that we have identified for supervised information extraction. F1 to F6 are examples of such features. These features could as such be supplied to a classification algorithm, e.g., a Naïve Bayes classifier. Given that the features of the two examples are very similar, only F5 is different, the classifier would probably decide that the relation represented by the two sentences would be the same.

Using the basic features and combining via a conjunction into one complex features, we would obtain two different complex features CF1 and CF2. A classifier that does not further consider the internal structure of these two features would now clearly distinguish the two cases.

Which is true?

A. Distant supervision requires rules for bootstrapping
B. Classifiers produced with distant supervision are more precise than rules
C. Distant supervision can help to detect rules

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.3.3.5 Matrix Factorization

Using the same data as for distant supervision

- Entity pairs from text, linked to relations from knowledge bases

Instead of learning a classifier, create low-dimensional representations for entity pairs and relations

Use those representations to

- Link text patterns to relation types and identify similar text patterns
- Extract relations from text

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Distant supervision aims at generating classifiers for relations that are based on syntactic features. Based on the same data as used for distant supervision we can also create representations for relationships by mapping them to low-dimensional vectors, as we did for words with word embeddings. This is the idea of using matrix factorization to the entity-pairs/relation matrix.

Matrix Representation

Create a matrix with

- Entity pairs as rows
- Relation types as columns
  - Relations from text patterns
  - Relations from knowledge base

The entity-pair/relation matrix is a sparse matrix (like in recommender systems)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The intention of using matrix factorization is align text patterns and relationships that have a related meaning in a latent space. This can help both in identifying text patterns that correspond to relationships and to directly extract those relationships from text.

The entity-pair/relation matrix is a sparse matrix. Thus, the situation is comparable to the one we encountered in matrix factorization for recommender systems. Algebraic factorization does not work. On the other hand, using matrix factorization based on SGD could not only create low-dimensional representations for relationships, but also help to detect new relationships. The idea is like the one used in recommender systems to estimate missing ratings.

Matrix Factorization

$R(i,j)$ positive examples of facts

$R'(x,y)$ negative examples of facts

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

By factorizing the matrix, we construct two factor matrices, one with low-dimensional representations of entity pairs, and one with low-dimensional representations of relationships. An entry R(i,j) in the entity-pair/relation matrix is then represented as the product of the corresponding representation of the entity pair and the relation from the factor matrices. It corresponds to a statement, the one that puts the two entities from the entity pair into the corresponding relation. The entries correspond to the probability that the statement is correct. For known facts, for which an entry in the original matrix exists, the factorized matrix should have high values. Unknown facts, for which no entry exits in the original matrix, are considered as negative examples of facts.

Bayesian Personalized Ranking (BPR)

Idea: give observed true facts higher ranking than unobserved (true or false) facts

Approach: create ranked pairs f^+ and f^-

Objective Function

$$\sum_{f^+,f^-} \log \sigma\left(\theta_{f^+} - \theta_{f^-}\right)$$

where

$$\theta_f = \boldsymbol{p} \cdot \boldsymbol{r}$$

Maximize with Stochastic Gradient Descent

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Assuming the absence of knowledge on a fact, means that the fact is wrong, is an assumption that is too strong. It could as well be that the fact is correct, but we do not have the knowledge. This is like the situation in recommender systems, where a missing rating does not necessarily imply that the rating is negative.

To better adjust to this situation an alternative to matrix factorization is a method called Bayesian Personalized Ranking. It is based on an alternative loss function for SGD. The loss function is constructed by choosing pairs of positive and negative examples of facts and maximizing their difference. More accurately, the logarithm of the sigmoid of the difference is used as optimization objective. Intuitively this objective function says that a known fact is ranked higher than an unknown fact but does not exclude the possibility that the unknown fact is also a correct fact.

Relation Embeddings

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The matrix factorization computed using BPR allows to map entity pairs and relationships in the same low-dimensional space. As illustrated in this figure, this allows to cluster both entity pairs and relations that correspond to similar relationships. This allows to infer new relationships for existing entity pairs, as well as new syntactic patterns for relationships.

Exploiting Relation Embedding Similarity

Similar in embedding space

Similar in embedding space

Entities		Relationship pattern	
Entity 1	Entity 2	COM owns part in COM	COM buys stake in COM
Renault	Nissan	x	x
BMW	Rover		x
Volkswagen	Porsche		
Ford	Toyota		

Possible inferences:

- BMW owns part in Rover (similarity of relationship)
- Volkswagen owns part in Porsche (similarity of entity pair)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates of how the method could be used to extract relationships for previously unknown syntactic patterns, as well as relations among entity pairs that are no previously known.

Question

When searching for an entity e_{new} that has a given relationship r with a given entity e

A. We search for e_{new} that have a similar embedding vector to e
B. We search for e_{new} that have a similar embedding vector to e_{old} which has relationship r with e
C. We search for pairs (e_{new}, e) that have similar embedding to (e_{old}, e)
D. We search for pairs (e_{new}, e) that have similar embedding to (e_{old}, e) for e_{old} which has relationship r with e

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Summary

Information extraction

- Populating knowledge bases and fact databases
- Taxonomy induction

Pattern-based approaches

- High precision, low recall, work intensive

Supervised learning

- Low precision, high recall, work intensive

Hybrid methods: bootstrapping, distant supervision, matrix factorization

Unknown types of relationships: open information extraction

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We have introduced the different methods of information extraction. These methods aim at extracting relationships of which the type is known, i.e., with specific entity types and a known meaning of the relationship. A more general problem in information extraction is so-called open information extraction. It is used for extracting statements from large collections of documents, such as the Web, where the nature of relations is not known a priori.

4.3.4 Taxonomy Induction

Information extraction

- Extract **isolated** facts from documents, e.g., *lion ISA animal*

Taxonomy induction

- Extract **related** facts from documents, e.g., classification of animals

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Information extraction concerns the extraction of isolated facts, such as ISA relationships. Taxonomy induction aims at extracting related facts and organizing them in a structured knowledge graph, e.g., a hierarchical taxonomy. It is a special case of the more general ontology induction, which organizes knowledge using arbitrary relationships.

Use of Taxonomies

Hyponyms (subordinate terms) can inherit properties from hypernyms (more general terms)

- Due to transitivity of ISA, no need to learn inferred facts

No unique taxonomies

- Depending on the perspective and application different taxonomies may be useful:

A tiger and a puppy are both Mammals and hence belong close together in a typical taxonomy, but tiger is a WildAnimal (in the perspective of Animal-Function) and a JungleDweller (in the perspective of Habitat), while a puppy is a Pet (as function) and a HouseAnimal (as habitat), which would place them relatively far from one another

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One of the advantages of taxonomy induction (and more generally ontology induction) is the possibility to perform inferences on the extracted knowledge. For example, in an ISA hierarchy, lower-level nodes in the hierarchy can inherit properties from higher-level nodes, thus these extra facts need not to be learnt separately.

One of the challenges in taxonomy induction is the fact that there is no notion of “correct” taxonomy. A taxonomy strongly depends in its intended use and on the specific perspective of the user on the domain. Integrating all possible perspectives into one single global taxonomy would not be feasible, as likely no agreement on a common view can be reached, and not useful, as the resulting taxonomy would potentially be too complex.

Taxonomy Induction Task

Starting from a root concept and a basic concept

1. Learn relevant terms and their hypernym / hyponym relationships
2. Filter out erroneous terms and relations
3. Induce a taxonomy structure

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We introduce in the following one specific approach to taxonomy induction. It was one of the first approaches proposed. It starts from the assumption that one general concept (e.g., animals) and at least one basic concept of the taxonomy (e.g., lion) are provided as input. From this starting point the task is to identify more relevant concepts, represented as terms, and establish the hypernym and hyponym relationships, . The approach also filters out erroneous terms and relations and finally induces an overall taxonomy structure. The figure illustrates the process: at the bottom level first additional related terms are identified. Then intermediate concepts, more abstract than the basic concepts are found. From this data finally the taxonomy is induced.

Learning Terms

Template approach: double-anchored patterns

- Given a **root concept c** (e.g., animal) and a **seed s** (e.g., lion)
- Hyponym pattern, detecting instances:

  $P_i(c, s, X)$ = **c** such as **s** and X
- Hypernym pattern, using known instances $\mathbf{t_1}$ and $\mathbf{t_2}$, detect classes:

  $P_c(t_1, t_2, X)$ = X such as $\mathbf{t_1}$ and $\mathbf{t_2}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The basic idea is to learn terms and relationships by querying a Web search engine. This is a very simple, but powerful tool to learn about the meaning of words. For querying, language patterns that capture hypernym and hyponym relationships are used. This is analogous to the use of Hearst patterns for extracting ISA relationships from text.

In a first phase the objective is to find more relevant terms. For this so-called double anchored patterns are used. that relate one known term to another unknown term of the same class of concepts. To start this search at least one term and one class need to be known. One example of such a pattern would be “c such as s and X”. Using such patterns new terms can be collected iteratively by applying the pattern to the terms known so far.

In a second phase, once many terms are known, new names of classes can be searched. For this purpose, hypernym patterns, like “X such as $\mathbf{t_1}$ and $\mathbf{t_2}$” can be used.

Finding Hyponyms

Iteratively harvest new terms using a Web search engine

```
T = {s}; w(t) = 0
while T changes
  for all t in T: submit Pi(c, t, X) to search engine
       if result is not empty
            add to T all new terms tnew found
                 in position X in a result
            w(t)++
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is the algorithm for finding new terms. While performing the search the algorithm keeps track with w(t) of the number of times a term t has been leading to the discovery of another term.

Example

"animal such as lion and"

All Images Videos News Shopping More Settings Tools

6 results (0,56 seconds)

General Driving Tips - Safe Overlanding Tips | Avis Safari Rental
https://www.avis.co.za/safari-rental/driving-tips/general-driving-tips
Undertaking repairs or doing vehicle extraction at night increases the risk exposure to opportunistic dangerous wild **animal such as lion and** hyena. The roads in ...

Digication e-Portfolio :: Natali Coronado Malena ePorfolio :: Child ...
https://hostos.digication.com/natali_coronado_malena_eporfolio/Child_Case_Study
He loves vegetables and his favorite vegetable is Zucchini, his favorite sport is basketball and he love the loud and fast **animal such as Lion and** Tiger. He loves ...

PHS 6 SCIENCE Mr.Gary
phs6ta1.blogspot.com/
Feb 22, 2017 - Predation is animal that is hunting other prey or **animal such as lion and** tiger . In prey such as zebra and pig.Example of predation is lion and ...

Download PDF - Springer Link
link.springer.com/content/pdf/10.1007%2Fs10739-007-9147-3.pdf
problem **animal (such as lion and** elephant) management.56 By 1945. Bigalke was insisting that "game preservation is the task of the scientist,. i.e., the work of ...

Existing Term: lion
New Terms: hyena, tiger, elephant

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an example of how terms related to an initial term “lion” would be found using double-anchored patterns. The example illustrates that the method works surprisingly well.

Finding Hypernyms

```
C = {c}; H = {}; w(t_i,t_j,h) = 0
for all t_1, t_2 in T with w(t_i) > 0:
     submit P_c(t_1, t_2, X) to search engine
     add new term h found in position X to C
     add t_1 ISA h and t_2 ISA h to the hypernym relations H
     w(t_1,t_2,h)++
```

Filtering

- rank concepts h by $\sum_{t_1 t_2} w(t_1, t_2, h)$
- keep top concepts

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Once the search for terms is completed, hypernyms can be searched by using the hypernym pattern with term pairs that have been found in the first step. Only terms that have been producing additional terms in the first phase are considered, using the condition w(t)>0, to avoid the addition of noisy concept names. The algorithm keeps track of how often a specific pair has produced a concept name.

Once the search for concept names is completed, a filtering step is performed. The concepts found are ranked by the number of times they have been discovered starting from different term pairs and only the highest ranked concepts are retained.

Example

"such as lion and tiger"

All Images Videos News Shopping More Settings Tools

About 10.600 results (0,30 seconds)

Language at the Speed of Sight: How We Read, Why So Many Can't, an…
https://books.google.ch/books?isbn=0465080650
Mark Seidenberg - 2017 - Science
Having accrued statistics about words **such as LION and TIGER** allows the listener (or, later, the reader) to infer much about the meaning of a new word such as ...

Were early hominids REALLY all that threatened by sabre toothed ...
https://www.thenakedscientists.com/forum/index.php?topic=44769.0
Jul 16, 2012 - 3 posts - 3 authors
It is likely that early humans gave major predators **such as lion and tiger** ancestors a wide berth. It is just too great of a risk to ourselves to hunt ...

Were early hominids REALLY all that threatened by sabre toothed ...
https://www.thenakedscientists.com/forum/index.php?topic=44769.0
Jul 16, 2012 - 3 posts - 3 authors
It is likely that early humans gave major predators **such as lion and tiger** ancestors a wide berth. It is just too great of a risk to ourselves to hunt ...

12 Animals With The Strangest Habit Of Sleeping - INVORMA
invorma.com › Family
Oct 15, 2015 - This unusual habit of sleeping is also applied for the family of big cats **such as lion and tiger**. These cats are also popular as nocturnal hunters.

New Classes: predators, big cats,

But also: words, …

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here a few examples of how higher-level concepts related to the basic concepts “lion” and “tiger” can be found. Note that this step can easily produce noisy terms, therefore filtering is important.

As a result of the two previous steps, we obtain basic data consisting of basic concepts T and higher-level concepts C, together with hypernym relationships indicated by dotted arrows.

Inducing Hypernym Graph

Many possible relationships among concepts have likely not been discovered

```
For each pair t1, t2 in C
     Construct query q1 = h(t1, t2) and q2 = h(t2, t1)
     with Hearst pattern h(X, Y) for ISA,
     e.g., h(X, Y) = "X such as Y"
     Submit query to search engine, count number of result
     If #results(q1) < #results(q2)
          then add t1 ISA t2 to H
          else add t2 ISA t1 to H
```

Result: A directed hypernym graph H

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the steps performed so far, many possible relationships among higher-level concepts and basic concepts, and among different higher-level concepts may not have been discovered. For finding those again queries are posed to a search engine. For each pair of terms designating a concept, using a query the algorithm tests whether they are in a hypernym relationship. For this purpose, standard Hearst patterns for the ISA relationship are used, such as “X such as Y”, “X are Y that”, “X including Y”, “X like Y”, “such X as Y”.

Example

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example shows how powerful this method is to test a direction of a relationship.

After adding all possible relationships to the set of concepts, we obtain a graph that contains many redundant paths. Many of the paths correspond to the transitive closure of other relationships. Hor example, the algorithm would have found that animal ISA chordate, and chordate is a vertebrate, but also that animal is a vertebrate. In a last step such redundant relationships are removed.

Cleaning the Hypernym Graph

1. Determine all **basic concepts**
   - Not hypernym of another concept
2. Determine all **root concepts**
   - Have no hypernyms
3. For each basic concepts - root concept pair:
   - Select all hypernym paths that connect them
4. Choose the longest hypernym paths for the final taxonomy

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For removing the redundant relationships, both root and basic concepts are identified and all paths that connect pairs of them. By choosing for each of those pairs the longest paths, and dismissing the others, all relationships that are part of the transitive closure are removed. Note that multiple longest paths may exist, since the taxonomy can have a lattice structure.

If t has no Hypernym ..

A. It is a root concept
B. It cannot match c such as t and X
C. It is identical to the initial root concept
D. It is a basic concept

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

References

Lecture partially based on

- Dan Jurafsky and James H. Martin, Speech and Language Processing (3rd ed. Draft), Chapter 21 https://web.stanford.edu/~jurafsky/slp3/
- Jay Pujara and Sameer Singh, Mining Knowledge Graphs from Text, Tutorial, https://kgtutorial.github.io

References

- Mintz, Mike, et al. "Distant supervision for relation extraction without labeled data." *ACL 2009*.
- Riedel, S., Yao, L., McCallum, A., & Marlin, B. M. Relation extraction with matrix factorization and universal schemas. *ACL 2013*.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.4 KNOWLEDGE INFERENCE

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Knowledge Inference

From available knowledge to more complete and precise knowledge

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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.

4.4.1 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?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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 $\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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.

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

$$\max(0, \gamma + \mathrm{f}(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t) - \mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'})) = 0$$

if $\mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'}) = 1$, then

$$\max(0, \gamma + \mathrm{f}(\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t) - \mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'})) = 1$$

☞ in order to minimize $J(\theta)$ $\mathrm{f}(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'}))$ needs to be increased

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example illustrates of how the loss function forces a separation of the scores of plausible and implausible tuples.

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 $\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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!)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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	$\|\mathbf{W}_r^h(\theta_r^h)\boldsymbol{v}_h + \boldsymbol{v}_r - \mathbf{W}_r^t(\theta_r^t)\boldsymbol{v}_t\|_{\ell_{1/2}}$; $\mathbf{W}_r^h, \mathbf{W}_r^t \in \mathbb{R}^{n\times k}$; $\theta_r^h, \theta_r^t \in \mathbb{R}$; $\boldsymbol{v}_r \in \mathbb{R}^n$	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	$\boldsymbol{v}_r^\top tanh(\boldsymbol{v}_h^\top\mathbf{M}_r\boldsymbol{v}_t + \mathbf{W}_{r,1}\boldsymbol{v}_h + \mathbf{W}_{r,2}\boldsymbol{v}_t + \mathbf{b}_r)$ $\boldsymbol{v}_r, \mathbf{b}_r \in \mathbb{R}^n$; $\mathbf{M}_r \in \mathbb{R}^{k\times k\times n}$; $\mathbf{W}_{r,1}, \mathbf{W}_{r,2} \in \mathbb{R}^{n\times k}$	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	$\boldsymbol{v}_t^\top g\left(\text{vec}\left(g\left(\text{concat}(\overline{\boldsymbol{v}}_h, \overline{\boldsymbol{v}}_r) * \boldsymbol{\Omega}\right)\right)\boldsymbol{W}\right)$; g denotes a non-linear function	Adam
ConvKB	$\mathbf{w}^\top\text{concat}\left(g\left([\boldsymbol{v}_h, \boldsymbol{v}_r, \boldsymbol{v}_t] * \boldsymbol{\Omega}\right)\right)$; $*$ denotes a convolution operator	Adam

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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 $f(\boldsymbol{v}_{h'}, \boldsymbol{v}_r, \boldsymbol{v}_{t'})$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

4.4.2 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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This example shows of how the probabilities of the labels, or the value unknown are initially distributed.

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)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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 $unknown$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

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

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

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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 than sparsely connected nodes

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Example Model

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

$d_v = (1 - c_v)\sqrt{H(v)}$, $H(v) = -\log\frac{1}{\deg(v)}$ if v is labelled,
$d_v = 0$ otherwise

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

$p_v^{con} = \frac{c_v}{z_v}$, $\quad p_v^{inj} = \frac{d_v}{z_v}$ for labelled nodes, 0 otherwise
$p_v^{aba} = 1 - p_v^{con} - p_v^{inj}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Conceptually there exist three principal approaches of how to address semantic heterogeneity. A first approach is to map all the models to one common global model. This approach is taken with standardization. For example, EDIFACT (https://www.unece.org/cefact/edifact/) is an international standard that models all concepts that are commonly used in business and trade. For exchanging information, the information systems map their data to EDIFACT and can thus share their data with other information systems.

A second approach consists of constructing directly a mapping among two information systems, without using any additional, shared knowledge in form of standards or ontologies

A third approach is to relate the model of an information system to a common model, frequently called ontology, and use this mapping to construct a direct mapping among the different models used in the information systems. . For example, in the financial domain there exists a standard reference ontology called FIBO (https://fib-dm.com/finance-ontology-transform-data-model/)

Direct Schema Mapping

Assume all data represented in canonical data model (e.g. relational)

- detect correspondences (schema matching)
- resolve conflicts
- integrate schemas (schema mapping)

Mappings are frequently expressed as queries (e.g. SQL Query)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Creating direct mappings among information systems has been widely applied for mapping data that is stored in relational, object-oriented and XML databases, where the model is represented as a database schema. The problem is known as the **schema mapping** problem. Given two schemas of databases, first schema elements are identified that are likely to correspond to each other, i.e., that are likely representing the same real-world aspect. This can be done by using many types of analysis using structural and content features of the database schema and the database. Tools for supporting these steps are called schema mapping tools. Once this step is performed some conflicting correspondences may occur, e.g., mapping some concept in one schema to two different ones in the other. These need to be resolved typically through decisions taken by humans. Once the correspondences are consistent, mappings can be automatically derived. The mappings are typically expressed as queries of the data manipulation language.

Schema Matching

Integration of heterogeneous data sources

- Every project on Big Data analysis first has to integrate data from different, heterogeneous data sources
- Different schemas, taxonomies, knowledge graphs
- One of the long-standing open problems in data management

How to find good "matches"?

How to choose the "best matches"?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Schema matching is a long standindg problem, both in industry and research.

Schema Matching Approaches

Manual matching

- still common practice today

Schema matching tools

- Based on structural and content features
  - names, domains, structure, values, …
- Establish correspondences and rank according to quality
  - Errors are frequent and unavoidable
  - Works well for small schemas
- Can now be implemented using LLMs

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Schema Matching Tools

The Problem

Given two hierarchical knowledge graphs D1, D2

Goal: find a 1:1 matching of nodes (entities, classes) that have the same or similar "meaning"

Assumption: two nodes (classes) are considered similar, if they have the same or similar instances (entities)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the following we will study the problem of integrating heterogeneous databases respectively knowledge graphs. We assume that we want to integrate data that are available in two knowledge graphs that are semantically related to each other.

We will make a few assumptions:

- the knowledge graph is hierarchical. It could also be a taxonomy.
- Some entities represent classes. We are in particular interested in matching those.
- Entities that represent classes share similar kinds of instances, which are entities.

Let us assume that we have two example knowledge graphs. By inspecting the databases, a human can easily recognize that class A in graph D1 corresponds to class B in graph D2. The problem is how to perform the task of identifying this correspondence automatically.

Universe of a Database

Universe U is a finite set of possible instances

Example:
U = {u, computer science, mathematics, life sciences, A, c1, c2, c3, information systems, aberer, signal processing, vetterli, databases, ailamaki, …}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One basic approach to assess whether two classes are similar, is to measure their level of similarity at the content level, i.e., to test to which extent the elements (=instances) of the two classes overlap.

Similarity of Classes

A, B are classes, classes are subsets of U

Similarity measure (Jaccard similarity)

$$sim(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{P(x \in A \ and \ x \in B)}{P(x \in A \ or \ x \in B)} = \frac{P(A,B)}{P(A,B)+P(\bar{A},B)+P(A,\bar{B})}$$

where $P(A,B) = P(x \in A \ and \ x \in B)$ and
$P(A,\bar{B}) = P(x \in A \ and \ x \notin B)$ etc

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The Jaccard measure is a standard approach to measure such a set similarity.

Example

$$sim(A, B) = \frac{2}{5}$$

Note: instances of A are
"information systems", "signal processing", "databases"

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In this example the two classes have 2 common elements, and the number of all elements in their union is 5, thus the Jaccard similarity is 2/5.

Problem 1

A similar to B?

Same information has **structurally different representation**

$$sim(A, B) = 0?$$

Note: instances of A are c1, c2, c3

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the previous example, the instances of the two classes have been leaf nodes in the knowledge graph. Therefore, it was straightforward to compare their instances directly. In a general knowledge graph, the correspondences are not always that simple. In this example, we see that classes A and B correspond to each other, even if A has as instances complex data (nodes ci) and B atomic data (Strings). More complex features of classes are required. For the example shown, this is easy to resolve, by considering the atomic data found at leaf level for an inner node, such as A.

Problem 2

Two databases might have

- **Classes** with similar meaning (e.g. Course)
- Different **Instances** for those classes (e.g. different courses at different universities)

Intension is similar, but **extension** is different

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Due to different database extensions and different naming conventions even classes that have a strong semantic similarity often have no common instances in two different databases.

Complex Features of Classes

Considering the instances (direct children) of classes is only an example of a simple class feature

Many complex features can be exploited (and have been exploited in schema matching)

- Names of attributes and relationships
- Structural relationships (data types)
- Distributional features (data values)
- Content features (e.g., text)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In database integration, many different features that can be associated with structural elements of a database have been considered for establishing semantic correspondences. For illustration, we will use in the following a simple feature: the set of atomic data values (strings) that can be reached from a node in the knowledge graph.

Content Feature of Classes

We consider as content feature, the set of terms associated with the leaf nodes that can be reached from a path starting at the instance

$T_i = \{t_1, \ldots, t_n\}$ with repeated occurrences (bag of words)

- $T_A = \{\, aberer,$
  $information\ systems, signal\ processing, vetterli, \ldots \}$
- $T_{c1} = \{\, aberer,\ information\ systems \}$
- $T_{aberer} = \{\, aberer\}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Another feature that can be exploited for this case would be the labels used along the paths. Note that the leaf nodes play a double role. They are instances, but at the same time their names serves as feature.

Finding Corresponding Classes

If $U1$ and $U2$ are the universes of DB1 and DB2 we may assume

$$U1 \cap U2 = \emptyset$$

Thus, no way to compute $P(A,B)$ directly!

Given $i \in A$, the question is whether it would be likely that considering its features also $i \in B$, even if i is not part of $U2$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Even if we have identified features that can help to spot correspondences between structurally different, but semantically equivalent elements of two databases, it might be the case that the coverage of a real-world aspect in two databases is different (e.g. courses in two different universities). Thus, directly comparing the features (e.g. the instances of a class) does not help to detect the correspondences.

Probabilistic Approach

We want to construct a function the gives the probability for a given instance i with feature T_i to belong to a class A

$P(A|T_i) = P(T_i|A)P(A)P(d) \propto P(T_i|A)P(A)$ (Bayes law)

$P(A|T_i)$ is a Naïve Bayes classifier to determine whether i belongs to A

Objective: determine $P(T_i|A)$ and $P(A)$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To tackle the problem, we will construct a model that determines (probabilistically) whether a given data instance with certain features is semantically related to a class.

Naïve Bayes Classifier

We know $P(A) = \frac{|A|}{|U_1|}$

Independence assumption: $P(T_i|A) = P(t_1|A) \dots P(t_n|A)$

With T_A being the bag of all terms occurring in all instances of A we have

$$P(t|A) = \frac{|t \in T_A|}{\sum_{t'} |t' \in T_A|}$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Computing P(A) is straightforward. We have just to determine the relative frequency of instances of A (how big A is) in the set of all possible instances.

For computing P(Ti | A) we first make an independence assumption: we assume that different terms occurring in the instance i of a class A are independent of each other. In practice this is not the case, but is a generally accepted assumption for simplicity. Then we have to compute the probability of a single term t in class A to occur.

By computing P(A) and P(Ti | A) we obtain (up to a constant) the probability of for a (new) instance I to belong to class A.

Example

Classifier for class $A = \{c1, c2, c3\}$

$T_A = \{information\ systems, aberer, signal\ processing, \ldots\}$

New instance cn: $T_{cn} = \{information\ systems, vetterli\}$

$P(vetterli|A) = \frac{1}{6}$, $P(vetterli|u) = \frac{1}{9}$

$P(T_{cn}|A) = \frac{1}{36}$, $P(T_{cn}|u) = \frac{1}{81}$

$P(A) = \frac{3}{13}$, $P(u) = \frac{4}{13}$ (13 instances total)

$P(A|T_{cn}) \propto \frac{1}{36}\frac{3}{13}$, $P(u|T_{cn}) \propto \frac{1}{81}\frac{4}{13}$

Thus, instance cn is considered to correspond more likely to A than to u

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We show here a complete example of how for a new instance cn, the most likely corresponding class can be determined in the database. The Naïve Bayes classifier assigns a higher probability for cn to belong to A than to U (which coincides with our intuition).

Computing Similarity between Classes A and B

1. Take all instances of $U1$ and train a classifier to decide whether an instance belongs to A or not
   - Training set are instances in A and $\bar{A}$
2. Select all instances in $U2$ belonging to B: $U2^B$
3. Apply the classifier trained with $U1$ to all instances in $U2^B$ to produce set $U2^{AB}$, the set of instances in $U2$ that could correspond to A
4. Do the same with roles of A and B exchanged
5. Compute $P(A,B) = \frac{|U1^{AB}|+|U2^{AB}|}{|U1|+|U2|}$
6. Compute similarly $P(A,\bar{B})$ etc
7. Then compute $sim(A,B) = \frac{P(A,B)}{(P(A,B)+P(\bar{A},B)+P(A,\bar{B}))}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For computing Jaccard similarity between two classes A and B we need to know how many elements are contained in the intersection of A and B and the union of A and B. To that end we can first separate in each of the two databases the elements that belong to the class present in the database (e.g. A in database D1 with universe U1), and then apply the classifier learnt from the other database, whether an element in each of those two sets belongs also to the other class, or not. We consider $U2^{AB}$ as the set of elements of B that are likely to belong to A, if they where part of database D1. In this way we determine (or better estimate) the sizes of the potential intersection of A and B and the union of A and B and can based on this compute an estimate for Jaccard similarity.

Example

Features of $U1\mathrm{A}$
{information systems, aberer, signal processing, vetterli, databases, ailamaki}
Features of $U1\bar{A}$
{computer science, mathematics, life science}

Train a classifier C_A to decide for a set of feauteres whether it belongs to A or not

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Example

Features of U2B
{information systems,computer graphics, databases, information retrieval}

Features of $U1\bar{B}$
{}

Features of $U1AB$
{infomration systems, databases}

Features of U$\bar{A}B$
{}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

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.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

5. DATA INDEXING AND MINING

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

5.1 Indexing for Information Retrieval

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For efficiently implementing a retrieval model in the ranking system, the statistical information about documents and the document access need to be supported by an indexing system. We will now introduce the basic methods that have been devised to that end.

Term Search

Problem: text retrieval algorithms need to find words in documents efficiently

- Boolean, probabilistic and vector space retrieval
- Given index term k_i, find document d_j

application →

B3, B17 ←

B1 A Course on Integral Equations
B2 Attractors for Semigroups and Evolution Equations
B3 Automatic Differentiation of Algorithms: Theory, Implementation, and Application
B4 Geometrical Aspects of Partial Differential Equations
B5 Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra
B6 Introduction to Hamiltonian Dynamical Systems and the N-Body Problem
B7 Knapsack Problems: Algorithms and Computer Implementations
B8 Methods of Solving Singular Systems of Ordinary Differential Equations
B9 Nonlinear Systems
B10 Ordinary Differential Equations
B11 Oscillation Theory for Neutral Differential Equations with Delay
B12 Oscillation Theory of Delay Differential Equations
B13 Pseudodifferential Operators and Nonlinear Partial Differential Equations
B14 Sinc Methods for Quadrature and Differential Equations
B15 Stability of Stochastic Differential Equations with Respect to Semi-Martingales
B16 The Boundary Integral Approach to Static and Dynamic Contact Problems
B17 The Double Mellin-Barnes Type Integrals and Their Applications to Convolution Theory

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to implement text retrieval models efficiently, efficient search for term occurrences in documents must be supported. For that purpose, different indexing techniques exist, among which inverted files are the by far most widely used.

5.1.1 Inverted Files

An inverted file is a word-oriented mechanism for indexing a text collection in order to speed up the term search task

- Addressing of documents and word positions within documents
- Most frequently used indexing technique for large text databases
- Appropriate when text collection is large and semi-static

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Inverted files support efficient addressing of words within documents. Inverted file are designed for supporting search on relatively static text collections. Therefore, their inverted files do not support continuous updates to the index when documents are updated in the corpus. Rather index construction is a one-shot process, analyzing a complete document collection. This distinguishes inverted files from typical database indexing techniques, such as B+-Trees.

Inverted Files

Inverted list l_k for a term k

$$l_k = \left[f_k : d_{i_1}, \dots, d_{i_{f_k}}\right]$$

- f_k number of documents in which k occurs
- $d_{i1},\dots,d_{ifk}$ list of document identifiers of documents containing k

Inverted File: lexicographically ordered sequence of inverted lists

$$IF = \left[i, k_i, l_{k_i}\right], i = 1, \dots, m$$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Inverted files are constructed from inverted lists. Inverted lists enumerate all occurrences of the terms in documents, by storing the document identifiers and the global frequency of occurrences. Inverted files are constructed by concatenating the inverted lists for all terms occurring in the document collection.

Storing the global frequency f_k is useful for determining inverse document frequency.

Example: Documents

B1 A Course on Integral Equations
B2 Attractors for Semigroups and Evolution Equations
B3 Automatic Differentiation of Algorithms: Theory, Implementation, and Application
B4 Geometrical Aspects of Partial Differential Equations
B5 Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra
B6 Introduction to Hamiltonian Dynamical Systems and the N-Body Problem
B7 Knapsack Problems: Algorithms and Computer Implementations
B8 Methods of Solving Singular Systems of Ordinary Differential Equations
B9 Nonlinear Systems
B10 Ordinary Differential Equations
B11 Oscillation Theory for Neutral Differential Equations with Delay
B12 Oscillation Theory of Delay Differential Equations
B13 Pseudodifferential Operators and Nonlinear Partial Differential Equations
B14 Sinc Methods for Quadrature and Differential Equations
B15 Stability of Stochastic Differential Equations with Respect to Semi-Martingales
B16 The Boundary Integral Approach to Static and Dynamic Contact Problems
B17 The Double Mellin-Barnes Type Integrals and Their Applications to Convolution Theory

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is an example of a (simple) document collection that we will use in the following as running example.

Example

1	Algorithms	3	:	3	5	7							
2	Application	2	:	3	17								
3	Delay	2	:	11	12								
4	Differential	8	:	4	8	10	11	12	13	14	15		
5	Equations	10	:	1	2	4	8	10	11	12	13	14	15
6	Implementation	2	:	3	7								
7	Integral	2	:	16	17								
8	Introduction	2	:	5	6								
9	Methods	2	:	8	14								
10	Nonlinear	2	:	9	13								
11	Ordinary	2	:	8	10								
12	Oscillation	2	:	11	12								
13	Partial	2	:	4	13								
14	Problem	2	:	6	7								
15	Systems	3	:	6	8	9							
16	Theory	4	:	3	11	12	17						

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we show the inverted lists that are obtained for our example document collection. The concatenation of those lists constitutes the inverted file.

Inverted files are a logical data structure, for which a physical storage organization needs to be designed.

The physical organization must consider the quantitative characteristics of the inverted file structure. A key observation is that the number of references to documents, corresponding to the occurrences of index terms in the documents is much larger than the number of index terms, and thus the number of inverted lists. As the number of index terms is much smaller, the index terms and the corresponding frequencies of occurrences can be kept in main memory, whereas the references to documents are kept in secondary storage. The access to the index file a data access structure is used. Examples of data access structures are binary search, hash tables or tree-based structures, such as B+-Trees or tries. The posting files consist of the sequence of all term occurrences of the inverted file. The index file is referring to the posting file by keeping for each index term a reference to the corresponding position in the posting file, at which the entries related to the index terms start. The occurrences stored in the posting file in turn refer to entries in the document file, which is also kept in secondary storage.

Heap's Law

An empirical law that describes the relation between the size of a collection and the size of its vocabulary

$$m = kn^{\beta}$$

Typical values observed: $\beta \approx 0.5, 30 < k < 100$

Parameters depend on collection type and preprocessing

- Stemming, lower case decrease vocabulary size
- Numbers, spelling errors increase

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Empirical studies show that for a document collection of size n the number of different index terms is typically O(n^β), where β is roughly 0.5 (Heap's law). More precisely, the relationship can be described as k n^β, where typical values of k are 30 < k < 100. For example, a document collection of size n= 10^6 could have approximately m=100 * 10^3=10^5 index terms.

Here we illustrate the physical organization of the inverted file for the running example. Note that only part of the data in the document file is shown.

Addressing Granularity

By default, the entries in postings file point to the document in the document file

Other granularities can be used

- Pointing to a specific position within the document
  - Example: (B1, 4) points to “Integral”
- Grouping multiple documents to one
  - Example: create groups of 4 documents, then G1 points to the group (B1,B2,B3,B4)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The addressing granularity determines of how positions of index terms are recorded in the posting file. There exist three main options:

• pointing to the start of the document in which the index term occurs (this is how we introduced inverted files)

• exact position of the index term within the document, or a sentence or paragraph containing the index term.

• grouping of multiple documents into blocks and pointing to the start of a block

The larger the granularity, the fewer entries occur in the posting file. In turn, with coarser granularity additional post-processing is required in order to determine exact positions of index terms within the documents.

Payload of Postings

Postings consist of the document identifier

- For Boolean retrieval this is sufficient

In addition, other data can be stored with the posting

- For VS retrieval term frequency can be stored
- Other data: positions of occurrence, term properties

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the simplest case the postings file consists of the document identifiers. This is sufficient for implementing a Boolean retrieval system, since the only necessary infomration for query evalaution is on the presence or absence of a term in the document. Any additional information about the term needs to be computed on the fly from the document content.

To make retrieval more efficient, additional information about the term can be stored in the postings file. For vector space retrieval it is useful to also store the term frequency in order to avoid repeated scans of the document content, whenever the term frequency is requested.

Other information about terms related to the document that can be stored in a postings file is the positions of occurrence, the context in which the term occurs (e.g. in the title) or results from some further text processing (e.g. using natural language processing such as named entity recognition about which will learn later in this lecture).

A posting indicates…

1. The frequency of a term in the vocabulary
2. The frequency of a term in a document
3. The occurrence of a term in a document
4. The list of terms occurring in a document

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When indexing a document collection using an inverted file, the main space requirement is implied by ...

1. The access structure
2. The vocabulary
3. The index file
4. The postings file

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Searching the Inverted File

Step 1: Vocabulary search

- the words present in the query are searched in the *index file*

Step 2: Retrieval of occurrences

- the lists of the occurrences of all words found are retrieved from the *posting file*

Step 3: Manipulation of occurrences

- the occurrences are processed in the *document file* to process the query

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Search in an inverted file is a straightforward process. Using the data access structure, first the index terms occurring in the query are searched in the index file. Then the occurrences can be sequentially retrieved from the postings file. Afterwards the corresponding document portions are accessed and can be processed (e.g. for counting term frequencies).

Here we illustrate the the 3 steps of the search process.

Construction of the Inverted File – Step 1

Step 1: Search phase

- The vocabulary is kept in an ordered data structure, e.g., a trie or sorted array, storing for each word a list of its occurrences
- Each word of the text is read sequentially and searched in the vocabulary
- If a word is not found in the ordered data structure, it is added to the vocabulary with an empty list of occurrences
- If a word is found, the word position is added to the end of its list of occurrences

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The index construction is performed by first constructing dynamically an order data structure (we will use a trie structure), in order to generate a sorted vocabulary and to create a list of the occurrences of index terms in the text.

Construction of the Inverted File – Step 2

Step 2: Storage phase (once the text is exhausted)

- The list of occurrences is written contiguously to the disk (posting file)
- The vocabulary is stored in lexicographical order (index file) in main memory together with a pointer for each word to its list in the posting file

Overall cost O(n)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

After the complete document collection has been traversed, the ordered data structure is sequentially traversed, and the posting file is written to secondary storage. The ordered data structure itself can be used as a data access structure for the index file that is kept in main memory.

Tries

A trie is a tree data structure to index strings

- For each prefix of each length in the set of strings a separate path is created
- Strings are looked up by following the prefix path

Example: trie for {a, to, tea, ted, ten, i, in, inn}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the following we will us tries as ordered data structure. The trie data structure is a so - called lexicographical index structure specifically suited for text processing. Variations of this data structure, specifically PAT trees can be used for efficient text processing including evaluation of regular expressions.

In this example we use word positions as addressing granularity for the postings file, since we have only a single document.

We demonstrate the initial steps of constructing the trie structure and adding to it the occurrences of index terms. The changes to the trie structure are highlighted for each step. Note that in the last step the tree structure of the trie does not change, since the index term "the" is already present.

Once the complete trie structure is constructed the inverted file can be derived from it. For doing this, the trie is traversed top-down and left-to-right. Whenever an index term is encountered it is added at the end of the inverted file. Note that if a term is prefix of another term (such as "a" is prefix of "are") index terms can occur on internal nodes of the trie. After to the construction of the inverted file the posting file can be extracted from it.

Example

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The resulting physical organization of the inverted file is shown here. The trie structure can be used as an access structure to the index file in main memory. Thus, the entries of the index files occur as leaves (or internal nodes) of the trie. Each entry has a reference to the position of the postings file that is kept in secondary storage.

Index Construction in Practice

When using a single node not all index information can be kept in main memory → Index merging

- When no more memory is available, a partial index I_i is written to disk
- The main memory is erased before continuing with the rest of the text
- Once the text is exhausted, a number of partial indices I_i exist on disk
- The partial indices are merged to obtain the final index

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In practice, the available main memory needed to keep the inverted file during index construction is insufficient in size, since the storage space consumed by the postings is proportional to the size of the document collection.

In this case, the index construction process needs be partitioned into smaller steps: in a first phase, the document collection is sequentially traversed, and partial indices are written to the disk whenever the main memory becomes full. This results in a set of partial indices, indexing consecutive partitions of the text. In a second phase the partial indices need to be merged into one index.

Index Merging

BSBI: Blocked sort-based Indexing

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This figure illustrates the merging process: 8 partial indices have been constructed. Step by step the indices are merged, by merging two indices into one, until one final index remains. The merging can be performed, such that the two partial indices which are to be merged are in parallel scanned on the disk, and while scanning the resulting index is written sequentially to the disk.

Merging the indices requires first merging the vocabularies. As we mentioned earlier, the vocabularies are comparably small and thus the merging of the vocabularies can take place in main memory. In case a vocabulary term occurs in both partial indices, their list of occurrences from the posting file need to be combined. Here we can take advantage of the fact that the partial indices have been constructed by sequentially traversing the document file. Therefore, these lists can be directly concatenated without sorting.

The computational complexity of the merging algorithm is $O(n \log_2(n/M))$. This implies that the additional cost of merging as compared to the purely main-memory based construction of inverted files is a factor of $O(\log_2(n/M)))$. This is small in practice, e.g., if the database size n is 64 times larger than the main memory size, then this factor would be 6.

This example illustrates how the merging process can be performed when the database is split into two partitions.

Addressing Granularity

Documents can be addressed at different granularities

- coarser: text blocks spanning multiple documents
- finer: paragraph, sentence, word level

General rule

- the finer the granularity the less post-processing but the larger the index

Example: index size in % of document collection size

Index	Small collection (1Mb)	Medium collection (200Mb)	Large collection (2Gb)
Addressing words	73%	64%	63%
Addressing documents	26%	32%	47%
Addressing 256K blocks	25%	2.4%	0.7%

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The posting file has the by far largest space requirements. An important factor determining the size of an inverted file is the addressing granularity used.

Experiments illustrate the substantial gains that can be obtained with coarser addressing granularities. Coarser granularities lead to a reduction of the index size for two reasons:

- a reduction in pointer size (e.g., from 4 Bytes for word addressing to 1 Byte with block addressing)
- and a lower number of occurrences.

Note that in the example for a 2GB document collection with 256K block addressing the index size is reduced by a factor of almost 100.

Index Compression

Documents are ordered and each document identifier d_{ij} is replaced by the difference to the preceding document identifier

- Document identifiers are encoded using fewer bits for smaller, common numbers

$$l_k = \left\langle f_k : d_{i_1}, \ldots, d_{i_{fk}} \right\rangle \rightarrow$$

$$l_k' = \left\langle f_k : d_{i_1}, d_{i_2} - d_{i_1}, \ldots, d_{i_{fk}} - d_{i_{fk}-1} \right\rangle$$

- Use of varying length compression further reduces space requirement
- In practice index is reduced to 10- 15% of database size

X	code(X)
1	0
2	10 0
3	10 1
4	110 00
5	110 01
6	110 10
7	110 11
8	1110 000
63	111110 11111

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A further reduction of the index size can be achieved by applying compression techniques to the inverted lists. In practice, the inverted list of a single term can be rather long. A first improvement is achieved by storing only differences among subsequent document identifiers. Since they occur in sequential order, the differences are much smaller integers than the absolute position identifiers.

In addition, number encoding techniques can be applied to the resulting integer values. Since small values will be more frequent than large ones this leads to a further reduction in the size of the posting file.

Using a trie in index construction …

1. Helps to quickly find words that have been seen before
2. Helps to quickly decide whether a word has not seen before
3. Helps to maintain the lexicographic order of words seen in the documents
4. All of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

5.1.2 Web-Scale Indexing: Map-Reduce

Pioneered by Google: 20PB of data per day (2008)

- Scan 100 TB on 1 node @ 50 MB/s = 23 days
- Scan on 1000-node cluster = 33 minutes

Cost-efficiency

- Commodity nodes, network (cheap, but unreliable)
- Automatic fault-tolerance (fewer admins)
- Easy to use (fewer programmers)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For Web-scale document collections traditional methods of index construction are not feasible. Therefore Google developed new approaches in terms of infrastructure and computing model to index very large document collections. A key element is the map-reduce programming model. It allows to parallelize index construction, within an infrastructure using potentially unreliable commodity hardware. The map-reduce programming model has been key in the ability of Google and later other Web platforms to scale up their applications. It actually led to a novel distributed programming paradigm and systems approach, that is tuned towards cost-efficiency and simplicity of programming.

Map-Reduce Programming Model

Data type: key-value pairs (k, v)

Map function: $(k_{in}, v_{in}) \rightarrow [(k_{inter}, v_{inter})]$

Analyses some input, and produces a list of results

Reduce function: $(k_{inter}, [v_{inter}]) \rightarrow [(k_{out}, v_{out})]$

Takes all results belonging to one key, and computes aggregates

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The map-reduce programming model is based on manipulating key-value pairs (k, v) and lists of key value pairs $[(k, v)]$. It is based on two functions.

The map function receives some input data (typically a piece of text to analyze or index) and produces a list of key-value pairs. They constitute a partial results of the analysis (e.g., the counts of words in the text).

The reduce function receives as input all results for a given key value, that have been computed by different mapper functions. It computes then an aggregate output value (e.g., the total count of words in the document corpus).

Example

Basic word counter program

```
def mapper(document, line):
for word in line.split(): output(word, 1)

def reducer(key, values):
output(key, sum(values))
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is a basic illustration of the use of the map-reduce programming paradigm for a word counter program. A mapper processes the text by splitting it into words and producing for each word a key value pair (word, 1).

The reducer receives a list of all values associated with a key (a word), and outputs the aggregate value, the word count.

This is the code that a programmer has to write. The map-reduce system is then in charge of distributing the data, executing these functions in a distributed infrastructure and maling sure that the key-value pairs produced by the map function are properly aggregated and passed on to the reduce function.

This figure illustrates the basic steps of a map-reduce computation for the basic example of word counting that are realized by the map-reduce infrastructure.

The system decides of how the document collection is partitioned and assigned to different mapper nodes. The mapper nodes extract word statistics for their partition of the document collection. Note that in this illustration the mapper emits only one key-value pair for each word and document, i.e., counts the total number of the occurrences of the word in the document (this is slightly smarter implementation than in the previous slide).

For each word, a reducer node is responsible. Based on the key, i.e., the word, the mapper nodes send their local results for the word to the responsible reducer node. This can be controlled, e.g., by hashing the key values and assigning hash values to reducer nodes. The reducer nodes aggregate the statistics that they receive from all the mapper nodes. When the reducer nodes have completed generating the partial indices for their key space, the results are written to the file system. The allocation of resources for the processes for mappers and reducers is performed automatically by the system and completely transparent to the developer of the code.

An important observation is that under this model, reducer nodes can start the aggregation only after all mapper nodes have finished, in order to assure that they receive all data. This can constitute an important bottleneck in such a system, since reducers can only start after the slowest mapper has terminated.

Here we show some additional functions that are part of the map-reduce model. Note that the computation is different from the previous example. Mapper nodes already aggregate the statistics for a single document and the final output of the reducer is the maximum count instead of the sum.

A combiner function can locally aggregate results on a node executing the mapper function (e.g., aggregating all counts of the same word), thus reducing the number of intermediate results.

Partitioners allow to control the strategy of distributing keys to reducer (to override the default strategy). This can be relevant if the programmer knows about the potential load distribution for keys, and can thus optimize the allocation of reducer resources.

What the Programmer Controls (and not)

The programmer controls

- Key-value data structures (can be complex)
- Maintenance of state in mappers and reducers
- Sort order of intermediate key-value pairs
- Partitioning scheme on the key space

The map-reduce platform controls

- where the mappers and reducers run
- when a mapper and reducer starts and terminates
- which input data is assigned to a specific mapper
- which intermediate key-value pairs are processed by a specific reducer

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The map-reduce model has been a successful approach to alleviate from the developers the tasks for dealing with the management of the distributed computing resources and concentrating on the logics of the data processing task.

Inverted File Construction Using Map-Reduce

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We have illustrated the use of map-reduce so far performing for simple statistics tasks on text. The same programming paradigm can be used to perfrom the construction of an inverted file index. In this case, instead of producing simple counts, mappers produce inverted lists for the document inputs they receive. These inverted lists are then forwarded to the reducer in charge of the specific term. The reducers then aggregate the inverted files for a given term into a common inverted list. Finally, the full lists of postings can be stored.

Note that in this illustration the addressing granularity is at document level, i.e. words are associated with documents. The mappers compute the total frequency of the term in the given document.

Inverted File Construction Program

```
def mapper(document, text):
            f = {}
            for word in text.split(): f[word] += 1
            for word in f.keys():
                  output(word, (document, f[word]))

def reducer(key, postings):
            p = []
            for d, f in postings: p.append((d, f))
            p.sort()
            output(key, p)
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is of how the inverted file construction illustrated on the previous slide could be implemented in the map-reduce model. Note that in the reducer the set of postings needs to be sorted, as they can be received from the mappers in arbitrary order.

Other Applications of Map-Reduce

Framework is used in many other tasks, particular for text and Web data processing

- Graph processing (e.g., PageRank)
- Processing relational joins
- Learning probabilistic models

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Though the initial motivation for developing map-reduce was the construction of Web-scale index structures, it turned out that the same model can be used for many other data processing tasks that allow for a high degree of parallel processing. These are called embarrassingly parallel workloads, where the input can be split with little effort into many partitions that can be processed in parallel.

Practical Computation of PageRank

Iterative computation

$$\vec{p}_0 \leftarrow \vec{s}$$
$$while\, \delta > \varepsilon$$
$$\vec{p}_{i+1} \leftarrow qR \bullet \vec{p}_i$$
$$\vec{p}_{i+1} \leftarrow \vec{p}_{i+1} + \frac{(1-q)}{N}\vec{e}$$
$$\delta \leftarrow \left\| \vec{p}_{i+1} - \vec{p}_i \right\|_1$$

ε termination criterion

s arbitrary start vector, e.g., $s = \frac{\vec{e}}{N}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For the practical computation of the PageRank ranking an iterative approach can be used. The vector e is used to add a source of rank. It can uniformly distribute weights to all pages, but it could also incorporate pre-existing knowledge on the importance of pages and bias the ranking towards them. The vector can also be used as initial probability distribution.

Implementation in Map-Reduce

Iterative computation can be viewed as passing messages among nodes

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For practical computation of a ranking for a Web graph, the link matrices are huge and can potentially not be processed on a single node. This problem is similar to the one we have addressed for the construction of inverted files and for which one solution was the use of the map-reduce computing paradigm. We can use this paradigm as well for computing PageRank in a distributed way.

To that end, we consider the iterative computation of the PageRank values as a process of transmitting messages among the nodes. In the computation of the PageRank values a node receives in each step from the incoming links weights. It then aggregates those weights, before the next round of computation is performed. Representing the computation in this form leads directly to an approach for implementing it in the map-reduce programming model.

Map-Reduce Implementation

Node objects: N(nid, PageRank, neighbors)

- Distributed over Mapper Nodes

```
def mapper(n, N):
        p = N[PageRank]/
                len(N[neighbors])
        for m in N[neighbors]:
                output(m, p)
        output(n, N)
```

```
def reducer(m, messages):
        M = None; s = 0
        for p in messages:
                if is_node(p):
                        M = p
                else
                        s += p
        M[PageRank] = s
        output(m, M)
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the map-reduce implementation two types of outputs are processed. Node objects that contain a node identifier nid, the current PageRank values, and a list of the neighbour ids. The mapper receives a node object N. It computes the contribution of its rank to its neighbours and outputs the neighbour node ids together with their weights. It also outputs the nodeid together with the node object.

The reducers collect all messages concerning a given node m. It receives both the node object and all weights. The weights are added up and used to update the node object. The reducer outputs the updated node object that will be serving as inputs to mappers in a subsequent map-reduce phase. Therefore, each execution of a mapreduce job corresponds to one iteration of the PageRank algorithm.

Maintaining the order of document identifiers for vocabulary construction when partitioning the document collection is important ...

1. in the index merging approach for single node machines
2. in the map-reduce approach for parallel clusters
3. in both
4. in neither of the two

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

5.1.3 Link Indexing

Connectivity Server: support for fast queries on the web graph

- Which URLs point to a given URL?
- Which URLs does a given URL point to?

Stores mappings in memory from

- URL to outlinks, URL to inlinks

Applications

- Link analysis (PageRank, HITS)
- Web graph analysis
- Web crawl control: crawl optimization

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For text retrieval we introduced inverted files as an indexing structure for fast lookup of documents based on text queries. Using such an index it becomes possible to efficiently compute that statistics needed for the ranking model.

Similarly as for link-based ranking, for computing statistics on the Web graph we need an efficient indexing structure for the link graph. This is called a connectivity server. It allows to answer efficiently the queries relevant for Web graph analysis, namely which URLs a page points to and is pointed to. Beyond performing Web graph analysis such a connectivity server has also other applications, especially for controlling Web crawling.

Sec. 20.4

Adjacency Lists

The set of URLs a node is pointing to (or pointed to) sorted in *lexicographical order*

Example: outgoing links from www.epfl.ch

actu.epfl.ch/feeds/rss/mediacom/en/
bachelor.epfl.ch/studies
futuretudiant.epfl.ch/en
futuretudiant.epfl.ch/mobility
master.epfl.ch/page-94489-en.html
phd.epfl.ch/home
www.epfl.ch/navigate.en.shtml

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A connectivity server has to store all outgoing (and incoming) links to a web page. For example, the home page of EPFL contains a large set of outgoing links, some of which are shown here. As a first step, the lists of links are sorted in lexicographical order. As a result, we obtain the adjacency list for a Web page, which we can consider as the equivalent to the posting list of a document in text indexing.

Representation of Adjacency Lists

Assume each URL represented by an integer

- For a 50 billion page web, we need 36 bits per node
- Naively, this demands 72 bits to represent each hyperlink (source and destination node); on average 10 links per page
- For the current Web: 4.5 TB
- Can we do better (for main memory storage)?

Node	Outdegree	Successors
...	...	...
15	11	13, 15, 16, 17, 18, 19, 23, 24, 203, 315, 1034
16	10	15, 16, 17, 22, 23, 24, 315, 316, 317, 3041
17	0	
18	5	13, 15, 16, 17, 50
...	...	...

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

As a first optimization of the representation of adjacency lists, we represent each URL by an integer, instead of storing it in its textual form. Using such an approach we can estimate the total size of adjacency lists for the current Web:

- The current (crawled) Web has around 50 billion pages (http://www.worldwidewebsize.com/, 2022)
- It is estimated that a page contains on average 10 links
- We need 32 bits for each URL, which demands 64 bits for the storage of a single link.

Therefore, the required storage is 4 TB.

Even with large memory sizes available today, this still is a significant index size. In the following, we will show how to reduce required storage to approximately ~3 bits/link which makes the size of the index much more manageable, i.e., about 20 times less storage. This will be achieved by systematically compressing the adjacency lists.

Properties of Adjacency Lists

Locality (within lists)

- Most links contained in a page are navigational, thus their indices are close in lexicographical order
  - e.g., www.epfl.ch contains the links futuretudiant.epfl.ch/en and futuretudiant.epfl.ch/mobility

Similarity (between lists)

- Observation 1: Either two lists have almost nothing in common, or they share large numbers of links
- Observation 2: Pages that occur close to each other in lexicographic order tend to have similar lists
  - e.g., futuretudiant.epfl.ch/en and futuretudiant.epfl.ch/mobility share many links

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For compressing adjacency lists we can exploit several observations on typical properties of Web pages.

Locality: Most links contained in a page are for navigating withing the same Web site. If we compare the source and target URLs of these links, we observe that as a result they often share a long common prefix. Thus, if URLs are sorted lexicographically, the index of the URL of a Web page and the URLs of the targets of the links of the Web page are close to each other. Locality is a property of one adjacency list, thus is an intra-list similarity property.

Similarity: In general, we can assume that either pages have many common links, because the belong to the same web site, or they have almost nothing in common, because they are from different Web sites. Furthermore, pages that are from the same Website will have URLs that are similar in lexicographical order, and therefore it is more likely to find pages with many common outgoing links close to each other in lexicographic order. Similarity is a property of different adjacency lists, this an inter-list similarity property.

We will now show of how to exploit these two properties.

Exploiting Locality

Use Gap Encoding (as in inverted files)

- For node x, $S(x) = (s_1, \dots, s_k)$ will be represented as
  $$(s_1(x), s_2 - s_1(x) - 1, \dots, s_k - s_{k-1} - 1)$$
- To avoid that the first entry is negative, the first entry is
  $$s_1(x) = \begin{cases} 2(s_1 - x), & if \;\; s_1 - x \geq 0 \\ 2|s_1 - x| - 1, & if \;\; s_1 - x < 0 \end{cases}$$
- Use of varying length encoding

Use Gap Encoding (as in inverted files)
- For node x, $S(x) = (s_1, \dots, s_k)$ will be represented as
  $(s_1(x), s_2 - s_1(x) - 1, \dots, s_k - s_{k-1} - 1)$
- To avoid that the first entry is negative, the first entry is
  $s_1(x) = \begin{cases} 2(s_1 - x), & if \;\; s_1 - x \geq 0 \\ 2|s_1 - x| - 1, & if \;\; s_1 - x < 0 \end{cases}$
- Use of varying length encoding

Node	Outdegree	Successors
...	...	...
15	11	3, 1, 0, 0, 0, 0, 3, 0, 178, 111, 718
16	10	1, 0, 0, 4, 0, 0, 290, 0, 0, 2723
17	0	
18	5	9, 1, 0, 0, 32
...	...	...

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Locality can be exploited in a way analogous of how compression of posting lists for text indexing has been performed. Instead of storing the absolute integer indices of the URL identifiers, their differences are stored. In other words, we perform gap encoding. The resulting differences are then encoded using a varying length compression scheme, such as gamma coding, as it has been applied with inverted files.

Exploiting Similarity

Copy data from similar lists (exploit observation 1)

- Reference list: reference to another list
  - Searched in a neighboring window of nodes (exploit observation 2)
- Copy list: bitmap indicates nodes copied from reference list
- Extra nodes: additional nodes not in reference list

Node	Outdegree	Successors
...	...	...
15	11	3, 1, 0, 0, 0, 0, 3, 0, 178, 111, 718
16	10	1, 0, 0, 4, 0, 0, 290, 0, 0, 2723
17	0	
18	5	9, 1, 0, 0, 32
...	...	...

Node	Outd.	Ref.	Copy list	Extra nodes
...	...	...	...	...
15	11	0		13, 15, 16, 17, 18, 19, 23, 24, 203, 315, 1034
16	10	1	01110011010	22, 316, 317, 3041
17	0			
18	5	3	11110000000	50
...	...	...	...	...

Result: about 3 bits / link (with some further compression)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For exploiting similarity, we exploit the redundancy among similar lists. Based on the observation that similar lists are more likely to occur in a lexicographical neighborhood, in a first step a sliding window of neighboring lists that have already been processed is searched for a most similar list. If such a list is found, it is called reference list. Then for the given adjacency list to be compressed, only the data necessary to reconstruct the list from the reference list will be stored. For this it is necessary to store two types of data:

1. Copy data: this is a bitlist that indicates for every entry in the reference list whether the URL is also part of the given adjacency list. This covers all URLs that the given list can “inherit” from the reference list
2. Extra nodes: the given list can also contain URLs that are not part of the reference list. For those the indices of the URLs need to be stored explicitly.

In the example we use Node 15 for the reference list and compress Node 16 and 18. By comparing the lists we see that for the case of Node 18 all, but one URL appear in the reference list of Node 15. There are indicated in the bitlist. The adjacency list of Node 18 contains also one URLs that does not appear in the reference list, namely 50. This is listed in the list of extra nodes.

Candidates for potential reference lists are searched among neighboring lists using a window of predefined size. The choice of the window size is important, as larger windows increase chances of finding good candidates, but also increase the cost of compression.

Together with some further compression applied to the copy lists and the extra nodes, this index compression scheme achieves about 3 bits/link cost in the representation of the Web graph.

When compressing the adjacency list of a given URL, a reference list

1. Is chosen from neighboring URLs that can be reached in a small number of hops
2. May contain URLs not occurring in the adjacency list of the given URL
3. Lists all URLs not contained in the adjacency list of given URL
4. All of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Which is true?

1. Exploiting locality with gap encoding may increase the size of an adjacency list
2. Exploiting similarity with reference lists may increase the size of an adjacency list
3. Both of the above is true
4. None of the above is true

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

5.1.4 Distributed Retrieval

Centralized retrieval

- Aggregate the weights for ALL documents by scanning the posting lists of the query terms
- Scanning is relatively efficient
- Computationally quite expensive (memory, processing)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

When using inverted files, a query involving multiple search terms requires the retrieval and scanning of the postings lists of all terms. In a centralized server this can be implemented relatively efficiently, though still resource-intensive, since scanning of disks is a comparably efficient operation.

Distributed Retrieval

Distributed retrieval

- Posting lists for different terms stored on different nodes
- The transfer of complete posting lists can become prohibitively expensive in terms of bandwidth consumption

Is it necessary to transfer the complete posting list to identify the top-k documents?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In a distributed setting the picture changes quite significantly. Assuming that posting lists for different terms are stored on different nodes, complete posting lists need to be transferred over the network. For frequent terms,

these postings lists can contain very large numbers of entries, and significant amounts of data need to be transferred over network in order to compute the query result, which results in a prohibitively high network bandwidth consumption. So, the question is, whether there exist more efficient ways to determine the top ranked (top-k) elements of the result of a query, without the need to inspect complete posting lists.

Remark: in the following we will use k to indicate the number of results retrieved, even though we have used earlier k to denote the size of the vocabulary. The terminology top-k is so well established, that it would be confusing to deviate from it.

Fagin's Algorithm

Entries in posting lists are sorted according to the tf-idf weights

- Scan in parallel all lists in round-robin till k documents are detected that occur in all lists
- Lookup the missing weights for documents that have not been seen in all lists
- Select the top-k elements

Algorithm provably returns the top-k documents for monotone aggregation functions!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Fagin's algorithm is an approach for efficient distributed retrieval. It has been originally devised for multimedia queries, where multiple features of an object (e.g., an image) need to be combined to determine the most similar ones. The algorithm tries to minimize the number of objects (in our case documents) that need to be inspected in that process.

An important assumption that is made in Fagin's algorithm, is that the elements in a posting list are ordered according to the scores of the documents and not by document identifiers. In the context of text retrieval we would use tf-idf weights as the scores. Note that this assumption implies that an additional one-time cost occurs for sorting the posting lists. The algorithm proceeds as follows:

Phase 1: The algorithm scans in a round-robin fashion the elements of the posting lists starting from those with the highest score. Whenever an element is encountered in multiple lists, their scores are combined (e.g., added). This processing is continued till k elements are detected that appear in all lists.

Phase 2: By then, many other documents also may have been detected, that do not occur in all lists. Therefore, in a next step the missing scores are retrieved from the lists. This requires random access, supported by an index. This constitutes the most expensive part of the algorithm.

Phase 3: Finally, the k elements with the highest scores are returned. These are not necessarily corresponding to those elements that have been identified in the Phase 1. They also might include elements for which additional scores have been retrieved in Phase 2.

The algorithm returns provably always the k elements with the highest combined score.

Example 1

Finding the top-2 elements for a two-term query

d1	0.9
d4	0.82
d3	0.8
d5	0.65
.....	
d6	0.51
d2	0.1
d7	0.0

d6	0.81
d2	0.7
d5	0.66
d1	0.45
.....	
d3	0.33
d7	0.15
d4	0.0

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The example illustrates a case where two lists are searched, i.e., processing a query with two terms. First 6 new different documents are detected in phase 1 and their scores are recorded.

Example 2

Finding the top-2 elements for a two-term query

d1	0.9
d4	0.82
d3	0.8
d5	0.65
.....	
d6	0.51
d2	0.1
d7	0.0

d6	0.81
d2	0.7
d5	0.66
d1	0.45
.....	
d3	0.33
d7	0.15
d4	0.0

d1	0.9	0.45	1.35
d6		0.81	0.81
d4	0.82		0.82
d2		0.7	0.7
d3	0.8		0.8
d5	0.65	0.66	1.34

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the next step we are detecting two documents, d1 and d5, that are occurring in both posting lists. Thus, we finish phase 1 of the algorithm, as we are now sure that the top-2 elements will be found in the documents detected so far.

Example 3

Finding the top-2 elements for a two-term query

d1	0.9
d4	0.82
d3	0.8
d5	0.65
.....	
d6	0.51
d2	0.1
d7	0.0

d6	0.81
d2	0.7
d5	0.66
d1	0.45
.....	
d3	0.33
d7	0.15
d4	0.0

d1	0.9	0.45	1.35
d6	0.51	0.81	1.32
d4	0.82	0.0	0.82
d2	0.1	0.7	0.8
d3	0.8	0.33	1.13
d5	0.65	0.66	1.31

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In phase 2, the missing scores of the other documents are retrieved using random access. Once they have been obtained, the top 2 documents are returned. In this example these are documents d1 and d6. Note that these are not the 2 documents that have been first discovered to occur in both lists, which where d1 and d5.

Discussion

Complexity

- $O((k\ n)^{1/2})$ entries are read in each list for n documents
- Assuming that entries are uncorrelated
- Improves if they are positively correlated

In distributed settings optimizations to reduce the number of roundtrips

- Send a longer prefix of one list to the other node

Fagin's algorithm may behave poorly in practical cases

- Alternative algorithm: Threshold Algorithm

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

It can be shown that the complexity of the Fagin algorithm in the case of two lists is $O((k\ n)^{1/2})$ for the number of entries that are read from each list, where n is the number of documents in the document collection. This is significantly smaller than reading the complete lists and reduces further if the entries are positively correlated (i.e., if a document is highly ranked in one list, then it has also higher probability to be highly ranked in the other list), which is likely to be the case. The results generalizes to the case of multiple lists.

In a distributed setting applying Fagin's algorithm directly is still not very practical, since for every element retrieved from a list a message would have to be exchanged with another node. To avoid this, variants of this algorithm have been proposed, where larger chunks of the list from one node are sent to the other. In the ideal case one node "guesses" how many entries from its list would have to be read and transmits this set of entries to the other node(s).

Threshold Algorithm

Threshold Algorithm

- Access sequentially elements in each list
- At each round
  - lookup missing weights of current elements in other lists using random access
  - Keep the top k elements seen so far
  - Compute threshold as aggregate value of the different elements seen in current round
- If at least k documents have aggregate value higher than threshold, halt

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The threshold algorithm is an alternative to Fagin's algorithm. It processes also elements in the lists in a round robin fashion.

At each round it computes a threshold, as the aggregate value of the weights of the different elements accessed in the current round. Since the lists are sorted, the threshold value will continuously decrease. For the elements considered in the current round, the missing values from the other lists are obtained using random access, to obtain the aggregate score of those elements. Then the k elements with the highest scores are retained.

The algorithm stops once all retained elements have higher score than the current threshold value.

The algorithm returns provably always the k elements with the highest combined score in case of monotone aggregation functions.

Example

Finding the top-2 elements for a two-term query

d1	0.9
d4	0.82
d3	0.8
d5	0.65
.....	
d6	0.51
d2	0.1
d7	0.0

d6	0.81
d2	0.7
d5	0.66
d1	0.45
.....	
d3	0.33
d7	0.15
d4	0.0

Threshold

1.71

d1	0.9	0.45	**1.35**
d6	0.81	0.51	**1.32**

d1 and d6 have aggregate weights lower than the threshold, therefore continue

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We execute the threshold algorithm for the same example, as we did for Fagin's Algorithm.

Example

Threshold

d1	0.9
d4	0.82
d3	0.8
d5	0.65
.....	
d6	0.51
d2	0.1
d7	0.0

d6	0.81
d2	0.7
d5	0.66
d1	0.45
.....	
d3	0.33
d7	0.15
d4	0.0

1.71
1.52

d1	0.9	0.45	**1.35**
d6	0.81	0.51	**1.32**

The documents d4, d2 have lower aggregate weights and are therefore dismissed

d1 and d6 have aggregate weights lower than the threshold, therefore continue

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Note that the threshold is always the aggregate value of the scores of the currently considered elements, in this case d4 and d2. In this round the weight of d4 is 0.82 and the wieght of d2 is 0.8. Both are lower than the weight of the current top-2 elements, d1 and d6. Therefore, d4 and d2 are dismissed.

Example

Threshold

d1	0.9
d4	0.82
d3	0.8
d5	0.65
.....	
d6	0.51
d2	0.1
d7	0.0

d6	0.81
d2	0.7
d5	0.66
d1	0.45
.....	
d3	0.33
d7	0.15
d4	0.0

1.71
1.52
1.46

d1	0.9	0.45	**1.35**
d6	0.81	0.51	**1.32**

The documents d3, d5 have lower aggregate weights and are therefore dismissed

d1 and d6 have aggregate weights lower than the threshold, therefore continue

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Also d3 and d5 can be dismissed in this round. Since the aggregate weights of the current top-2 elements are lower than the threshold, the algorithm continues.

Example

d1	0.9
d4	0.82
d3	0.8
d5	0.65
.....	
d6	0.51
d2	0.1
d7	0.0

d6	0.81
d2	0.7
d5	0.66
d1	0.45
.....	
d3	0.33
d7	0.15
d4	0.0

Threshold

1.71
1.52
1.46
1.1

d1	0.9	0.45	**1.35**
d6	0.81	0.51	**1.32**

The document d5 has lower aggregate weights and is therefore dismissed

In this round d1 and d6 have aggregate weights higher than the threshold and the algorithm terminates

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Finally the threshold has dropped below the weights of d1 and d6, and so the algorithm stops.

Discussion

TA in general performs fewer rounds the FA

- Therefore, fewer document accesses
- But more random accesses

TA is also provably correct for monotone aggregation functions

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The threshold algorithm terminates faster than FA in general, at the expense of performing more random accesses.

Applications

Beyond distributed document retrieval these algorithms have wider applications

- Multimedia, image retrieval
- Top-k processing in relational databases
- Document filtering
- Sensor data processing

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Fagin's algorithm has found many applications apart from distributed retrieval. It is being used in multimedia retrieval (it's original application), but also in processing data from relational databases (e.g., finding tuples with a highest combined value for multiple attributes), or sensor data processing.

When applying Fagin's algorithm for a query with three different terms for finding the k top documents, the algorithm will scan ...

1. 2 different lists
2. 3 different lists
3. k different lists
4. it depends how many rounds are taken

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With Fagin's algorithm, once k documents have been identified that occur in all of the lists ...

1. These are the top-k documents
2. The top-k documents are among the documents seen so far
3. The search has to continue in round-robin till the top-k documents are identified
4. Other documents have to be searched to complete the top-k list

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

References

Lin, J., Dyer, C. (2010). Inverted Indexing for Text Retrieval. In: Data-Intensive Text Processing with MapReduce. Synthesis Lectures on Human Language Technologies. Springer, Cham.

Ronald Fagin, Amnon Lotem, Moni Naor. Optimal aggregation algorithms for middleware, PODS 2001.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

References

Course material based on

- Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008 (http://www-nlp.stanford.edu/IR-book/)

Relevant articles

- Boldi, Paolo, and Sebastiano Vigna. "The webgraph framework I: compression techniques." Proceedings of the 13th international conference on World Wide Web. ACM, 2004.
- Jimmy Lin and Chris Dyer. Data-Intensive Text Processing with MapReduce, Morgan & Claypool Synthesis, 2011.
- Lin, J., Dyer, C. (2010). Inverted Indexing for Text Retrieval. In: Data-Intensive Text Processing with MapReduce. Synthesis Lectures on Human Language Technologies. Springer, Cham.
- Ronald Fagin, Amnon Lotem, Moni Naor. Optimal aggregation algorithms for middleware, PODS 2001.

5.2 Data Mining

Data is gathered at an increasing speed and volume (size doubles each year – Big Data)

- 4 new petabytes per day (2021)
- 500 hours of video uploaded every minute (2020)
- 1 billion tweets each 48 hours (2022)
  1st billion took 3 years)
- 12 EB (12 billion GB) of storage in Utah

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The growing use of information technology in many domains produces rapidly growing amounts of data in business, science and on the Web. Recently the rate at which digital data is produced started exceeding the amount of storage that can be provided.

These data collections potentially contain a wealth of hidden information, that needs to be discovered. Businesses can learn from their transaction data more about the behavior of their customers and therefore can become more efficient by exploiting this knowledge. Science can obtain from observational data (e.g., satellite data, sensor data) new insights on scientific problems. Web usage data can be analyzed and used to optimize information access, but as well to implement novel business models, such as for advertising on the Web.

The task of extracting useful information from large datasets is called **data mining** and has in the recent years become one of the most dynamically evolving areas in computer science in general.

Data Mining

Most of this data is useless

- pictures of drunk people
- videos of cats
- royal baby, Justin Bieber
- SMS to girlfriend

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Having increasing amounts of data does not necessarily imply having more useful information.

Data Mining Tasks

But some are interesting

- political orientation of people
- acts of violence
- opinions on products
- SMS by a president

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

However, some of the information could be very interesting.

Data Mining Challenge

Transform masses of data into actionable intelligence

- From transaction data to market insights
- From observational data (satellite, sensors) to new scientific hypothesis
- From web usage data to better user interfaces
- ...

Analysis of large data sets to find relationships and to summarize the data in novel ways that are both <u>understandable</u> and <u>useful</u>

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The data mining challenging is to extract from large amounts of data, the information that is relevant for performing a given task. That is called actionable intelligence (but has many other names as well, such as business intelligence, market intelligence, open source intelligence, technology intelligence etc). For information to be useful it has to satisfy the following two key criteria:

- Understandable: a human should be able to interpret the insights to take decisions
- Useful: the insights should help to take decision that have practical impact and utility; this also implies that insights that are not « surprising » but just reproduce existing knowledge are also not very useful.

Tackling the Data Mining Challenge

Practical Questions

- Data Access (ownership!)
- Domain knowledge (expertise!)

Technical Questions

- Data mining: algorithmic approaches to produce insights
- (Big) Data management (store, index, retrieve)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The challenges to obtain insights with data mining are numerous. The following are some of the key challenges:

1. The data: often massive amounts of data exist, but are often not easily accessible, protected for legal, organizational, economic or political reasons. Also the data can be distributed over many different systems that are not very accessible.
2. The questions: searching for insights into data requires to have at least a general idea of what we are looking for, or what could be an interesting and useful insight. Without understanding the objective, it is hard to find useful answers.
3. The algorithms: Extracting interesting information out of data, requires efficient and smart algorithms. This is what we will study in the following. Once we understand and master such algorithms, the real challenges will be point 1 and 2.
4. Systems for handling Big Data: this is an area that has made huge progress in the recent years, in particular due to the needs of the big Internet companies. Many of the tools are now available as open source and within the cloud.

A classical example of a data mining problem is ”shopping basket analysis". Retail stores gather information on what items are purchased by their customers. The expectation is, by finding out what products are frequently purchased together, identifying ways to optimize the sales of the products by better targeting certain groups of customers (e.g., by planning the layout of a store or planning advertisements). A well-known example was the (presumable) discovery that people who buy diapers also frequently buy beers (probably exhausted fathers of small children). Therefore, nowadays one finds frequently beer close to diapers in supermarkets, and of course also chips close to beer. Similarly, amazon exploits this type of associations in order to propose to their customers books that are likely to match their interests. Searching for methods for address this type of problem was resulting one of the best-known data mining techniques: association rule mining.

Other Uses for Association Rules

Amazon

Analysis of Query Logs (Query Expansion)

– Users that search for “Obama President” often also search for “Obama President Elections”

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The original term of “shopping basket analysis” does not imply that the resulting methods are limited to analyzing shopping behaviors. The same techniques can and have been applied in many other contexts, including recommender systems, search systems, text analysis, query log analysis etc.

Classes of Data Mining Problems

Local properties

- Patterns that apply to part of the data
  - e.g., buy diapers ➔ buy beers

Global model

- Descriptive **structure** of the data
  - e.g., 3 types of customer behaviour
- Predictive **function** of the data
  - e.g., dist(beer, diapers)=1 ➔ +10% beer sales

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Association rule mining is just one example of a data mining algorithm. Data mining algorithms can in general by classified according the type of models they provide.

A first distinction is made among data mining algorithm that identify global structures of the data, either in the form of summaries of the data or as globally applicable rules, and data mining algorithms that provide models that apply only locally, i.e., to some subset of the data, as patterns or outliers found in the data. The example of association rule mining is a typical case of discovering local patterns.

Data mining algorithms for finding global models of the data can be further distinguished into approaches that are used to characterize the data by identifying some global structure, and into techniques that allow to make predictions on data that has not yet been seen.

Data Mining Overview

Local properties		Global model	
Patterns		Descriptive model	Predictive model
Association rules	Pattern mining	Clustering	Classification
		Information retrieval	Regression
Exploratory data analysis			
Interactive tools		Visualisation	

unsupervised supervised

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This graphic shows a basic categorization of different data mining methods. In this categorization we can interpret information retrieval as the use of descriptive models. In addition to purely algorithmic methods, data mining comprises also interactive approaches for exploratory data analysis. This is particularly interesting when the user has no good idea of what data to expect.

Another important categorization of data mining methods is the distinction of unsupervised and supervised methods, as in general for machine learning. In supervised methods the algorithms are provided with data that provides the intended outputs for prediction tasks.

Components of Data Mining Algorithms

1. Pattern structure/model representation
   - What we look for?

2. Scoring function
   - How well the model fits the data set?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Each data mining algorithm can be characterized by four components:

- The models or patterns that are used to describe what is searched for in the data set. Typical examples of models are dependencies, clusters and decision trees.
- The scoring functions that are used to determine how well a given data set fits the model. In information retrieval these were the different measures used to evaluate the quality of a method, like recall or precision.
- The method that is applied in order to identify models from the data set that score well with respect to the scoring function. This requires efficient search algorithms to quickly limit the relevant models based on analyzing the data.
- Finally, the scalable implementation of the methods for large data sets. Here indexing techniques and efficient secondary storage management are typically required. This corresponds to the use of inverted files in information retrieval.

Components of Data Mining Algorithms

3. Optimisation and search
   - How to tune the parameters of the model? [opt]
   - How to find data satisfying a pattern? [search]

4. Data management
   - How to implement the algorithm for very large data sets?

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In particular, the aspects of optimization of data search and efficient data management of large dataset delimit data mining from related areas such as statistics and machine learning: data mining algorithms can be understood as statistical or machine learning techniques that scale well to large data sets.

Data mining algorithms are part of data mining system that support the pre-processing of the date and post-processing of the results. A data mining system performs the following typical tasks:

- First, the data needs to be collected from the available data sources. Since these data sources can be distributed and heterogeneous databases, database integration techniques are applied. The integrated data is kept in so-called data warehouses or data lakes, databases replicating and consolidating the data from the various data sources. An important concern when integrating the data sources is data cleaning, i.e., removing inconsistent and faulty data as far as possible. The tasks of data integration and data cleaning are supported by so-called data warehousing systems.
- Once the data is consolidated in a data warehouse, subsets of the data can be selected from the data warehouse for performing specific data mining tasks, i.e., tasks targeting a specific question. This task-specific data collections are sometimes called data-marts. The data-mart is the database to which the specific data mining algorithm is applied.
- The data mining task is the process of detecting interesting patterns in the data. This is what generally is understood as data mining in the narrow sense. We will introduce in the following examples some of the most common techniques that are used to perform this task (e.g., association rule mining).
- Once specific patterns are detected they can be further processed. Further processing may include the evaluation of the "interestingness" of patterns for the specific problem at hand and the implementation of actions to react on the discovery of a pattern.

Each of the steps described can influence the other steps. For example, patterns or outliers detected during data mining may indicate the presence of erroneous data, rather than interesting features in the source databases. This may imply adaptations of the data cleaning process during data integration.

Data Mining ≠ Machine Learning

What is in common

- In DM for data analytics frequently typical ML methods are used, though not always (e.g., visual mining, simple statistics)

What is different

- Data: DM is always applied to large datasets, ML not (e.g., reinforcement learning)
- Scope: DM comprises of the whole process of data integration, cleaning, analysis
- Goal: DM aims at detecting unsuspected patterns, ML may have other goals (e.g., winning a game)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Often data mining and machine learning are used like synonyms. Though both are exploiting the same algorithmic techniques, they can have different scope and goals. In particular, data mining is specifically targeted at analysis of large datasets and concerned with the resulting performance questions.

Detecting unusual behaviours in network logs is typically done by using

A. local rule discovery
B. predictive modelling
C. descriptive modelling
D. exploratory data analysis
E. all of these methods can be applied

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The classical example of a data mining problem is "market basket analysis". Retail stores gather information on what items are purchased by their customers. The expectation is, by finding out what products are frequently purchased together (i.e., are associated with each other), identifying ways to optimize the sales of the products by better targeting certain groups of customers (e.g., by planning the layout of a store or planning advertisements). A well-known example was the discovery that people who buy diapers also frequently buy beers (probably exhausted fathers of small children). Therefore, nowadays one finds frequently beer close to diapers in supermarkets, and of course also chips close to beer. This type of problem was the starting point for one of the best-known data mining techniques: association rule mining.

5.2.1 Association Rules

We search association rules of the form

Body ➔ Head [support, confidence]

1. Body: predicate(*x*, item) a property of *x*
2. Head: predicate(*x*, item) a property likely to be implied by the Body
3. Support, confidence: measures of the validity of the rule

Example: buy(x, diapers) ➔ buy(x, beer) [0.5%, 60%]

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Association rule mining is a technique for discovering unsuspected data dependencies and is one of the best-known data mining techniques. An association rule captures dependencies of different properties of an entity, denoted by x. For shopping basket analysis, the entity would be a shopping basket. The properties are expressed as predicates involving some items. The resulting rules state that if the entity has the property stated in the body, then the entity has also the property stated in the head with the probability stated by the confidence. The support expresses additionally how well-founded the rule is. We will introduce later in detail how these measures are defined.

In principle, such rules could be easily found by an exhaustive exploration of all possible dependencies, which is however prohibitively expensive. Association rule mining thus solves the problem of how to search efficiently for those dependencies.

Single- and Multi-dimensional Rules

Single-dimensional rules

buy(x, diapers) ➔ buy(x, beer) [0.5%, 60%]

Multi-dimensional rules

age(x, 19-25) $\wedge$ buy(x, chips) ➔ buy(x, coke) [10%, 75%]

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In the simplest case association rules use only a single predicate, such as buy. It is not hard to imagine that also other properties of the entities could be used in rules. For example, in the shopping basket example the age of the buyer could be another property of the shopping basket. Then we are talking about multi-dimensional rules.

From Multi- to Single-dimensional Rules

Use predicate/value pairs as items

age(x, 19-25) $\wedge$ buy(x, chips) ➔ buy(x, coke)

⇩

customer(x, age=19-25) $\wedge$ customer(x, buy=chips) ➔ customer(x, buy=coke)

Simplified notation of single-dimensional rules

{diapers} ➔ {coke}

{age=19-25, buy=chips} ➔ {buy=coke}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

However, it is straightforward to transform multi-dimensional association rules into single-dimensional rules, by combining predicate names and values into a new set of items of uniform type, consisting of predicate/value pairs. Therefore, we need to consider in the following only the case of single-dimensional rules. Since for similar rules there is only a single predicate applied to the entities, we can simplify the notation by dropping this predicate at all, and simply list the set of items in the body and head of the rules. This results in the usual notation adopted for association rules that captures dependencies among sets of items, or itemsets as they are usually called.

Basic Concepts

Transaction ID	Items Bought
2000	beer, diaper, milk
1000	beer, diaper
4000	beer, milk
5000	milk, eggs, apple

For a rule body ➔ head

Support: probability that body and head occur in transaction
p(body ∪ head)

Confidence: probability that if body occurs, also head occurs
p(head | body)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Association rule mining uses as input a set of transactions that consist of itemsets. We can illustrate the relationships among items using Venn diagrams. Each subset of the diagram represents a set of transactions containing some itemset or item. The measures of support and confidence for a rule body -> head are then defined as follows. The support gives the probability that a transaction contains all the items occurring in the rule, i.e., the probability that both all items of the body and of the head occur in the transaction. In other words, this is the probability that the rule applies to a transaction. The confidence is the probability that if the items of the body of a rule occur in the transaction, then also the items in the head of the rule occur in the transaction. In other words, this is the probability that the rule makes a correct prediction. The support expresses of how many relevant transactions exist for a rule, and therefore how well-founded it is, whereas the confidence expresses how reliable a rule is.

Support and Confidence

Transaction ID	Items Bought
2000	beer, diaper, milk
1000	beer, diaper
4000	beer, milk
5000	milk, eggs, apple

Rule 1: {beer} ➔ {diaper}

p({beer, diaper})=2/4
p({diaper} | {beer})=2/3

{beer} ➔ {diaper}[50%, 66%]

Rule 2: {diaper} ➔ {beer}

p({diaper, beer})=2/4
p({beer} | {diaper})=2/2

{diaper} ➔ {beer}[50%, 100%]

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we compute support and confidence for two different rules.

Considering two itemsets A and B, we can define one association rule for each direction, A->B and B->A. Assuming that the support for the union of items in A and B is high, we can find 4 different combinations for the confidence of the two possible rules. This shows that an association rule is indeed directed, and not necessarily symmetric. A directed association rule, which is in general the interesting case, is related to an approximate subset relationship of the corresponding transaction sets.

Definition of Association Rules

Terminology and Notation

- Set of all items I, subset of I is called **itemset**
- **Transaction** (tid, T), $T \subseteq I$ itemset, transaction identifier tid
- Set of all transactions D (**database**), Transaction $T \in D$

Association Rules $A \rightarrow B$ [s, c]

- A, B itemsets ($A, B \subseteq I$)
- $A \cap B$ empty
- $s(A \rightarrow B) = s(A \cup B) = p(A \cup B) = count(A \cup B) / |D|$ (support)
- $c(A \rightarrow B) = p(B \mid A) = s(A \cup B) / s(A)$ (confidence)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we summarize the concepts and notations that are used for specifying and characterizing association rules. Association rules are defined for itemsets that are subsets of set of times. They are defined with respect to a set of transactions that are itemsets. The rules are defined such that the body and head of the rule have empty intersection. The support is defined as the probability of the union of the body and head to occur in a transaction. This probability is computed by counting the number of transactions that contain those items. The confidence is the conditional probability of the head of the rule occurring, given the body. This probability can be computed from the support of the rule, and the support of the body.

10 itemsets out of 100 contain item A, of which 5 also contain B. The rule A ➔ B has:

A. 5% support and 10% confidence
B. 10% support and 50% confidence
C. 5% support and 50% confidence
D. 10% support and 10% confidence

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

10 itemsets out of 100 contain item A, of which 5 also contain B. The rule B ➔ A has:

A. unknown support and 50% confidence
B. unknown support and unknown confidence
C. 5% support and 50% confidence
D. 5% support and unknown confidence

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Association Rule Mining: Problem

Problem

Given a database D of transactions (tid, T)

Find

all rules A ➔ B [s, c]

such that $s > s_{min}$ (high support)

and $c > c_{min}$ (high confidence)

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The problem of mining association rules is now defined as follows: Given a database of transactions, find all rules that correlate the presence of one itemset with that of another itemset where the support and the confidence are above a given threshold. In other words, find all association rules that have high support and confidence.

5.2.2 Apriori Algorithm

Two step approach:

1. Find **frequent** itemsets
   $J \subseteq I$ such that $p(J) > s_{min}$

2. Select **pertinent** rules
   A ➔ B such that $A \cup B = J$
   and
   $p(B|A) > c_{\min}$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

We will approach the problem in two steps, by first considering only the problem of finding itemsets that satisfy the condition on having a high support, which is a necessary condition for a rule to be an association rule, and then extracting from those itemsets the rules that have a high confidence. We have already noticed in the confidence value can be computed from support values of itemsets. Thus it is natural to first identify itemsets with high support.

Frequent Itemsets

Transaction ID	Items Bought
2000	beer, diaper, milk
1000	beer, diaper
4000	beer, milk
5000	milk, eggs, apple

Frequent Itemset	Support
{beer}	0.75
{milk}	0.75
{diaper}	0.5
{beer,milk}	0.5
{beer, diaper}	0.5
{milk, diaper}	0.25

1. A ➔ B can be an association rule, only if A ∪ B is a frequent itemset
2. Any subset of a frequent itemset is also a frequent itemset (**Apriori property**)

$$p(J) > s_{min} \rightarrow p(J' \subseteq J) > s_{min}$$

3. Find frequent itemsets with increasing cardinality, from 1 to k, to reduce the search space

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

A necessary condition for finding an association rule of the form A->B is that it has a sufficiently high support. Therefore, for finding such rules, we can first identify itemsets within the transactions that occur sufficiently frequent. These are called *frequent itemsets*. Second, we can observe that any subset of a frequent itemset is necessarily also a frequent itemset. This is called the *apriori property*. The a priori property is the central idea in the algorithm we will introduce in the following. It will be used for multiple purposes. A first use of this property is the observation that we can search for frequent itemsets by searching them by increasing cardinality: once frequent itemsets of lower cardinality are found, only itemsets of larger cardinality need to be considered that contain one of the frequent itemsets already found. This allows us to dramatically reduce the search space.

Exploiting the Apriori Property

If we know all frequent (k-1)-itemsets, which are candidate frequent k-itemset **X**?

1. Any (k-1)-itemset that is subset of **X** must be frequent
2. Any ordered (k-1)-itemset that is subset of **X** must be frequent
3. *In particular, two ordered (k-1)-itemsets that are subsets of **X** and differ only in their last position must be frequent*

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Assume that we know frequent itemsets of size k-1 and want to construct an itemset of size k (a k-itemset). What properties does such a k-itemset have to satisfy? We can say, that any subset of size k-1 must be frequent. We can also order those subsets by their elements. We can that even say that two ordered subsets that differ only in the last position must be frequent. This gives is now a possibility to construct k-itemsets systematically.

Exploiting the Apriori Property: Candidates

The union of two (k-1)-itemsets that differs only by one item is a **candidate** frequent k-itemset

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

If two ordered subsets of size k-1 of a frequent itemset of size k that differ only in the last position must be frequent, we can invert the argument. Assume we know all frequent itemsets of size k-1 then, then the itemset of size can only be frequent if it can be constructed from two k-1 itemsets in exactly this way. An itemset of size k constructed this way is then called a candidate frequent k-itemset. We do not know whether it is frequent indeed. The construction is a necessary condition, but not a sufficient one.

Example

Transaction ID	Items Bought
2000	beer, diaper, milk
1000	beer, diaper
4000	beer, milk
5000	milk, eggs, apple

Frequent Itemset	Support
{beer}	0.75
{milk}	0.75
{diaper}	0.5
{beer,milk}	0.5
{beer, diaper}	0.5
{milk, diaper}	0.25

{beer} and {milk} differ by one item, thus {beer,milk} is a candidate 2-itemset

{beer, milk} and {beer, diaper} differ by one item, thus {beer, diaper, milk} is a candidate 3-itemset

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Note that the candidate itemsets generated by the join step are not necessarily frequent itemsets, as the example illustrates.

Exploiting the Apriori Property: JOIN step

Given the frequent (k-1)-itemsets, L_{k-1}, we can construct a candidate set C_k by joining two (k-1)-itemsets that differ by exactly 1 item in the last position

Algorithm (**JOIN**)

1. Sort the itemsets in L_{k-1}
2. Find all pairs with the same first k-2 items, but different k-1th item
3. Join the two itemsets in all pairs and add to C_k

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

In order to create candidates efficiently we consider only one specific way of constructing candidates. We only combine itemsets that have the first k-2 elements identical and differ in their last position. If a k-itemset is frequent it necessarily is (also) constructed this specific way.

For example, if we have

ABC

ACD

BCD

If we combine ACD and BCD (which also differ by one item) we would get ABCD. But ABCD have been already generated by combining ABC and ABD.

JOIN step: Example

L_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 4}
{2 5}

join

C_3

itemset
{1 2 3}
{1 2 5}
{1 3 5}
{2 3 4}
{2 3 5}
{2 4 5}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we illustrate the join step for itemsets of size 2, to construct itemsets of size 3. Only pairs of itemsets of size 2 are considered that coincide in their first position.

The k-itemsets constructed in the join step are not necessarily frequent k-itemsets. One possible reason is that they contain some subsets of items which are not a frequent itemset. These can be eliminated in a prune step. By checking if every (k-1) itemset of a candidate itemset is indeed a frequent (k-1) itemset some of the candidate itemsets can be eliminated.

Final Step

For the remaining k-itemset in the candidate set C_k eliminate those that are not frequent by counting how often they occur in the database

- The final step is the most expensive (full access to database D)
- Advantage: Performed only for a smaller number of candidate itemsets, reduced by **JOIN** and **PRUNE**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

After the prune step is completed, still the remaining candidates can be non-frequent. Therefore, the frequency of these itemsets needs to be determined by accessing the database and counting the number of their occurrences. An important aspect of the apriori algorithm is that this last step is the most expensive one, since it requires access to the complete database. By performing the join and prune steps the operation needs to be performed only for a reduced number of itemsets.

Apriori Algorithm

```
k := 1, L_k := { J ⊆ I : |J|=1 ∧ p(J) > s_min }        //frequent items
while L_k != ∅
        C'_{k+1} := JOIN(L_k)            // join itemsets in L_k that differ by one element
        C_{k+1} := PRUNE(C'_{k+1})                     // remove non-frequent itemsets
        for all (id,T) ∈ D                             // compute the frequency of candidate
                for all J ∈ C_{k+1} , J ⊆ T
                        count(J)++
                end for
        end for
        L_{k+1} := { J ⊆ C_{k+1} : p(J) > s_min }      // add candidate frequent itemsets
        k := k+1
end while
return ∪_k L_k
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This is the summary of the complete apriori algorithm.

Here we give a complete example of the exection the apriori algorithm. Notice, in this example, how the scan steps (when determining the frequency with respect to the database) eliminate certain itemsets. Pruning does not lead to any elimination of itemsets in this example. The algorithm built 9+3 (frequent + non frequent) = 12 itemsets out of 2^5 = 32 total possible itemsets

Select pertinent rules

```
R := ∅                                  // initial set of rules
L := Apriori(D)                         // set of frequent itemsets
for all J ∈ L
        for all A ⊆ J, A != ∅
                r := A ➔ J \ A          // create candidate rule
                if c(A ➔ J \ A) = s(J)/s(A)> c_min
                        R := R ∪ r
                end if
        end for
end for
```

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Once the frequent itemsets are found with the apriori algorithm, the derivation of relevant association rules is straightforward. One checks for every frequent itemset whether there exists a subset A that can occur as the body of a rule. For doing that, the support count, i.e., the frequency of the itemset in the database, which was obtained during the execution of the Apriori algorithm, is used to compute the confidence as a conditional probability. Note that also L\A is a frequent itemset, and therefore the support count is available for that set as well from the apriori algorithm.

Note that $c(A \rightarrow J \setminus A) = s((J \setminus A) \cup A)/s(A) = s(J)/s(A)$.

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2
{2 3 5}	2

c_{min}= 0.75

J={1,3}

A={1}, J\A={3} {1} ➔ {3} c = sup({1,3})/sup({1}) = 1 OK

A={3}, J\A={1} {3} ➔ {1} c = sup({1,3})/sup({3}) = 0.66 KO

J={2,3,5}

A={3,5}, J\A={2} {3,5} ➔ {2} c = sup({2,3,5})/sup({3,5}) = 1 OK

A={2}, J\A={3,5} {2} ➔ {3, 5} c = sup({2,3,5})/sup({2}) = 0.66 KO

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we illustrate the extraction of association rules using the known set of frequent itemsets.

Given the frequent 2-itemsets {1,2}, {1,4}, {2,3} and {3,4}, how many 3-itemsets are generated and how many are pruned?

A. 2, 2

B. 1, 0

C. 1, 1

D. 2, 1

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

After the join step, the number of k+1-itemsets ...

A. is equal to the number of frequent k-itemsets
B. can be equal, lower or higher than the number of frequent k-itemsets
C. is always higher than the number of frequent k-itemsets
D. is always lower than the number of frequent k-itemsets

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

If rule {A,B} ➔ {C} has confidence c_1 and rule {A} ➔ {C} has confidence c_2, then ...

A. $c_2 >= c_1$
B. $c_1 > c_2$ and $c_2 > c_1$ are both possible
C. $c_1 >= c_2$

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Interesting Association Rules

Not all high-confidence rules are interesting

	$Coffee$	$\overline{Coffee}$	
Tea	150	50	200
$\overline{Tea}$	650	150	800
	800	200	1000

{Tea} ➔ {Coffee} has support 0.15 and confidence 0.75

- But 80% of people drink coffee anyway
- Drinking tea decreases the probability to drink coffee!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

High confidence and support do not necessarily imply that a rule is interesting or useful. We illustrate this by the following example. The matrix indicates the number of entities that like or do not like coffee or tea. If we consider the rule {Tea} –> {Coffee} we will find that it has support 0.15 and confidence 0.75, which we may consider as very high. At the first glance this looks like a good rule. Looking more carefully, we will realize that in fact 80% of people drink coffee. In view of this, the rule just holds, because people in general drink a lot of coffee. Even worse, considering that 80% of people drink coffee, among the people drinking tea the propensity to drink coffee is less pronounced, only 75%. In conclusion, drinking tea has a negative impact on drinking coffee which is the opposite to what the rule suggests. This motivates the need for alternative measures to evaluate whether a rule is interesting.

Alternative Measures of Interest

Added Value (A, B itemsets)

$$AV(A \rightarrow B) = confidence(A \rightarrow B) - support(B)$$

Interesting rules are those with high positive or negative interest values (usually above 0.5)

Alternative: $Lift(A \rightarrow B) = \frac{confidence(A \rightarrow B)}{support(B)}$

Example:

- AV({Tea} ➔ {Coffee}) = 0.75 – 0.8 = -0.05
- Lift({Tea} ➔ {Coffee}) = 0.75 / 0.8 = 0.9375

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Quantitative Attributes

Transforming quantitative (numeric ordered values) into categorical ones

- *Static discretisation* into predefined bins
- *Dynamic discretisation* based on the distribution of the data

The rules depend on the chosen discretisation!!

(1) age{*x*, [18,19]} ∧ live{*x*, Lausanne} ➔ work{*x*, student}

(2) age{*x*, [20,21]} ∧ live{*x*, Lausanne} ➔ work{*x*, student}

vs.

(1) age{*x*, [18,21]} ∧ live{*x*, Lausanne} ➔ work{*x*, student}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Association rules can also be derived for quantitative data. In order to obtain finite itemsets, the quantitative domains are transformed to discrete ones using discretization. Methods of discretization are widely used in machine learning for data preprocessing. Two basic approaches are static discretization, splitting the data in bins of uniform size, and dynamic discretization splitting the data in bins with uniform number of elements. Also, semantic aspects can be taken into consideration when performing discretization, as the example illustrates. A static discretization could create there bins that are semantically less meaningful.

Improving Apriori for Large Datasets

1. Transaction reduction
   - A transaction that does not contain any frequent k-itemset is useless in subsequent scans
2. Sampling
   - Mining on a sampled subset of DB, with a lower support
3. Partitioning (SON algorithm)
   - Any itemset that is potentially frequent in a DB must be frequent in at least one of the partitions of the DB

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Though the basic Apriori algorithm is designed to work efficiently for large datasets, there exist several possible improvements:

•Transactions in the database that turn out to contain no frequent k-itemsets can be omitted in subsequent database scans.

•The sampling method selects samples from the database and searches for frequent itemsets in the sampled database, using a correspondingly lower threshold for the support.

•One identify first frequent itemsets in partitions of the database, that fit in memory. This method exploits the idea that if an itemset is not frequent in any of the partitions (local frequent itemset) then it is also not be frequent in the whole database.

Sampling

Approach

1. Randomly sample transactions with probability p
2. Detect frequent itemsets with support p*s
3. Eliminate **false positives** by counting frequent itemsets on complete data after discovery

Refinements

- If we assume that the m transactions are randomly sorted, we can just choose the first p*m ones
- **False negatives** can be reduced by choosing a lower threshold, e.g., 0.9 p*s

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Random sampling data before searching association rules must consider two issues: false positives and false negatives. False positives can be dealt with by verifying the support for the itemsets on the complete transaction set. For false negatives, no such possibility exists. By decreasing the support threshold, the number of false negatives can be reduced, at the cost of testing more candidate itemsets when scanning the complete dataset.

A further optimization can be achieved, when we know that the transactions occur in random order. Then sampling can be replaced by simple using the first p*m elements of the database, where m is the size of the database.

Partitioning

Approach

1. Divide transactions in 1/p partitions and repeatedly read partitions into main memory
2. Detect in-memory algorithm to find all frequent itemsets with support threshold p*s
3. An itemset becomes a candidate if it is found to be frequent in **at least** one partition
4. On a **second pass**, count all the candidate itemsets and determine which are frequent in the entire set of transactions

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

With partitioning we are not sampling the dataset but processing the complete transaction set partition by partition. If an itemset is frequent in one partition (with a correspondingly adapted threshold), then it becomes a candidate. At this stage it is not sure that it is also frequent for the whole transaction set. Therefore, in a second pass itemsets that are frequent are determined by scanning the whole transaction database.

Partitioning – Why it works

Key "monotonicity" idea

- an itemset cannot be frequent in the entire set of transactions unless it is frequent in at least one subset
- Proof: If for all partitions support is below p*s, the total support is less than (1/p) p*s = s!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The second pass is needed, since the condition of being frequent in at least one partition is necessary, but not sufficient.

Partitioning – Distributed Version

Partitioning lends itself to distributed data mining

Transactions distributed among many nodes

- Compute frequent itemsets at each node
- Distribute candidates to all nodes
- Accumulate the counts of all candidates

MapReduce implementation!

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Partitioning is an embarrassingly parallel algorithm, and can therefore be naturally executed in a distributed environment, e.g., using Map-Reduce

A false negative in sampling can only occur for itemsets with support smaller than …

A. the threshold s

B. p*s

C. p*m

D. None of the above

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

5.2.3 FP Growth

Frequent itemset discovery <u>without</u> candidate itemset generation

Proceeds in 2 steps:

1. Build a data structure, called the FP-tree
   – Requires 2 passes over the dataset
2. Extract frequent itemsets directly from the FP-tree

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Though Apriori is as the original method the most popular association rule mining algorithm, other algorithms have been devised that attempt to provide better performance under certain circumstances. FP Growth is one example of such an algorithm that is mainly aiming at main memory implementations (and thus less suitable for distributed implementation). It does not generate candidates for frequent itemsets but constructs them directly using a data structure called FP-Tree. The FP-Tree allows to extract directly the association rules.

FP-Tree Data Structure

Transaction Data Set

TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

- Items are sorted by decreasing support
- Nodes correspond to single items
- Counters indicate frequency of itemset from root to node
- Different nodes for the same item are connected by a linked list

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The structure of an FP-Tree is comparable to that of a trie. In order to construct the FP-Tree, first items in the itemsets are sorted. The nodes in the tree correspond to items, and paths from the root correspond to itemsets. Itemsets sharing the same prefix of items, share the same path prefix in the tree. Counters at the nodes indicate how frequent the itemset corresponding to the path from the root to the node is. If we consider the left most path, we see that item **a** occurs 8 times, itemset ab 5 times, **abc** 3 times, and **abcd** once. The same item can occur in the tree in multiple places. The different occurrences are connected as linked list. For itemsets that occur in different paths (e.g., bc, which occurs together with **a** and without **a**), the frequencies can be computed by following the linked lists of the last item in the itemset and testing whether the remaining items are present in the path.

FP-Tree: Item Sorting

Transaction Data Set

TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

Item frequencies
a: 8
b: 7
c: 6
d: 5
e: 3

FP-Tree Size: The FP-Tree is more compact the more common prefixes the itemsets share

- Sorting items by decreasing support increases this probability

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Sorting items in itemsets by decreasing support is an important heuristics to keep the FP-Tree compact, as more frequent items tend to be on the top of the tree which increases the likelihood that different itemsets share a common prefix. In the example the items were sorted already according to their frequencies. If it is not the case, they need to be reordered according to their frequencies.

Step 1: FP-Tree Construction

Requires 2 passes over the dataset

Pass 1: Compute support for each item

- Pass over the transaction set
- Sort items in order of decreasing support

Pass 2: Construct the FP-Tree data structure

- Tree is expanded one itemset at a time

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

To construct the FP-Tree one proceeds in two passes (not to confuse with the two steps of the overall FP-Growth algorithm mentioned in the beginning). In a first pass the transactions are scanned to compute the support for each single item. The items are then sorted in decreasing order. This order is being used to sort itemsets before the construction of the FP-Tree. In the second pass the FP-Tree is constructed by adding one itemset at a time.

Step 1: FP-Tree Construction

Transaction Data Set

TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}

Adding itemset 1

Adding itemset 2

Adding itemset 3

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Constructing the FP-Tree proceeds in a way analogous to constructing a trie structure. The itemsets in the transaction database are processed sequentially. The items in an itemset are processed one by one. If for the current item there exists already a node in the FP-Tree, no new node is created. If it does not exist, a new branch is created for the item. The number of occurrences is updated by 1 at each node that is visited (since each subset of the itemset is also an itemset). Finally, links are added between nodes with the same labels.

Step 2: Frequent Itemset Extraction

For each item extract the tree with paths ending in the item

- Used to find frequent itemsets containing the item

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

After the FP-Tree has been constructure in step 1, In the second step frequent itemsets are extracted from it. For each item, first a subtree is extracted from the full FP-Tree that contains only the paths ending in the item. For extracting those subtrees, the linked lists are helpful. One needs to traverse just the corresponding list and retain the paths from the traversed nodes to the root. The figures illustrates the resulting subtrees for the 5 items.

Step 2: Divide and Conquer Strategy

Start with item with lowest support (e.g., item *e*) and extract its tree

Check whether the item has sufficient support (e.g., 2)

- Add up counts along the list of the item
- e.g., item *e* has support 3

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

For extracting frequent itemsets, the algorithm starts with the item that has the lowest support. It extracts its corresponding tree, as described before. Next the support of the item can be computed by traversing the leaves of the tree, following the links. In the example, item **e** would this have support 3. If the support is above threshold, then processing proceeds to find larger itemsets containing the item, otherwise processing can stop since no frequent itemsets containing this item can exist.

Step 2: Divide and Conquer Strategy

If item has sufficient support, check for itemsets ending in the item

- Example: if item *e* is frequent check whether itemsets *de, ce, be, ae* are frequent
- again, in order of lowest support

In order to check whether these itemsets are frequent compute a **conditional FP-Tree**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

If we find that the current item has sufficient support (e.g., item e), then we check next whether the itemsets consisting of two items and ending in the item have also sufficient support. For doing so we derive a **conditional FP-Tree** from the tree that we have constructed for the item before. The conditional tree is constructed in three steps:

- First, the support counts in the tree are updated such that only the number of itemsets containing the current item is considered. This can be performed traversing the paths from the bottom to the root.
- Second, the leaf nodes corresponding to the current item are removed
- Finally, all nodes that have an insufficient support count are also removed from the conditional FP Tree

Conditional FP-Tree

Corresponds to the FP-Tree that would be built if

- we only consider transactions containing a particular itemset
- then omit this itemset (e.g., {e})
- drop items that are not frequent in the subset of the transaction set (e.g., {b})

Transaction Data Set

TID	Items
~~1~~	~~{a,b}~~
~~2~~	~~{b,c,d}~~
3	{a,c,d,~~e~~}
4	{a,d,~~e~~}
~~5~~	~~{a,b,c}~~
~~6~~	~~{a,b,c,d}~~
~~7~~	~~{a}~~
~~8~~	~~{a,b,e}~~
~~9~~	~~{a,b,d}~~
10	{b,c,~~e~~}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

One way to understand the construction of the conditional FP Tree is that it is the FP Tree that would be constructed when we only consider the transactions that consider the current item (or itemset) and then remove **e** and any non-frequent item from those itemsets. In the example this would be item **e** that we consider, and item **b** would be eliminated, since not frequent in the remaining transactions.

Deriving Conditional FP-Tree

The conditional FP-Tree can be derived from extracted tree

1. Update support counts to itemsets containing the item
2. Remove the nodes of the item
3. Remove nodes with insufficient support count

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Here we illustrate the constructing of the conditional FP-Tree for **e**. Note, when we update the support counts, for the node **b**:2, there are are two itemsets that contribute to the support count of **b**. The itemset {b} and the itemset {b,c,e}. Since only one of them contains the item **e**, the support count for **b** is changed to b. Similarly for **a**:8 and **c**:2. Since the total support for **b** drops below 2, the node is finally removed from the tree.

Here we show all conditional FP-Trees that would be constructed from 2-itemsets ending in item **e**. Once the conditional FP Trees have been constructed, we can read off the support for all 2-itemsets, by traversing the lists at the leaf level (as we did first for item **e** itself). Once this step is finished, the algorithm continues in the same way for the remaining 2-itemsets. For example, for itemset **de** we would now construct a conditional FP Tree, by first extracting from tree (b) the tree with paths ending in **de**, resulting in tree (c), and then extracting the conditional FP Tree for itemset **de** (tree (d)).

Result

Frequent itemsets detected in this order

Suffix	Frequent Itemsets
e	{e}, {d,e}, {a,d,e}, {c,e},{a,e}
d	{d}, {c,d}, {b,c,d}, {a,c,d}, {b,d}, {a,b,d}, {a,d}
c	{c}, {b,c}, {a,b,c}, {a,c}
b	{b}, {a,b}
a	{a}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This table summarizes the frequent itemsets in the order they are detected, i.e., starting from the items with the lowest support. The first row contains exactly the itemsets containing the item **e**, in the order in which they are detected.

In the first pass over the database of the FP Growth algorithm

A. Frequent itemsets are extracted
B. A tree structure is constructed
C. The frequency of items is computed
D. Prefixes among itemsets are determined

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

The FP tree below is …

A. not valid, b is missing
B. not valid, since count at leaf level larger than 1
C. possible, with 2 transactions {a}
D. possible, with 2 transactions {a,c}

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Performance Comparison

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

This performance comparison shows the behavior of FP Growth as compared to apriori. As the support threshold gets smaller, and therefore more rules are detected, the runtime of both algorithms increases. For smaller thresholds, FP Growth exhibtis increasingly better performance as compared to apriori., whereas for larger thresholds the advantage is less pronounced and disappears.

Summary FP Growth

Advantages

- Only 2 passes over the dataset
- Compresses dataset
- (Generally) much faster than Apriori

Disadvantages

- Works less efficiently for high support thresholds
- Must run in main memory
- Difficult to find distributed implementation

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

FP Growth Works less efficiently for high support thresholds, because it prunes only single items.

Components of Data Mining Algorithms

1. Pattern structure/model representation
   - **association rules**
2. Scoring function
   - **support, confidence**
3. Optimisation and search
   - **JOIN, PRUNE**
   - **FP-Tree, ordering of items**
4. Data management
   - **transaction reduction, partitioning, sampling**

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis

Our study of association rule mining nicely illustrates the 4 main aspects of every data mining algorithm. The model type and scoring function are common to all machine learning algorithms. When it comes to large datasets also algorithmic optimizations and efficient management of large datasets become important questions.

References

Textbook

- http://www.mmds.org/mmds/v2.1/ch06-assocrules.pdf
- Jiawei Han, *Data Mining: concepts and techniques*, Morgan Kaufman, 2000, ISBN 1-55860-489-8
- Pang-Ning Tan, Michael Steinbach, Vipin Kumar: *Introduction to Data Mining*, Addison-Wesley. Chapter 6: Association Analysis: Basic Concepts and Algorithms

Some relevant research literature

- R. Agrawal, T. Imielinski, and A. Swami. *Mining association rules between sets of items in large databases*. SIGMOD'93, 207-216, Washington, D.C.
- J. Han, J. Pei and Y. Yin. *Mining Frequent patterns without candidate generation*. SIGMOD 2000, 1–12.

©2023, Karl Aberer, EPFL-IC, Laboratoire de systèmes d'informations répartis