Software Artefact Traceability Management and Recovery

Prof. Andrea De Lucia
University of Salerno

Dr. Rocco Oliveto
University of Molise

SE@SA Lab - Software Engineering Lab

Outline

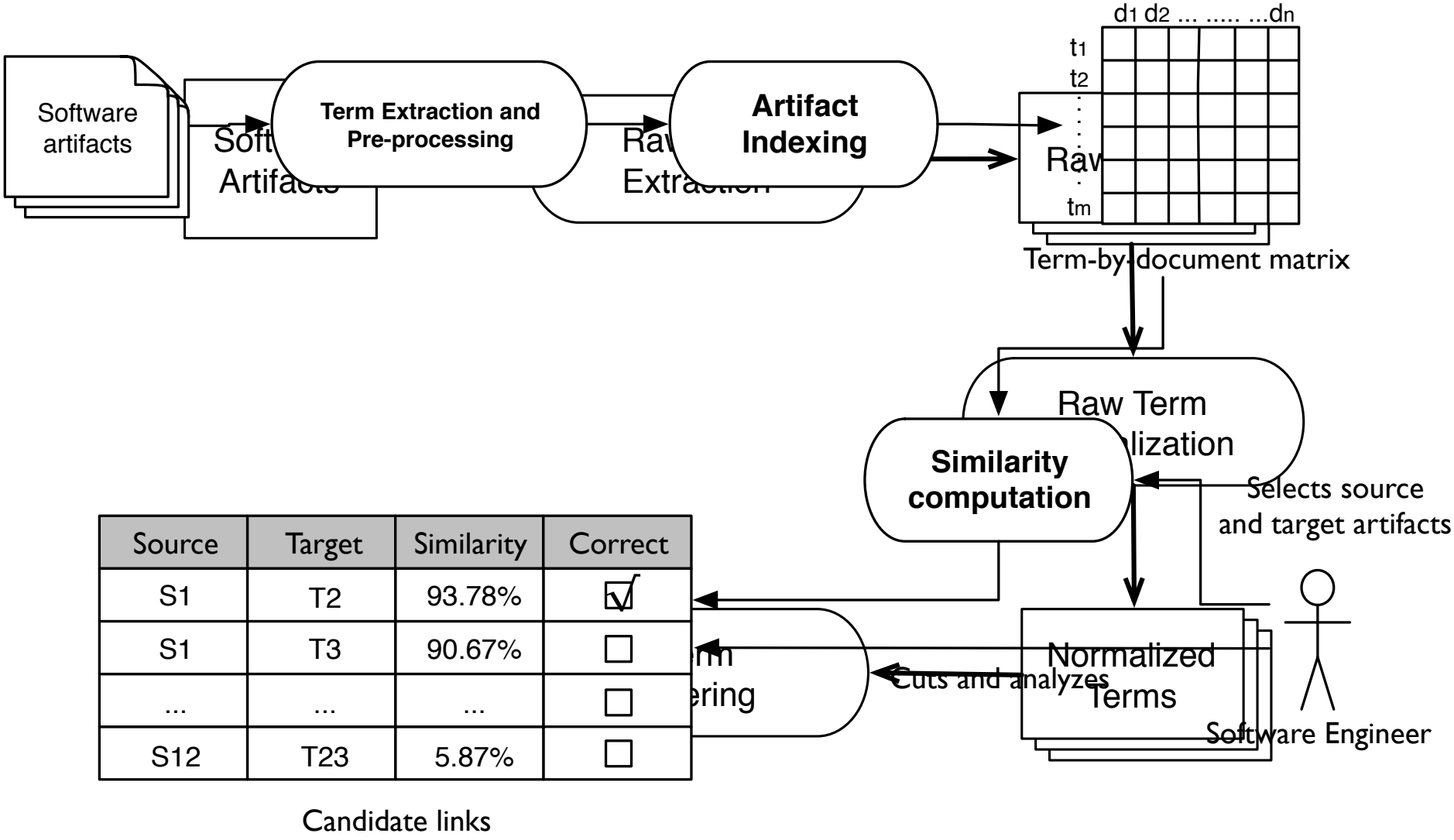
Part A: Traceability Management

- Done!

Part B: IR-based Traceability Recovery

- Traceability recovery as an IR task
- Enhancing strategies
- Empirical evaluation of IR-based Traceability Tools
- How to (effectively) use traceability recovery tool

IR-based Traceability Recovery Process



Indexing a Source Code Artefact (1)

Input - Source code artefact

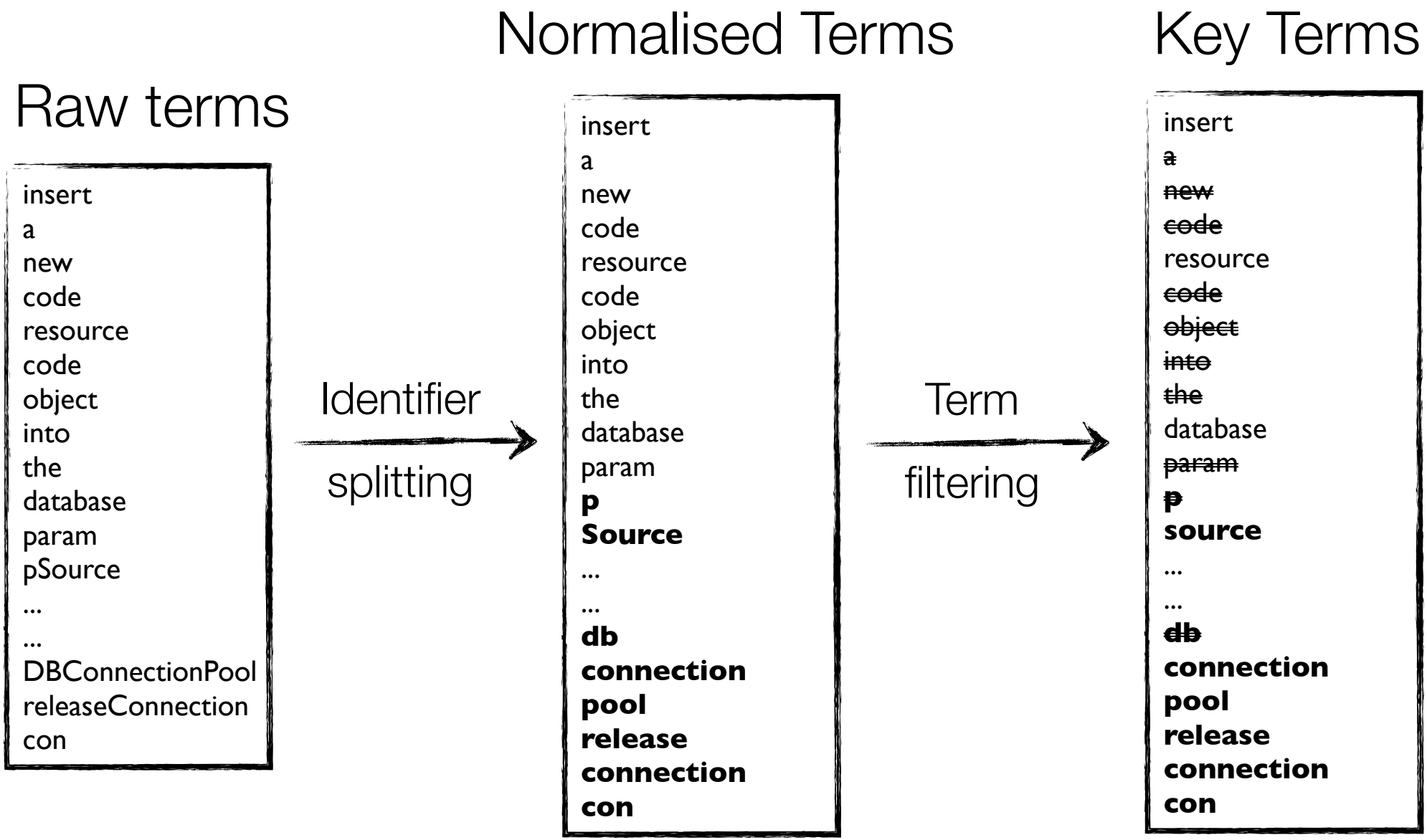
```
/**
 * Insert a new <code>Resource</code> object into the database
 *
 * @param pSource
 *        The <code>Resource</code> object to insert
 * @throws SQLException
 */
public void insert(Resource pSource)throws SQLException {
    Connection con = null;
    PreparedStatement stmt = null;
    try {
        // Obtain a db connection
        con = DBConnectionPool.getConnection();

        stmt = con.prepareStatement("INSERT INTO resources VALUES (?, ?, ?)");
        stmt.setString(1, pSource.getLogin());
        stmt.setString(2, pSource.getName());
        stmt.setString(3, pSource.getSurname());
        stmt.execute();

        // Force the commit
        con.commit();
    } finally {
        // Release the resources
        if (stmt != null)
            stmt.close();
        if (con != null)
            DBConnectionPool.releaseConnection(con);
        }
}
```

Term extraction

Raw terms

insert
a
new
code
resource
code
object
into
the
database
param
pSource
...
...
DBConnectionPool
releaseConnection
con

Indexing a Source Code Artifact (2)



Artefact Representation

Software artefacts (methods) to be indexed

```
1  public class Math {
2      /**
3       * Returns the sum between op1 and op2
4       */
5      public int sum(int op1, int op2) {
6          int sum = op1 + op2;
7          return sum;
8      }
9
10     /**
11      * Returns the difference between op1 and op2
12      */
13     public int diff(int op1, int op2) {
14         int diff = op1 - op2;
15         return diff;
16     }
17 }
18
```

Software artefacts (methods) representation

	sum	diff
returns	1	1
sum	4	0
op1	3	3
op2	3	3
differen	0	1
diff	0	3

Term-by-document matrix - Bags of words

Stemming

The process of stripping affixes, such as prefixes and suffixes, from words to form a stem

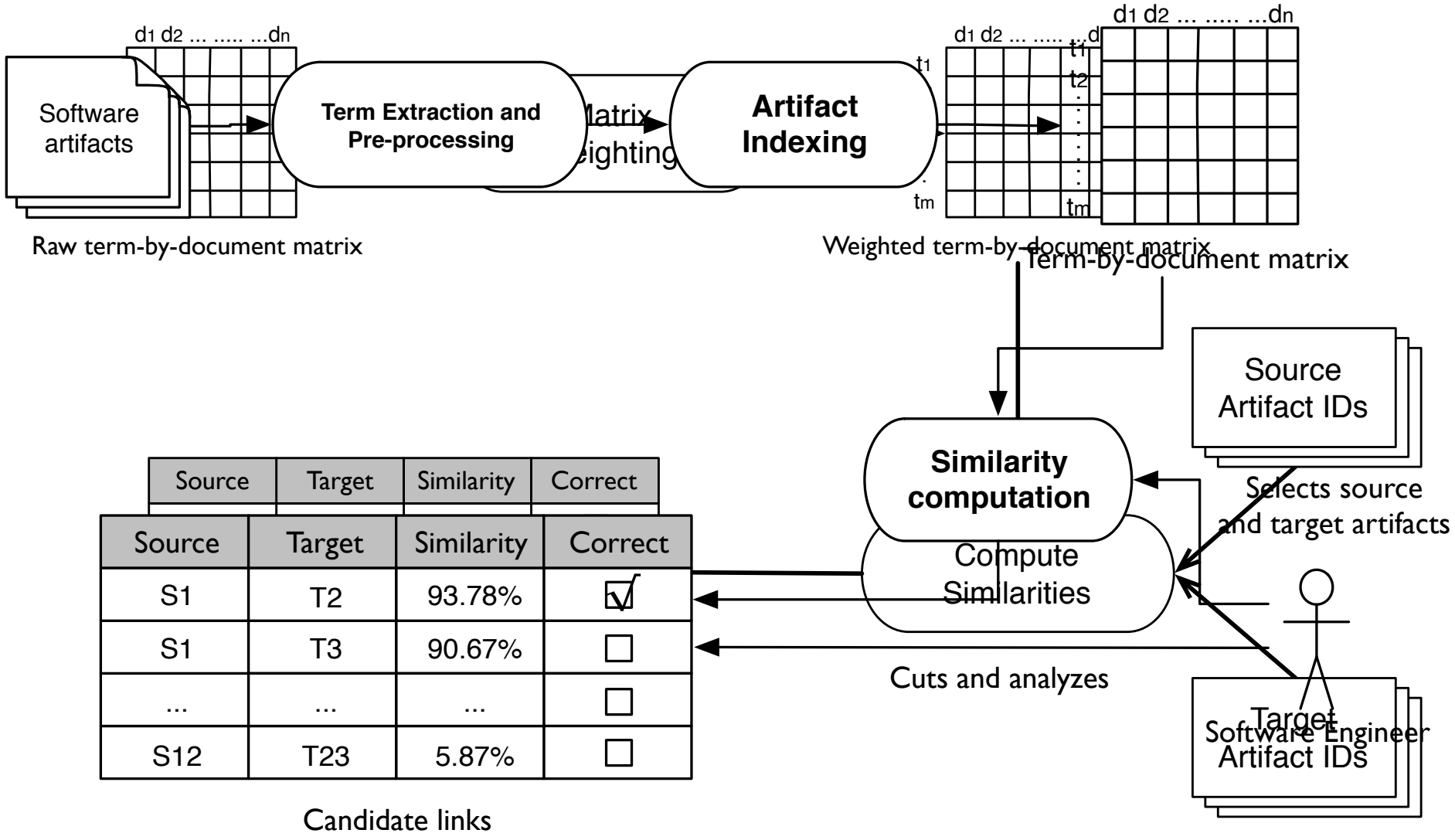
Example: terms *worker*, *working*, *works*, *worked* have the same stem *work*

Often applied to terms in order to group together as the same concept terms with almost the same meaning, but superficial spelling differences

Word conflation

Example: if a user searches for the query ‘adding auctions’ in an IR system, the user would most likely be interested in documents containing the words ‘add’ and ‘auction’. By appropriate use of stemming, IR systems can recognize that ‘add’ and ‘adding’, as well as ‘auction’ and ‘auctions’, map to the same ideas, despite surface differences in spelling.

IR-based Candidate Link Recovery Process



Traceability recovery and IR

Traceability recovery requires the use of IR models based on free text queries

- Rather than a query language of operators and expressions, the query is represented by more terms in a human language (extracted from the source artefact content)

Free text queries are usually associated with ranked retrieval models

- Rather than a set of documents satisfying a query expression, in ranked retrieval, the system returns an ordering over the (top) documents in the collection for a query

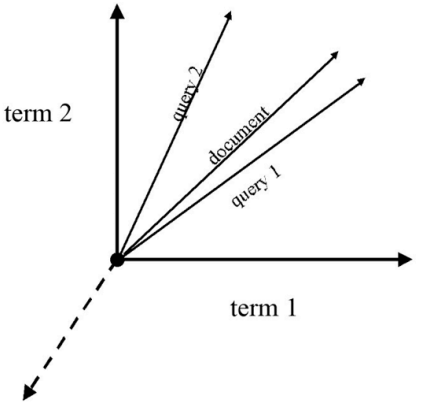
IR Models

Probabilistic model

The similarity between a source and a target artefact is based on the probability that the target artefact is related to the source artefact
Not discussed in details in this talk...

Vector space model

Source and target artefacts are represented in a vector space and the similarity is computed through vector operations, e.g. cosine of the angle between the two vectors



But also many improvements of the basic models

Term Frequency

The term frequency $tf_{t,d}$ of term *t* in document *d* is defined as the number of times that *t* occurs in *d*

Raw term frequency is not what we want!

A document with 10 occurrences of the term is more relevant than a document with 1 occurrence of the term. But not 10 times more relevant!

Relevance does not increase proportionally with term frequency

The log frequency weight of term *t* in *d* is

$$w_{t,d} = \log_{10}(1 + \mathrm{tf}_{t,d})$$

0 → 0, 1 → 1, 2 → 1.3, 10 → 2, 1000 → 4, etc.

Document Frequency (1)

Rare terms are more informative than frequent terms

- The use of a stop word list is also based on such a conjecture

Consider a term in a source artefact that is rare in the collection (e.g., student)

A document containing this term is very likely to be relevant to the query

- We want a high weight for rare terms!

Document Frequency (2)

Frequent terms are less informative than rare terms

- Consider a source artefact term that is frequent in the collection (e.g., insert, delete, modify)

A document containing such a term is more likely to be relevant than a document that does not

- But it is not a sure indicator of relevance

For frequent terms, we want high positive weights, but lower weights than for rare terms

We will use document frequency (*df*) to capture this

Inverse Document Frequency

df_t is the document frequency of t, i.e., the number of documents that contain t

df_t is an inverse measure of the informativeness of t

$df_t <= N$ (number of documents)

We define the *idf* (inverse document frequency) of t by

$$\mathrm{idf}_t = \log_{10}(N/\mathrm{df}_t)$$

We use *log(N/dft)* instead of N/df_t to "dampen" the effect of idf

tf-idf Weighting Schema

The *tf-idf* weight of a term is the product of its *tf* weight and its *idf* weight

$$w_{t,d} = \log(1 + \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

Best known weighting schema in information retrieval

- Increases with the number of occurrences within a document
- Increases with the rarity of the term in the collection
- Brings the weight to zero when the term appears in alla documents.

General Form of a Weighting Schema

Usually a weighting schema includes a local and a global weights

Local weight: based on the frequency of occurrences of the term in the artifact

This might be further weighted by the position of the term in the document (never used in traceability recovery)

Global weight: the more the term is spread in the artefact space the less it is relevant to the subject artefact

Software Artefacts as Vectors

The *term-by-document* matrix can be viewed as a vector space

- Terms are axes of the space
- Artefacts are points or vectors in this space

Using such a representation it is possible to rank target artefacts according to their proximity to the source artefact in this space

Formalising Vector Proximity

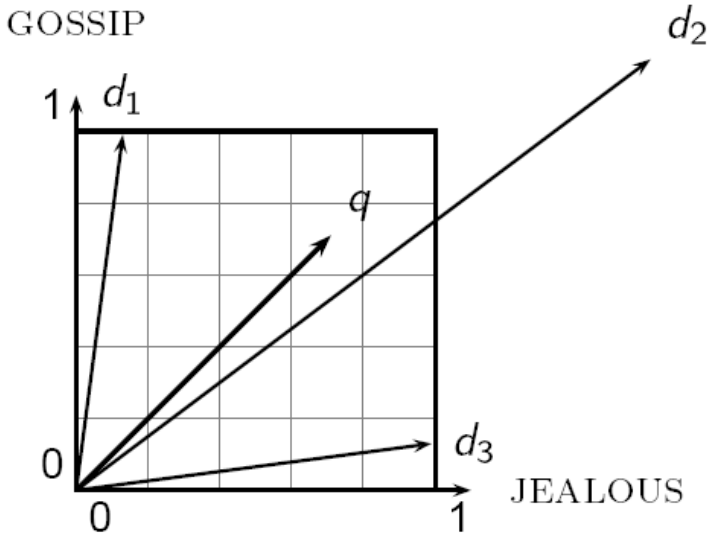
First cut: distance between two points

Distance between the end points of the two vectors

Euclidean distance?

No, it is a bad idea, because Euclidean distance is large for vectors of different lengths

The Euclidean distance between q and d_2 is large even though the distribution of terms in the query q and the distribution of terms in the document d_2 are very similar



Cosine Similarity (1)

Thought experiment: take a document d and append it to itself. Call this document d′

"Semantically" d and d′ have the same content

The Euclidean distance between the two documents can be quite large

While, the angle between the two documents is 0, corresponding to maximal similarity

Key idea: Rank target artefacts according to the angle with source artefact

The following two notions are equivalent

Rank documents in decreasing order of angle(source, target)

Rank documents in increasing order of cosine(source,target)

Cosine Similarity (2)

Cosine of the angle between the two corresponding vectors

$$sim(D,Q) = \frac{\overrightarrow{D} \cdot \overrightarrow{Q}}{\|\overrightarrow{D}\| \cdot \|\overrightarrow{Q}\|} = \frac{\sum_{t_i \in D,Q} w_{t_i D} \cdot w_{t_i Q}}{\sqrt{\sum_{t_i \in D} w_{t_i D}^2} \cdot \sqrt{\sum_{t_i \in Q} w_{t_i Q}^2}}$$

The cosine

- has values in [0,1] since the maximum angle is 90°
- increases as more terms are shared

Thus, two artefacts are considered similar if their corresponding vectors point in the same direction

- The angle is close to 0°

Cutting the Ranked List (1)

Using the IR model is possible to obtain a ranked list of pairs of artefacts (links) with their similarity

Conjecture: The higher the similarity the higher the likelihood that the two artefacts should be traced each other

Methods have to be used to cut the ranked list and retrieve only the top links in the list

The simplest methods do not take into account the similarity values

Fixed cut point: a fixed number of top links are retrieved

Ranked list percentage: a percentage of links in the ranked list are retrieved

The second method takes into account variability in the size of source and target artefact set

Cutting the Ranked List (2)

Other methods use a threshold on the similarity values

Constant threshold: the links with similarity above the threshold are returned

Scale threshold: the threshold is a percentage of the best similarity value between two artefacts, i.e.,

$$t = c \cdot \text{MaxSimilarity}, \text{ where } 0 \leq c \leq 1 \text{ and } \text{MaxSimilarity} > 0$$

Variable threshold: Constant threshold projected onto the interval [min similarity, max similarity] (instead of [-1, +1])

The first method does not consider variability in the verbosity of software artefacts

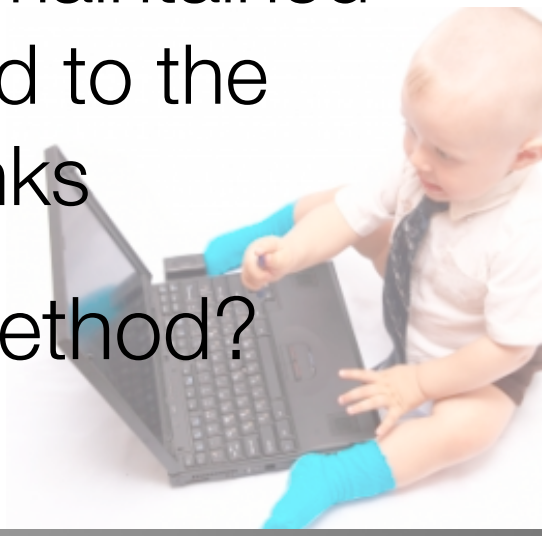
IR-based Traceability Recovery: 1st Limitation

The process is **semi-automatic**

The list of candidate links has to be analysed by the software engineer that traces the correct links and discards false positives (i.e., artefacts having high texual similarity but that are actually not related)

Knoweledge based traceability is usually maintained explictly in order to save the effort required to the software engineer to classify candidate links

How accurate is a traceability recovery method?



Evaluation of IR-based Traceability Recovery

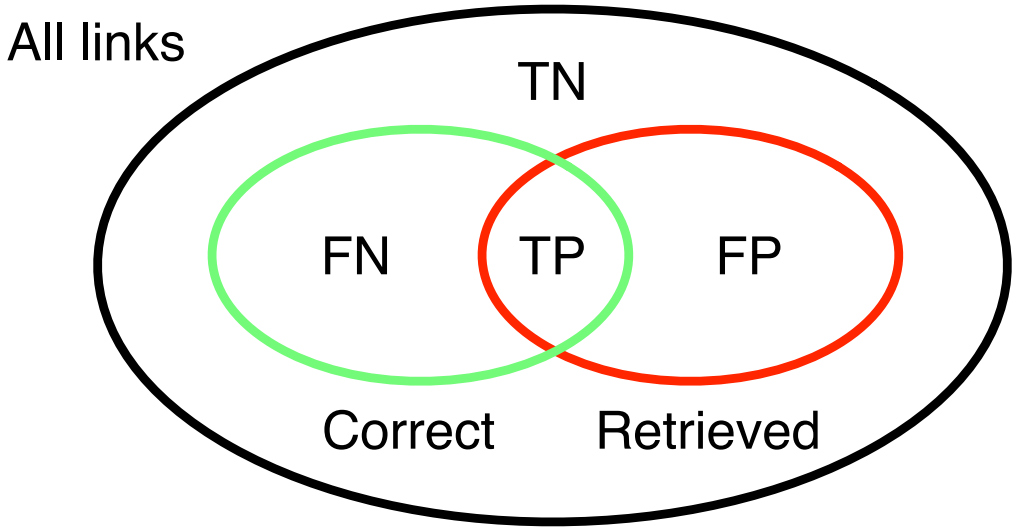
Pre-condition: The "correct" traceability matrix is available

- Such a matrix contains all the relationships between artefacts
- It has been defined manually by the original developers

Comparing the ranked list of candidate links with the traceability matrix is possible to define

- **Precision**: fraction of retrieved artefacts that are relevant
- **Recall**: fraction of relevant artefacts that are retrieved

Recall and Precision (1)



TN = True negative
FN = False negative
TP = True positive
FP = False positive

$$\text{Recall} = TP / (TP + FN)$$

$$\text{Precision} = TP / (TP + FP)$$

Recall and Precision (2)

You can get high recall (but low precision) by retrieving all target artefacts for all source artefacts!

- The similarity threshold is set to zero in this case
- No support at all!

Recall is a non-decreasing function of the number of artefact retrieved

Usually, precision decreases as either the number of artefacts retrieved or recall increases

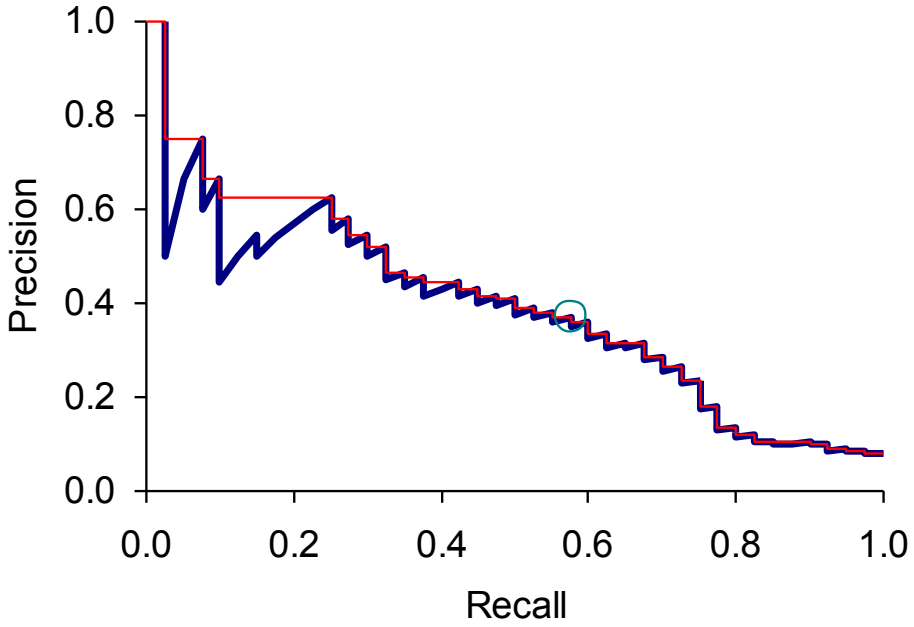
- Not a theorem, but a result with strong empirical confirmation

Recall/Precision Graph

Recall and precision depend by the threshold used to cut the ranked list

By taking various numbers of the top returned artefacts (levels of recall), it is possible to produce a precision-recall curve

Usually, precision is measured at 11 levels of recall varying from 0 to 1



A Combined Measure

Combined measure that assesses precision/recall tradeoff is F measure (weighted harmonic mean):

$$F_\beta = \frac{(1+\beta^2)\cdot(precision\cdot recall)}{\beta^2\cdot(precision+recall)}$$

People usually use balanced F1 measure

$$F = \frac{2\cdot(precision\cdot recall)}{(precision+recall)}$$

where recall and precision are evenly weighted

Yet more evaluation measures…

Mean average precision (MAP)

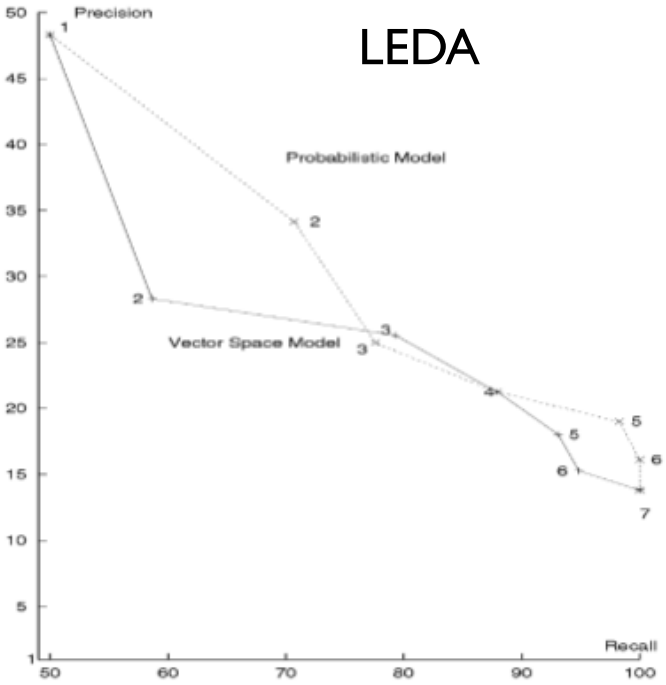
Average of the precision value obtained for the top *k* artefacts, each time a relevant artefact is retrieved

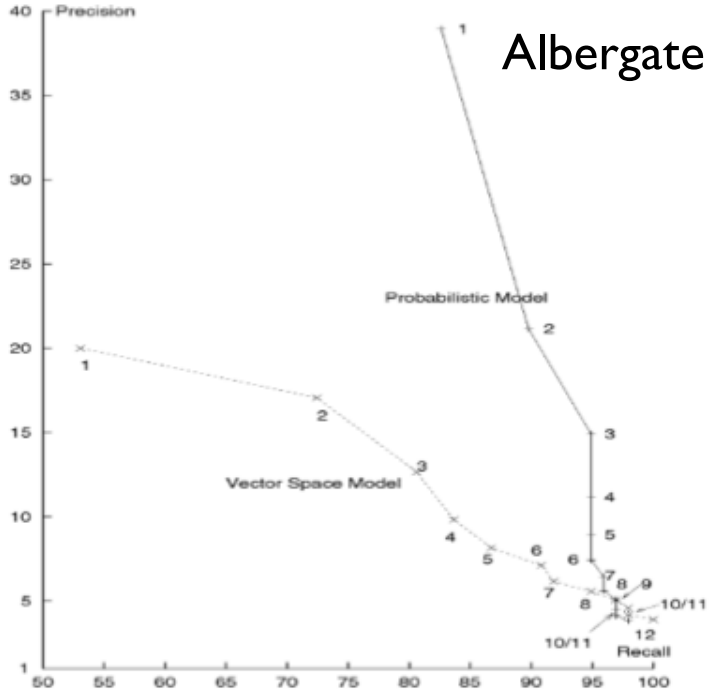
Statistical analysis

Statistical test can be used to test the following null hypothesis: "*The false positives retrieved by method M1 are comparable to false positives retrived by method M2*"

The false positives retrieved for each correct link is considered as dependent variable

IR-based Traceability Recovery: 1st evaluation





Compared VSM and probabilistic model
Study conducted on LEDA and Albergate
The two approaches exhibits almost the same accuracy

Giuliano Antoniol, Gerardo Canfora, Gerardo Casazza, Andrea De Lucia, Ettore Merlo: Recovering Traceability Links between Code and Documentation. IEEE Trans. Software Eng. 28(10): 970-983 (2002)

IR-based Traceability Recovery: 2nd Limitation

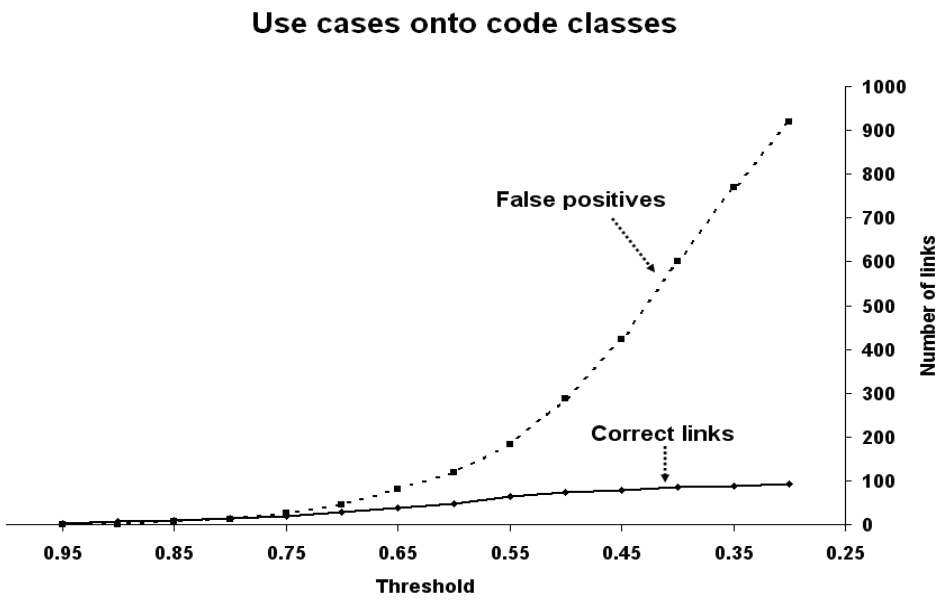
Recovering all correct links is in general impractical!

A “simple” strategy: use a very low threshold

- Low threshold => High #candidate links retrieved
- High #candidate links retrieved => High recall
- BUT
- High #links retrieved => High #false positives
- High #false positives => Low precision

Density of correct links

Trend of False Positives and Correct Links



To trace all (93) correct links the software engineer has to set 0.3 as similarity threshold. With such a threshold 1013 links are retrieved where 990 are false positives. 1110 is the number of possible links!

Andrea De Lucia, Fausto Fasano, Rocco Oliveto, Genoveffa Tortora: Recovering traceability links in software artifact management systems using information retrieval methods. ACM Trans. Softw. Eng. Methodol. 16(4): (2007)

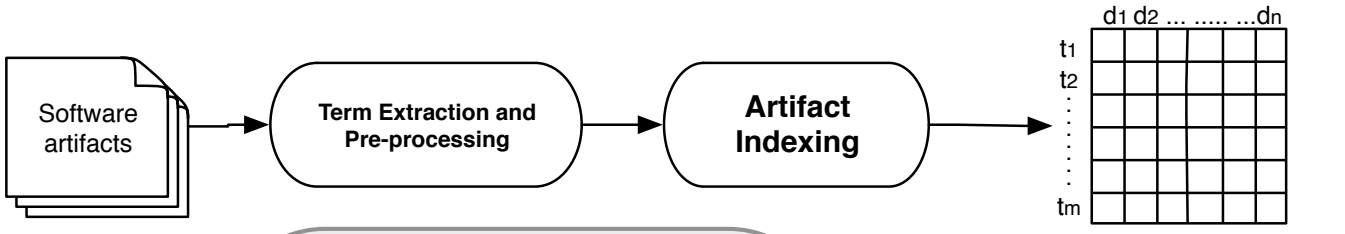
We need a solution!

Low accuracy makes the recovery process tedious, time consuming and error prone!

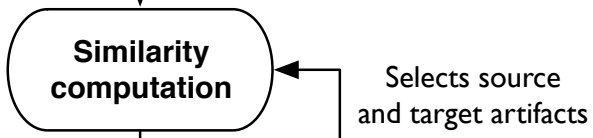
> A long term vision for the traceability research community is that of "one click tracing." This goal envisions a future where software engineers are developing and maintaining traceability information as they perform their normal duties, with traceability being captured and updated as a by-product of their work and completely unbeknownst to them (or perhaps as a result of "one click" of the mouse).
>
> Hayes et al., TEFSE 2009

Enhancing strategies

Key-phrases and
thesaurus
Project glossary
Grammatical analysis

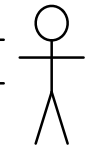


Smoothing filters
IR methods



Source	Target	Similarity	Correct
S1	T2	93.78%	☑
S1	T3	90.67%	☐
...	...	...	☐
S12	T23	5.87%	☐

Candidate links



User feedback analysis

Key-phrases and thesaurus (1)

Key-phrases

Phrases are extracted besides keywords. Phrases are identified analysing a list of technical terms

Example: “ecs production environment” and “ecs archive metadata”

Thesaurus

Thesaurus is used to mitigate the synonym problem

Jane Huffman Hayes, Alex Dekhtyar, James Osborne: Improving Requirements Tracing via Information Retrieval. RE 2003: 138-147

Key-phrases and thesaurus (2)

Formal definition of thesaurus

Each entry is a triple (k_i,k_j,a_{ij}), where k_i and k_j are vocabulary terms and $a_{ij} \in [0,1]$ is a perceived similarity coefficient of k_i and k_j

How to build a thesaurus

Keywords (or phrases) are extracted from the artefacts

For each pair (k_i,k_j) the software engineer defines a perceived similarity coefficient

Key-phrases and thesaurus (3)

	SuperTracePlus Tool	Analyst	Retrieval with thesaurus algorithm
Correct links	41	41	41
Correct links found	26	18	**35**
Total number of candidates	67	39	86
Missed requirements	**3**	6	4
Average recall	69.37%	53.30%	**71.69%**
Average precision	**56.48%**	53.55%	32.76%
Overall recall	63.41%	43.9%	**85.36%**
Overall precision	38.80%	**46.15%**	40.69%

The retrieval with key-phrases algorithm resulted in improved recall but with decreased precision

The retrieval with thesaurus algorithm outperforms the standard method and the key-phrases based method in terms of recall and sometimes - in terms of precision

The senior analyst with 20 years of experience spent 4 hours to trace the links, while the thesaurus was built in 20 minutes

Term Coverage, Phrasing, and project glossary

Term Coverage

Increase the similarity between artefacts that share two or more distinct terms

Phrasing

Index phrases (and not only single terms) that co-occur in pairs of artefacts

Two-word phrases (defined as sequences of two nouns) were considered

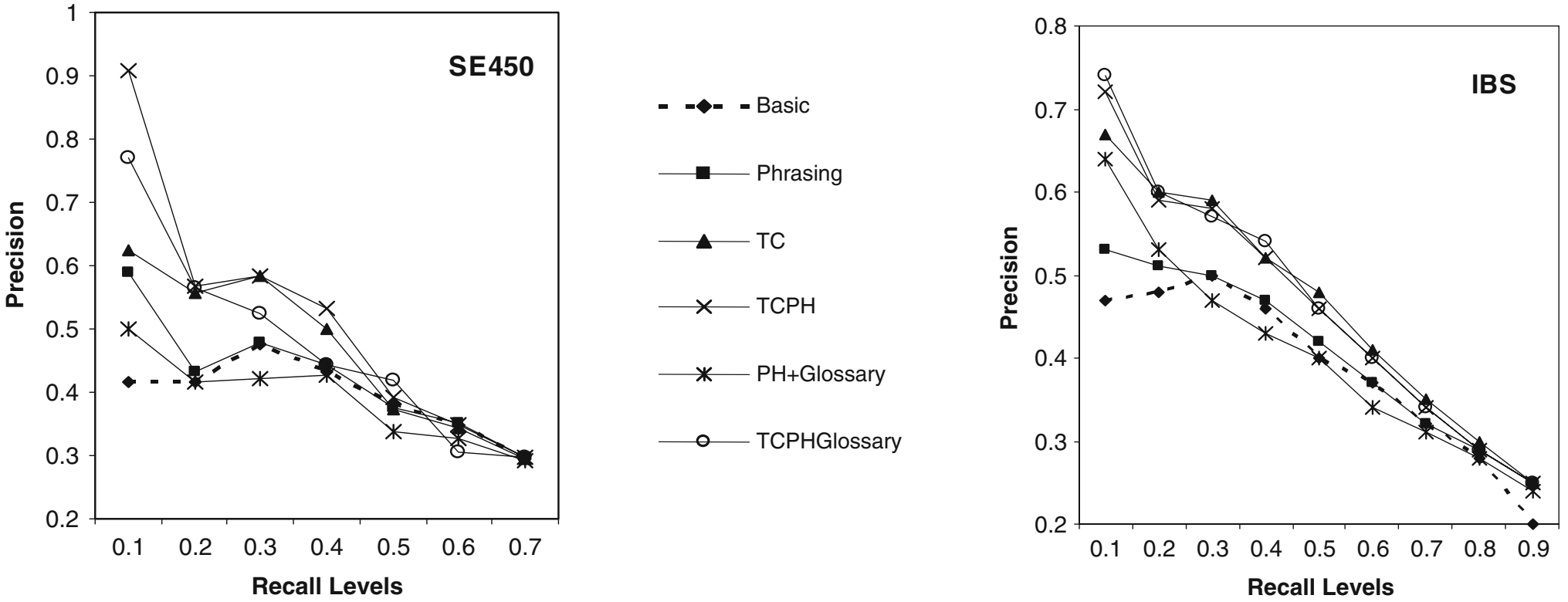
Project glossary

Increase the similarity between artifacts *q* and *d* that share either terms or phrases defined in the project glossary

Terms in project glossary are really keywords since they have the ability to capture more critical concepts for a given project

Xuchang Zou, Raffaella Settimi, Jane Cleland-Huang: Improving automated requirements trace retrieval: a study of term-based enhancement methods. Empirical Software Engineering 15(2): 119-146 (2010)

Term Coverage, Phrasing, and project glossary



The synergistic application of TC and phrasing together is shown to be generally the most effective (see SE450)

When a meaningful project glossary (see IBS) is available, the synergistic application of the TC, phrasing and project glossary methods yields the highest increase in precision

The role of nouns (1)

UC–MOD–LAB - Insert a new laboratory

DESCRIPTION: This functionality allows the Operator to modify the data of a chemical laboratory.
PRIORITY: Low.
DURATION: Minutes.
INITIALISING: Operator.
ACTOR: He/she wants to add a new laboratory in the Hospital Informative System (H.I.S.)
PRE-CONDITIONS: The Operator has been admitted by the System (Cfr. UC–VAL–OP) and he/she has the data of the new laboratory.
GUARANTIES:
Failure: the data in the H.I.S. is not modified.
Success: the new laboratory is correctly stored in the H.I.S.
START:The System shows the list of all the laboratory and the Operator activates the functionality to add a new laboratory.

Main Scenario
1. System shows a form to insert the data of a new laboratory.
2. Operators fills-in the form, inserting name, VAT number, address, city, province, postal code, telephone numbers, and additional information.
3. Operators submits the data.
4. System verifies the data inserted by the Operator.
5. System stores the data.
6. System notifies that the operation was correctly completed.

Alternative scenario: Data not valid
4.1. System shows a message highlighting the incorrect data.
4.2. System recovers the execution from point 1 setting the form fields with the data inserted by Operator.

Error scenario: The Operator aborts the functionality
3.1. System stop the execution of the use case with no success.

The role of nouns (2)

UC–MOD–LAB - Insert a new **laboratory**

DESCRIPTION: This **functionality** allows the **Operator** to modify the **data** of a chemical **laboratory.**
PRIORITY: Low.
DURATION: Minutes.
INITIALISING: **Operator.**
ACTOR: He/she wants to add a new **laboratory** in the **Hospital Informative System** (H.I.S.)
PRE-CONDITIONS: The **Operator** has been admitted by the **System** (Cfr. UC–VAL–OP) and he/she has the **data** of the new **laboratory.**
GUARANTIES:
Failure: the **data** in the H.I.S. is not modified.
Success: the new **laboratory** is correctly stored in the H.I.S.
START:The **System** shows the **list** of all the **laboratory** and the **Operator** activates the **functionality** to add a new **laboratory.**

Main Scenario
1. **System** shows a **form** to **insert** the **data** of a new **laboratory.**
2. **Operators** fills-in the **form**, inserting **name, VAT number, address, city, province, postal code, telephone numbers, and additional information.**
3. **Operators** submits the **data.**
4. **System** verifies the **data** inserted by the **Operator.**
5. **System** stores the **data.**
6. **System** notifies that the **operation** was correctly completed**.**

Alternative scenario: **Data** not valid
4.1. **System** shows a **message** highlighting the incorrect **data.**
4.2. **System** recovers the **execution** from **point** 1 setting the **form fields** with the **data** inserted by **Operator.**

Error scenario: The **Operator** aborts the functionality
3.1. **System** stop the **execution** of the **use case** with no **success.**

The role of nouns (3)

The language used in software documents can be classified as a **sectorial** language

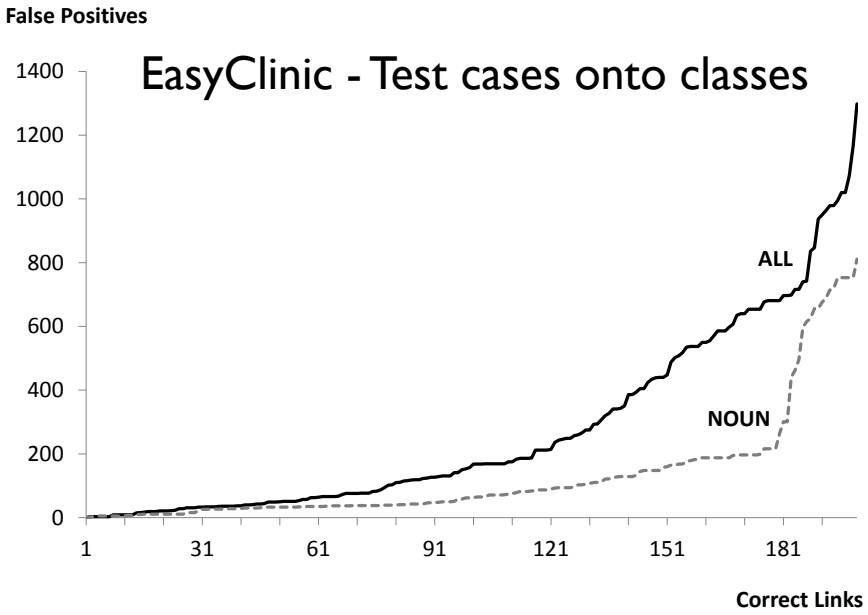
Here nouns are crucial, while the verbs tend to play a connection role and have a generic semantics

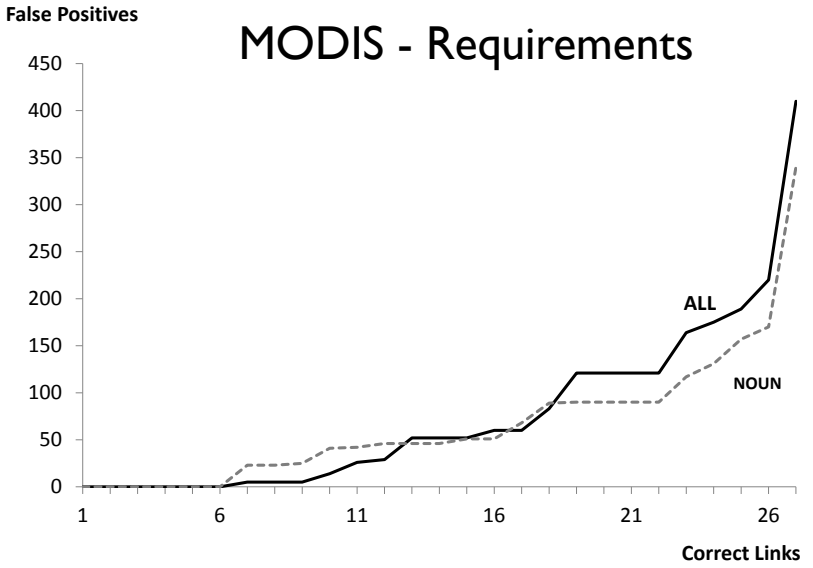
Only nouns are extracted, while other terms are pruned-out

A Part-of-speach tagger is used to derive the grammatical nature of terms

Giovanni Capobianco, Andrea De Lucia, Rocco Oliveto, Annibale Panichella, Sebastiano Panichella: On the role of the nouns in IR-based traceability recovery. ICPC 2009: 148-157
G. Capobianco, A. De Lucia, R. Oliveto, A. Panichella, S. Panichella. Improving IR-based Traceability Recovery via Noun-based Indexing of Software Artifacts. Journal of Software Maintenance and Evolution: Research and Practice, 25(7):743-762, 2013

The role of nouns (4)





The approach improves the retrieval accuracy of both probabilistic and vector space based models

When recall is 80% there are several cases where it is possible to achieve an improvement of precision of about 20%

When tracing requirements of MODIS the approach does not provide any improvement at all

Limitations

It is not always easy to identify key-phrases

The glossary is not always available

The use of a thesaurus is expensive

POS tagger could be not accurate

Noise in Software Artefacts

Artefacts contain noise!

- Linguistic forms specific of the language used in the software artifacts
  - Examples: use cases can contain recurring terms related to the use case template, while source code can contain programming language key-words or library function/method names.
- Common words that have low information content

How to remove the noise?

Noise can be removed with a stop word list

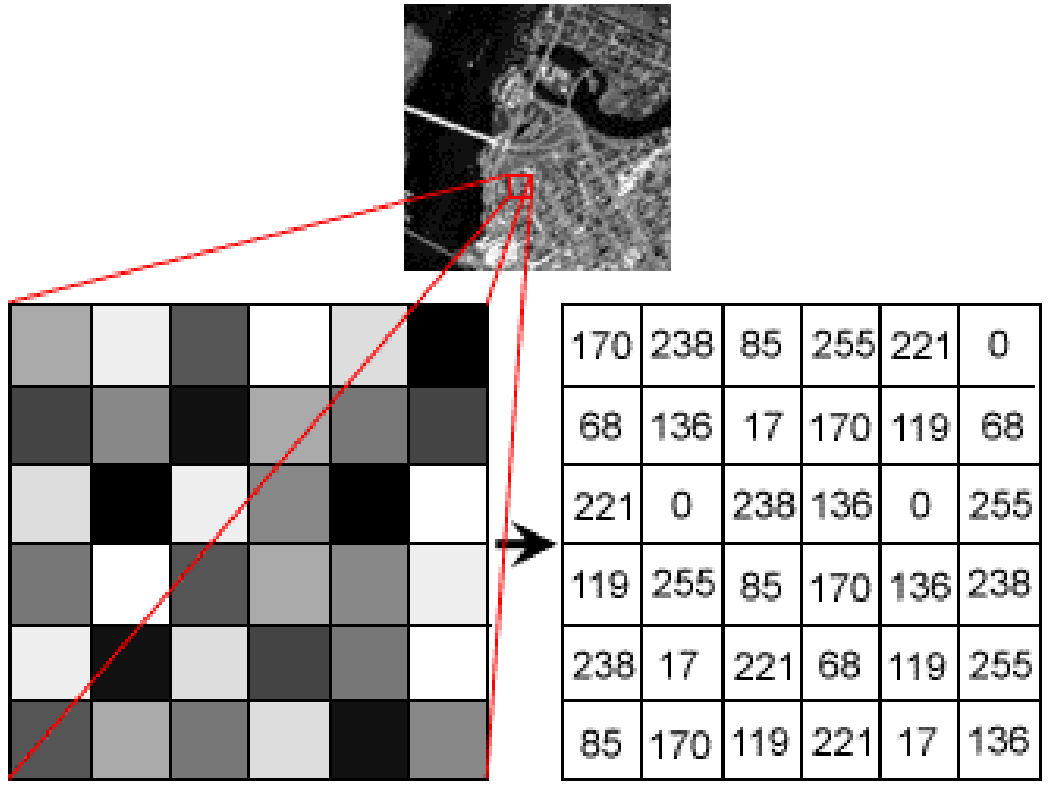
But...

Template variability: a development team often uses templates to produce software documentation

Developer subjectivity: each team member has a unique way of processing and writing a software document with respect to her own personal vocabulary

Ad-hoc stop word lists should be defined

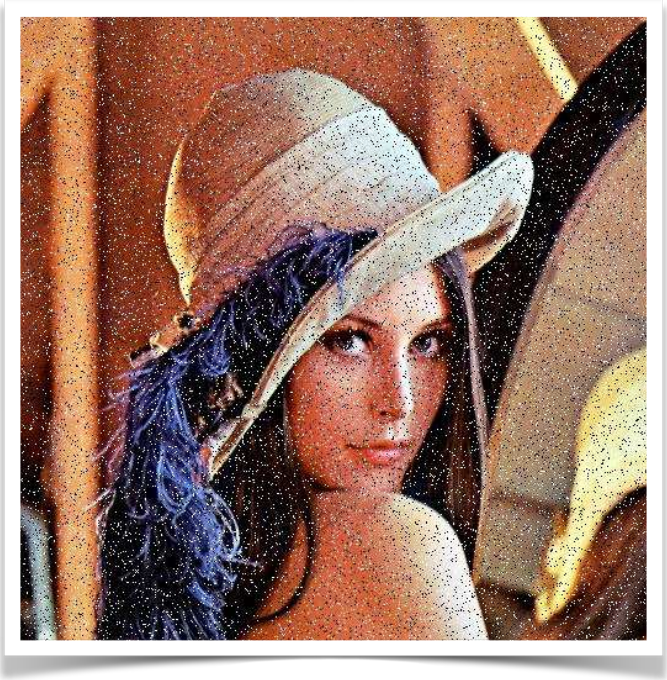
A similar problem: image processing

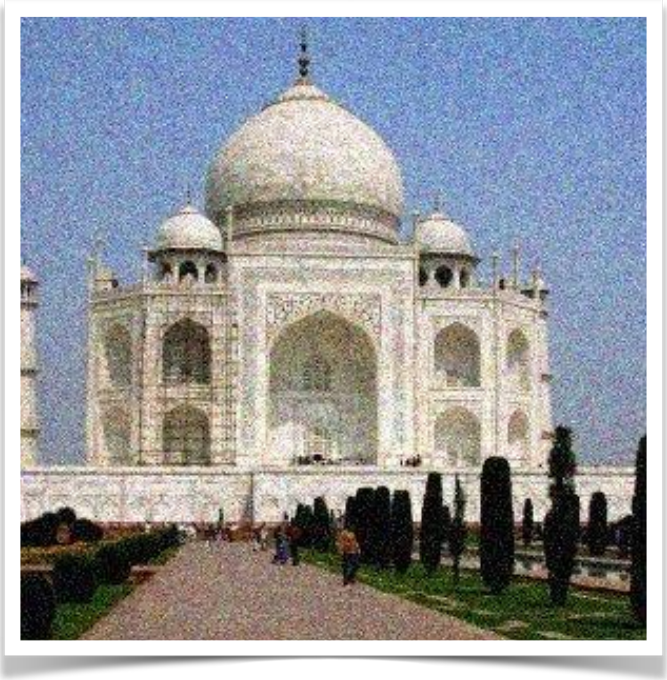


170	238	85	255	221	0
68	136	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

Andrea De Lucia, Massimiliano Di Penta, Rocco Oliveto, Annibale Panichella, Sebastiano Panichella: Improving IR-based Traceability Recovery Using Smoothing Filters. ICPC 2011: 21-30
Andrea De Lucia, Massimiliano Di Penta, Rocco Oliveto, Annibale Panichella, Sebastiano Panichella: Applying a smoothing filter to improve IR-based traceability recovery processes: An empirical investigation. Information and Software Technology, 55(4):741-754, 2013.

Noisy Images



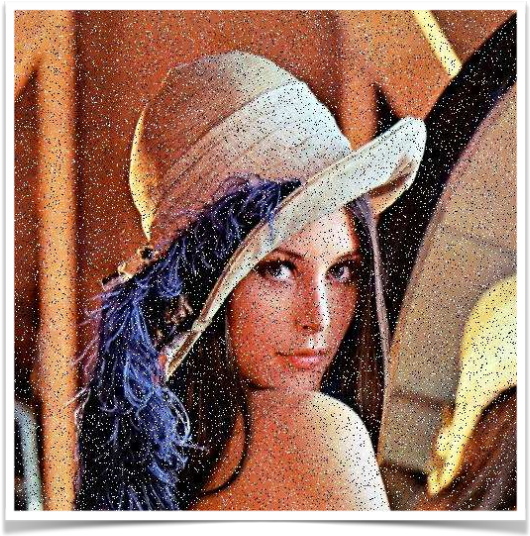


Pixels with peaks of low color intensity

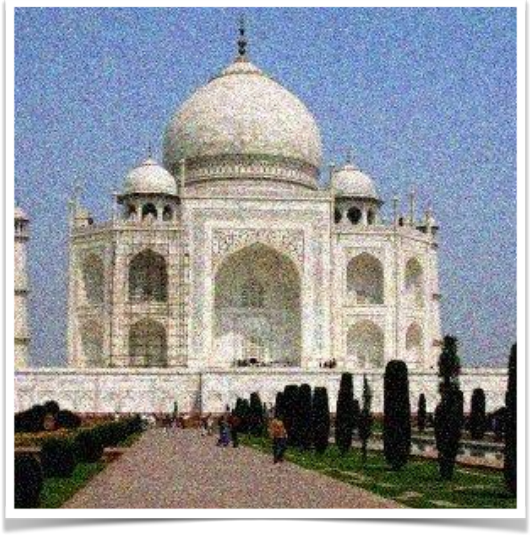
Pixels with peaks of high color intensity

Noise

Smoothing Filters



$$g(x,y) = \frac{1}{M} \sum_{f(n,m) \in S} f(n,m)$$



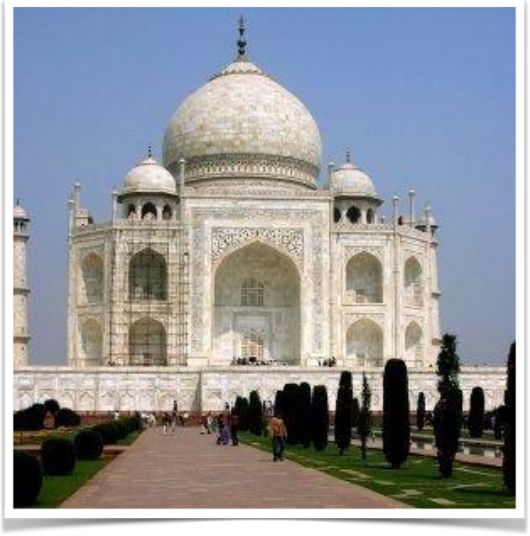
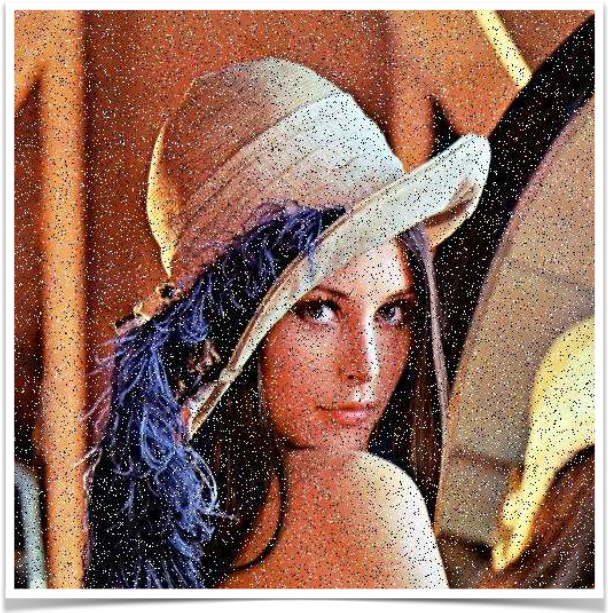
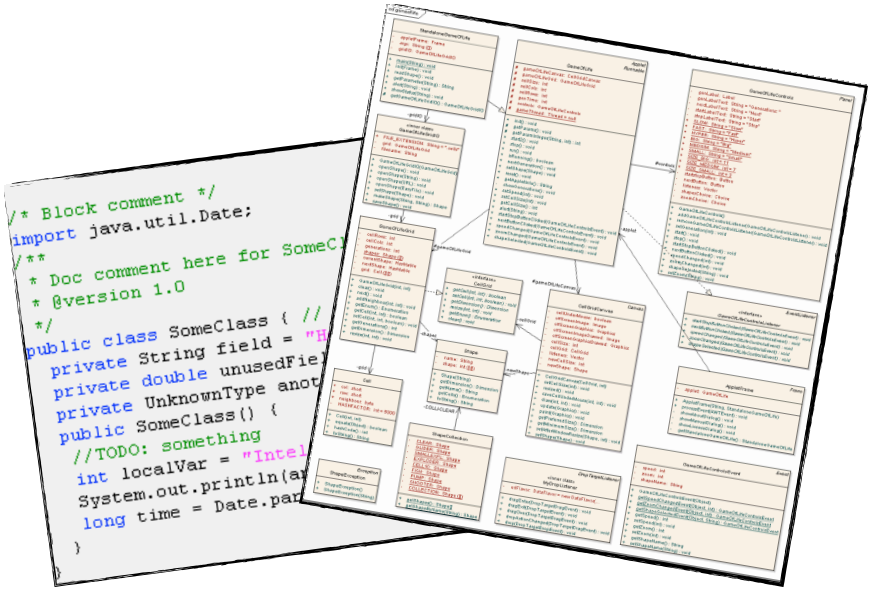


Image vs Software Artefact Noise



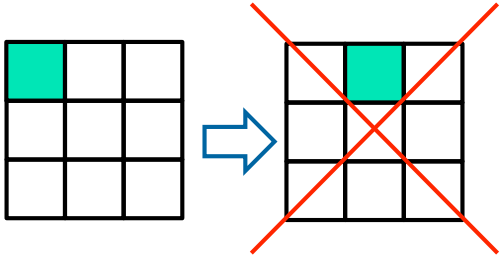


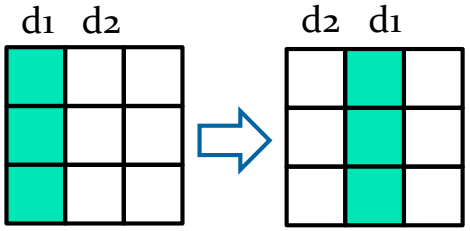
Pixels with high or low color intensity

Pixels are position dependent

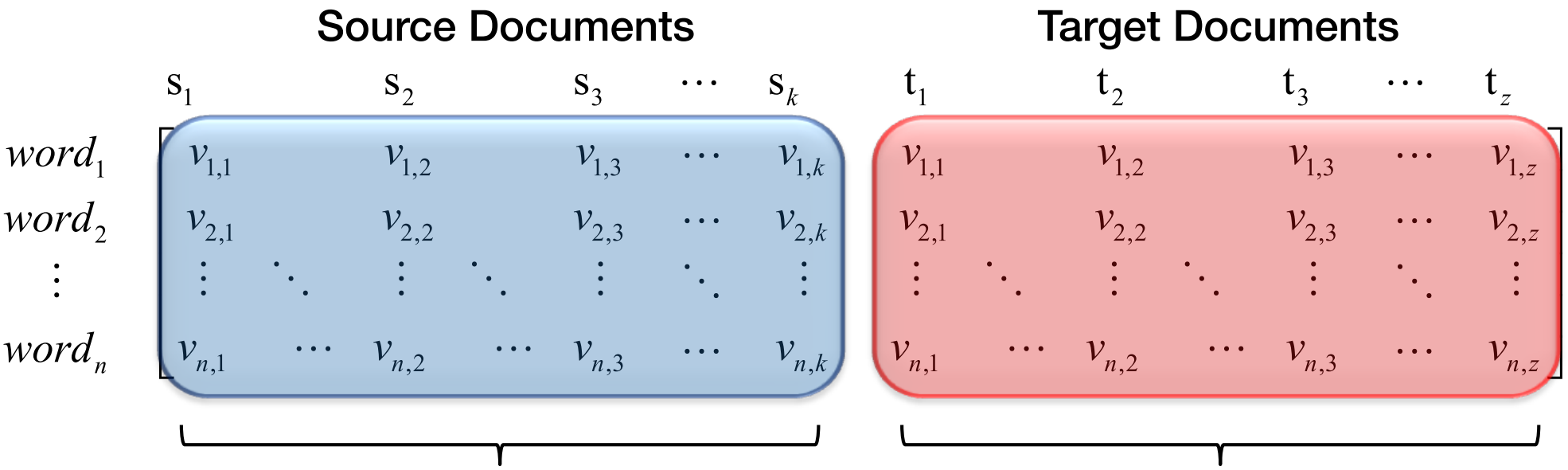
Terms and linguistic patterns occurring in many artefacts of a given category

Artefacts (columns) are position independent





Representing the Noise



Linguistic information strictly belonging to source artefacts

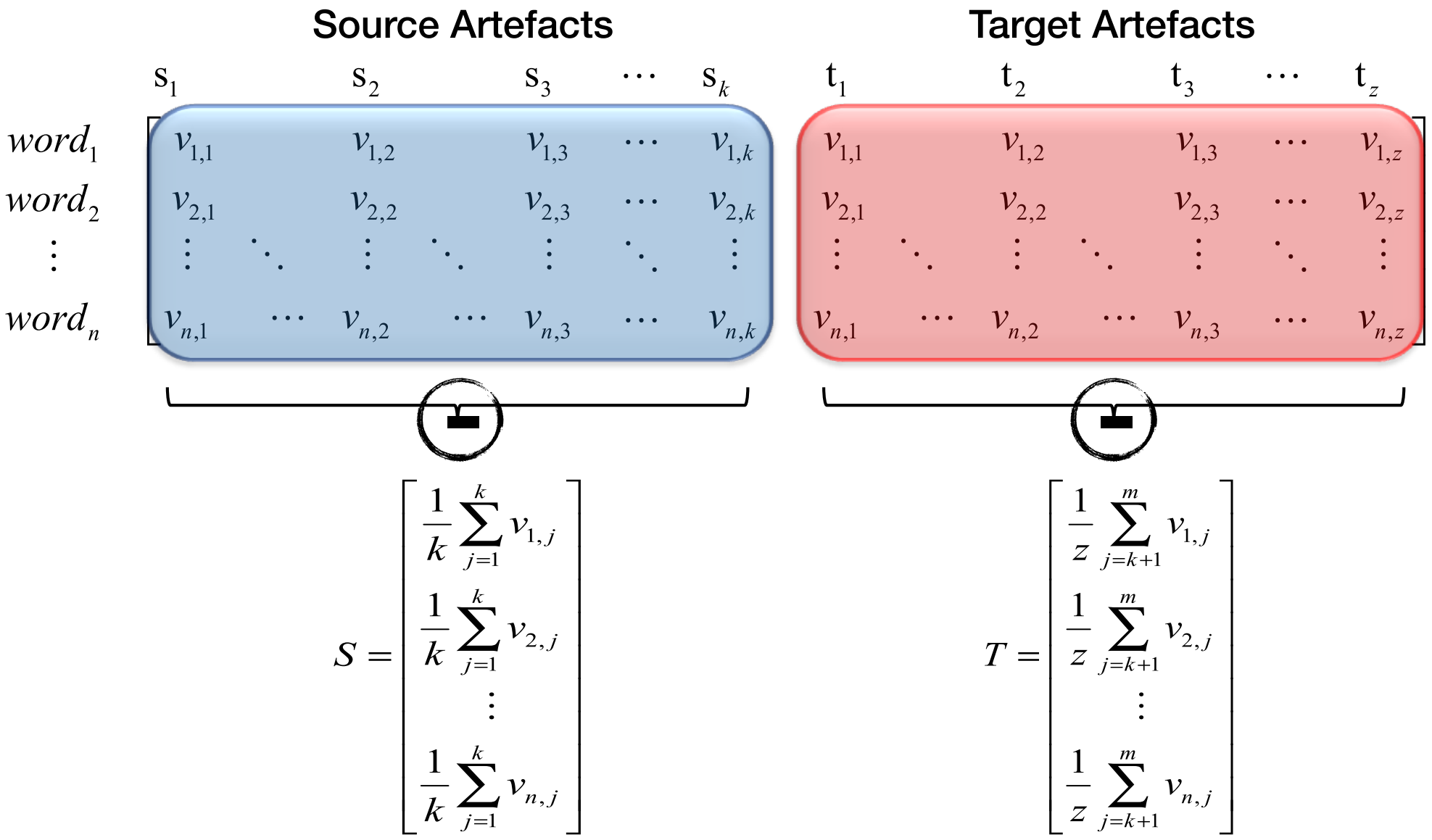
Linguistic information strictly belonging to target artefacts

Common information in source documents

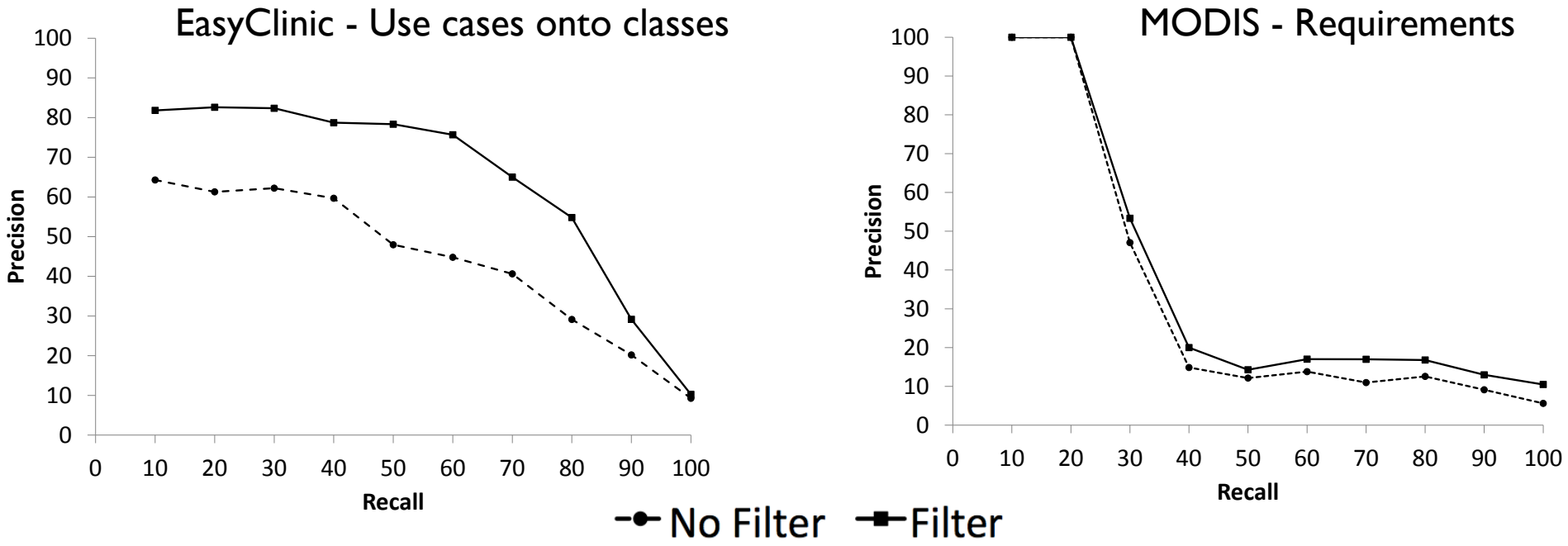
Common information in target documents

Noise

Removing the Noise



Smoothing Filters on Action



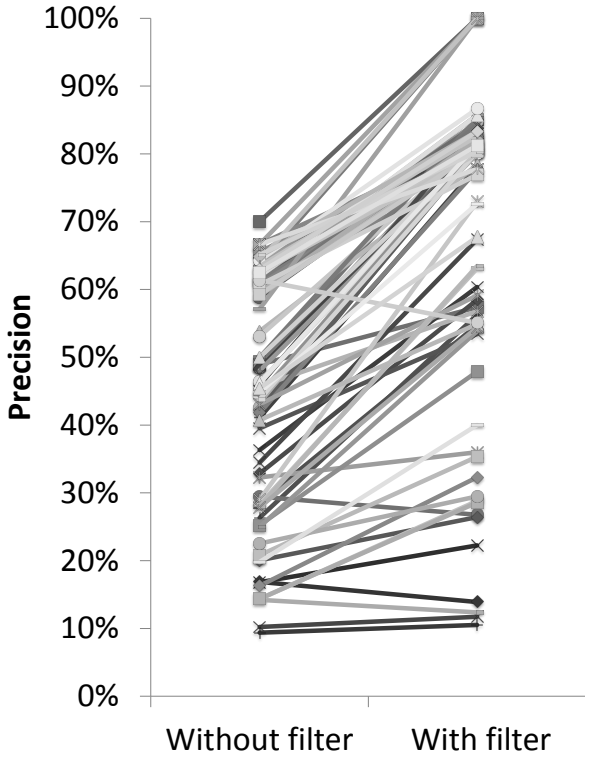
The filter significantly improves IR-based traceability recovery based on VSM, LSI and JS

Filter is particularly suitable for artefacts having a higher verbosity

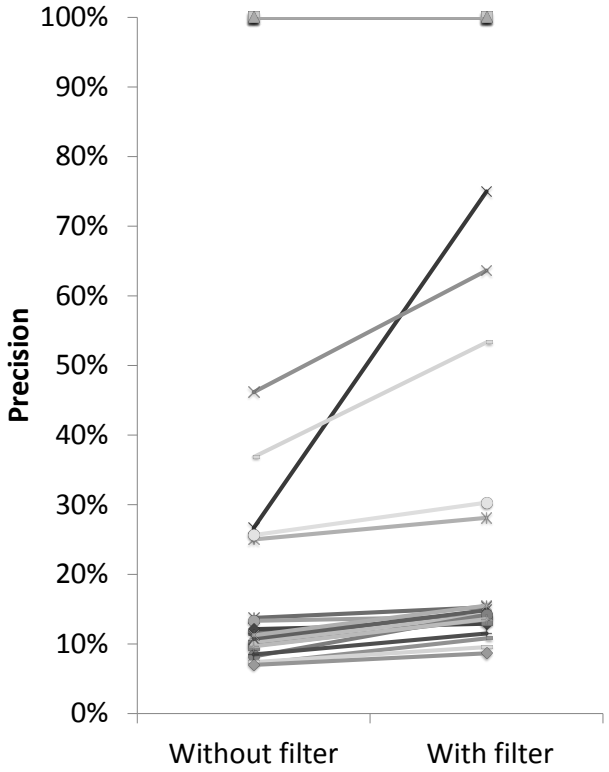
Less useful for artefacts composed of short sentences and using a limited vocabulary

A Practical View

EasyClinic - Use cases onto classes



MODIS - Requirements



Rank of correct links in the list of candidate links

Limitations of VSM and Probabilistic Model

Such models do not take into account relations between terms

They suffer of the synonymy and polysemy problems

synonymy: different words with the same meaning

polysemy: same words with different meanings (depending on the context)

For instance, having “automobile” in one artefact and “car” in another artefact does not contribute to the similarity measure between these two artefacts

Latent Semantic Indexing (LSI)

Extension of the Vector Space Model

- Provides a way to automatically deal with synonymy and polisemy
- Avoids preliminary morphological analysis

How does LSI mitigate the synonumy and polisemy problems?

- It analyses the co-occurrence of the terms by using the Singular Value Decomposition (SVD)

SVD is used to decompose the term-by-document matrix into a set of *k* orthogonal factors from which the original matrix can be approximated by linear combination

- The idea is to reduce the space of the terms
- Reducing the term space we also reduce the noice in the word usage caused by synonymy and polisemy words

Singular Value Decomposition

Let A be the term-by-document computed as well as in the VSM

$$A = T_0 \cdot S_0 \cdot D_0$$

where:

T_0 is the m x r matrix of the terms containing the left singular vectors (rows of the matrix)

D_0 is the r x n matrix of the documents containing the right singular vectors (columns of the matrix)

S_0 is an r x r diagonal matrix of singular values

r is the rank of A

SVD and Concept Clustering

SVD can be viewed as a technique for deriving a set of uncorrelated indexing factors or concepts

- their number is given by the rank r of A,
- their relevance is given by the singular values in S_0.

Concepts are a way to cluster related terms with respect to documents and related documents with respect to terms

The product S_0 x D_0 (T_0 x S_0, respectively) is a matrix whose columns (rows, respectively) are the document (term, respectively) vectors in the r-space of the concepts

The cosine of the angle between two vectors in this space represents the similarity of the two documents (terms, respectively) with respect to the concepts they share

Reduced Concept Space

SVD also allows a simple strategy for optimal approximate fit using smaller matrices

If the singular values in S_0 are ordered by size, the first k largest values may be kept and the remaining smaller ones set to zero

$$A \approx A_k = T \cdot S \cdot D$$

S is obtained from S_0 by deleting zero rows and columns

The truncated SVD captures most of the important underlying structure in the association of terms and documents, at the same time it removes the noise or variability in word usage

The choice of k is critical and the proper way to make such a choice is still an open issue

LSI *vs* Other IR Methods

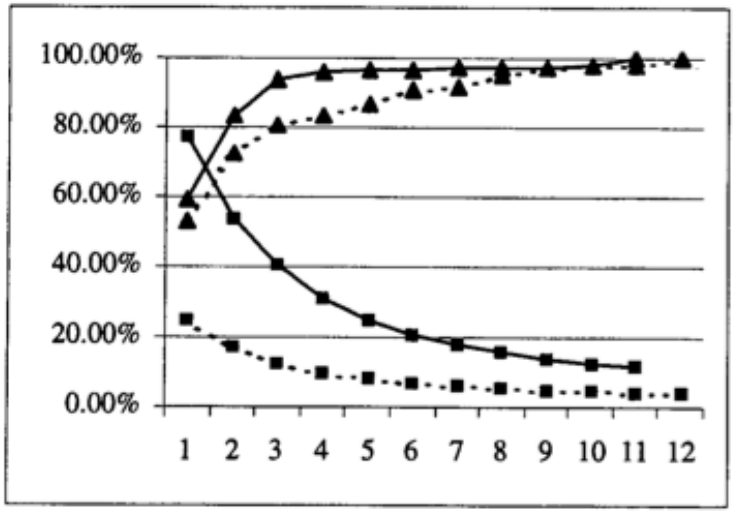


Figure 2. Recall and precision values for experiment by Antoniol and experiments with LSI using LEDA. x-axis represents the cut point and y-axis represents recall/precision values.

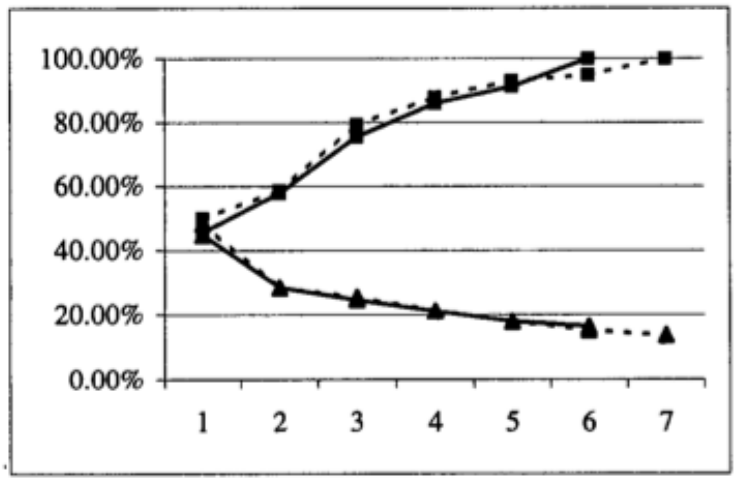


Figure 3. Recall and precision values for experiment by Antoniol and experiments with LSI using Albergate. x-axis represents the cut point and y-axis represents recall/precision values.

Compared LSI with VSM and probabilistic model
Study conducted on LEDA and Albergate
LSI seems to perform better than the others

Andrian Marcus, Jonathan I. Maletic: Recovering Documentation-to-Source-Code Traceability Links using Latent Semantic Indexing. ICSE 2003: 125-137

Limitations of the Employed IR Methods

Synonymy and polysemy problems

- VSM and Probabilistic models (in general) do not take into account relations between terms
- Example: “automobile” in one artefact and “car” in another artefact does not contribute to the similarity measure between the artefacts

Space reduction as a way to manage synonymy and polysemy

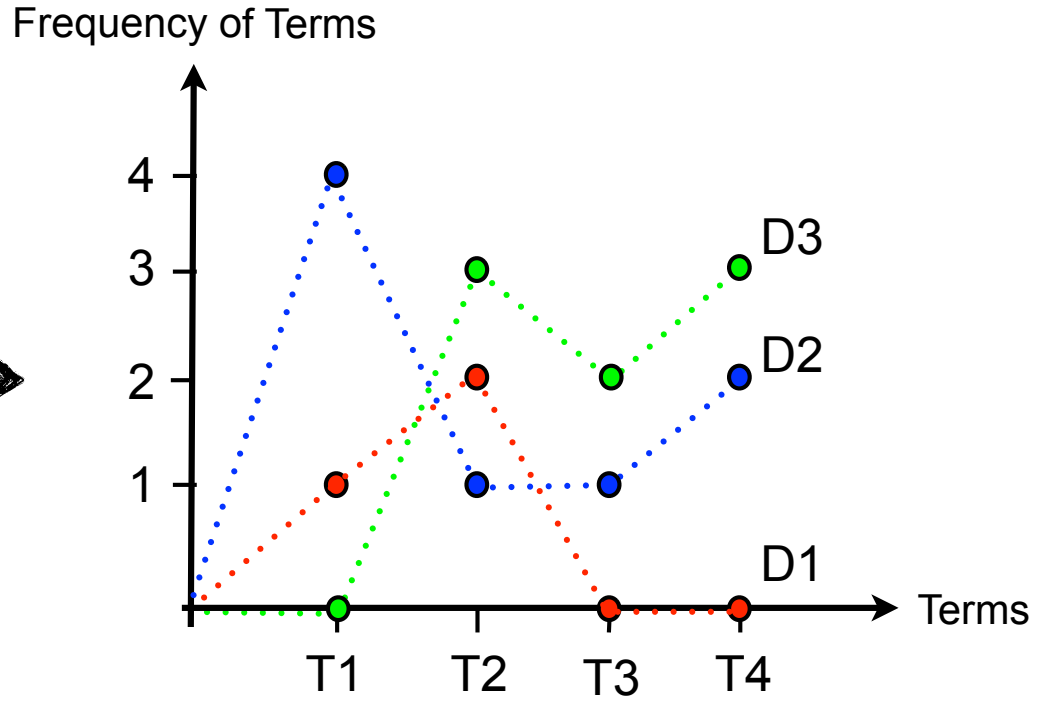
- LSI applies Singular Value Decomposition
- Software artefact repositories (thousands of artefacts) are small collection compared with canonical document collections (millions) used in IR
  - The reduction is not useful when the document collection is small
- Computing the space reduction is expensive

An Alternative Artefact Representation

Each artefact is represented by as a set of points in the Cartesian plane

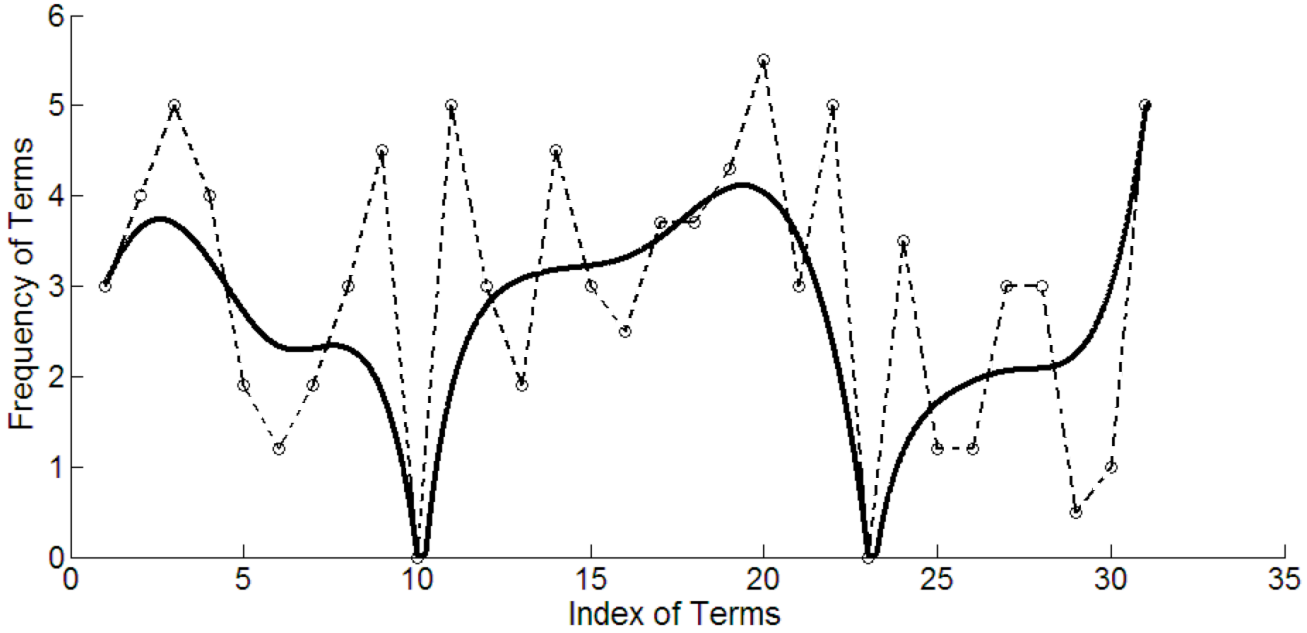
Term-by-document matrix

	D1	D2	D3
T1	1	4	0
T2	2	1	3
T3	0	1	2
T4	0	2	3



Giovanni Capobianco, Andrea De Lucia, Rocco Oliveto, Annibale Panichella, Sebastiano Panichella: Traceability Recovery Using Numerical Analysis. WCRE 2009: 195-204

Artefact Represenation via B-spline



Control points influence the shape of the curve but the curve did not interpolate the control points

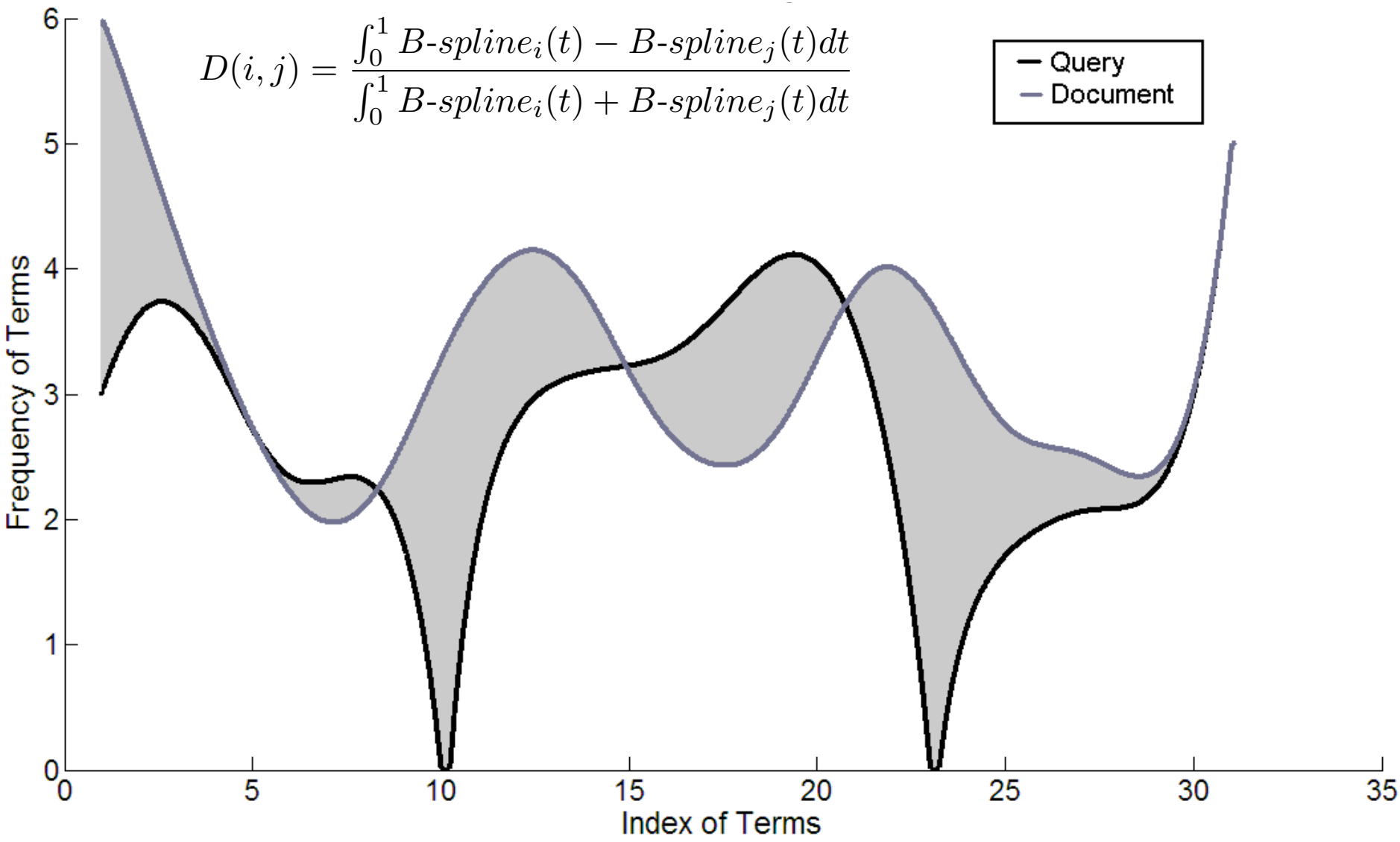
- Excepting the first and the last points

Possible to force the B-spline to interpolate a set of given points

- Points that represent terms having a weight equal to 0 are interpolated

Similarity Computation via B-spline

$$D(i,j) = \frac{\int_0^1 B\text{-}spline_i(t) - B\text{-}spline_j(t)dt}{\int_0^1 B\text{-}spline_i(t) + B\text{-}spline_j(t)dt}$$



Advanatages of B-spline

Fast and easy computation

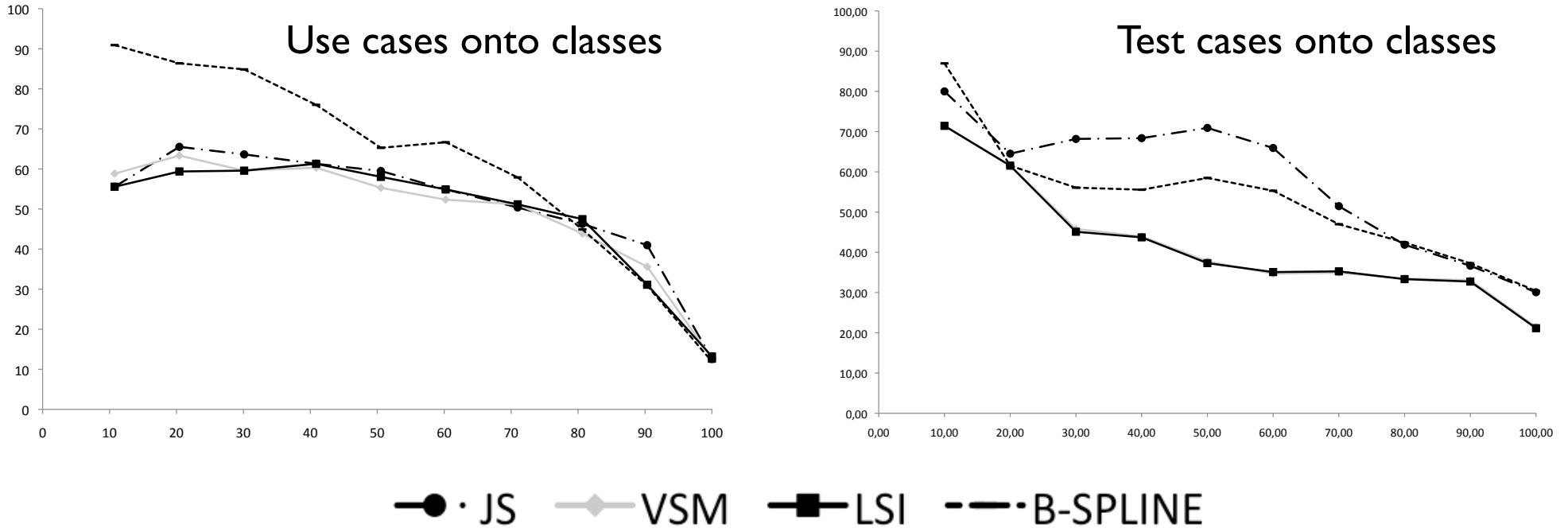
B-spline provides local control on the curve

The degree of the B-spline is independent of the number of control points (i.e., terms)

k = 30 is a good compromise between the computational complexity and the retrieval accuracy

Using a curve that interpolates the set of points representing an artefact allows us to take into account relations between terms (for mitigating the synonymy and polysemy problems)

B-splines vs Other IR Methods



When tracing use cases onto classes B-Spline is the best method when recall < 80%. When recall > 80% all methods provide similar accuracy

When tracing test cases onto classes, JS is the best method. B-spline overcomes VSM and LSI, while VSM and LSI achieve similar accuracy

Equivalence of IR Methods

JS, VSM, LSI and LDA were used to recover links between the documentation artefacts and source code of two systems

The ranked list provide by the different methods were compared using overlap metrics

The results indicate that while JS, VSM, LSI are equivalent, LDA is able to capture a dimension unique to the set of employed techniques

However, LDA-based traceability recovery technique provided lower accuracy as compared to other IR-based techniques

Rocco Oliveto, Malcom Gethers, Denys Poshyvanyk, Andrea De Lucia: On the Equivalence of Information Retrieval Methods for Automated Traceability Link Recovery. ICPC 2010: 68-71

Combining Different IR Methods (1)

No single method that sensibly outperforms the others

It has been empirically shown that some methods recover different, yet complementary traceability links

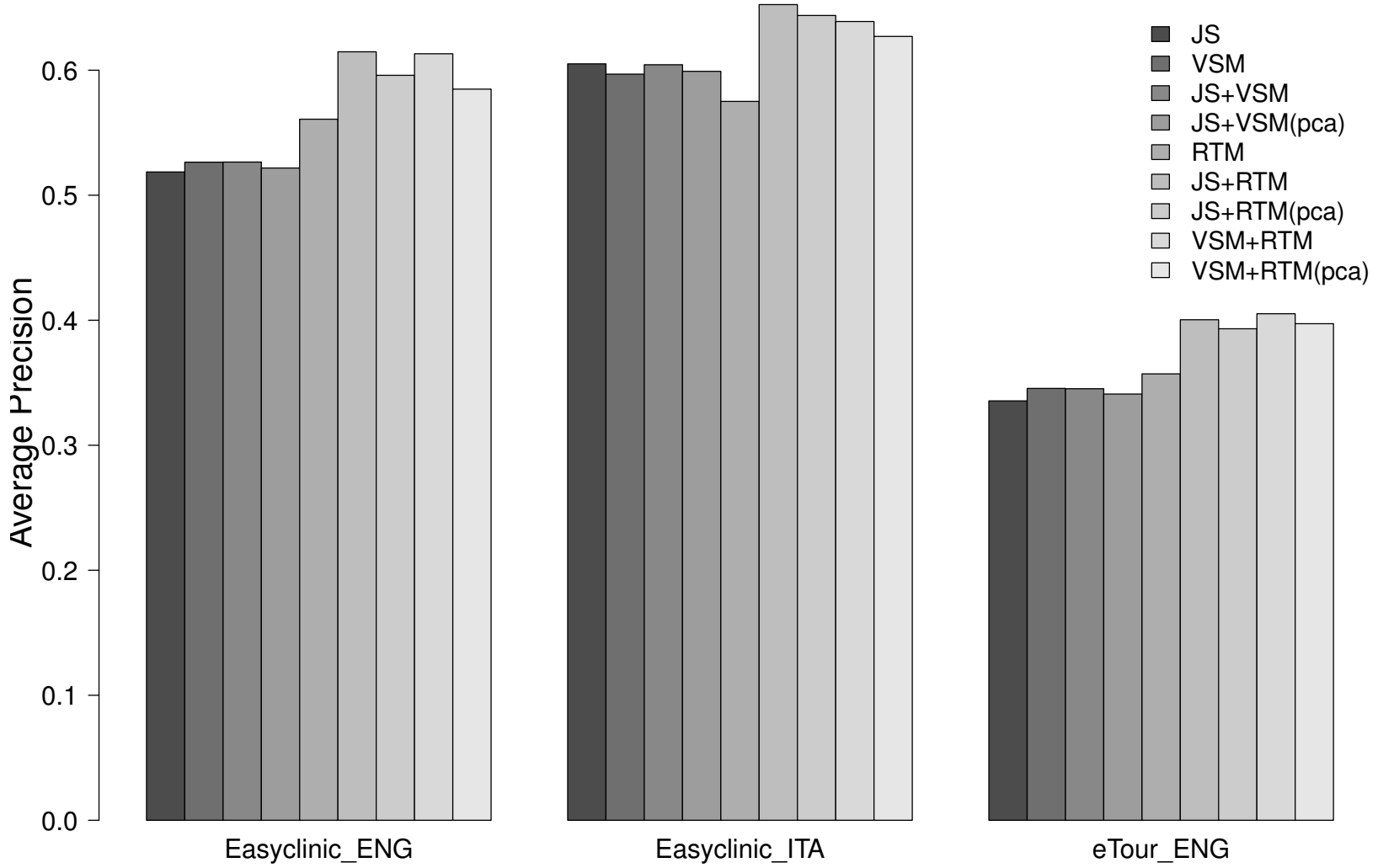
These results suggest to combine different IR methods

The similarity between artifacts is computed using different IR methods, and the final textual similarity is represented by the mean of these similarities

Vector Space Model (VSM), Jensen and Shannon (JS) model, and Relational Topic Modeling (RTM) were combined (only pairwise combination)

Malcom Gethers, Rocco Oliveto, Denys Poshyvanyk, Andrea De Lucia: On integrating orthogonal information retrieval methods to improve traceability recovery. ICSM 2011: 133-142

Combining Different IR Methods (2)



Combining Different IR Methods (3)

Summarising

RTM provides complementary ranked list as compared to other IR methods

The recovery accuracy can be improved by combining RTM with other canonical IR methods (e.g., JS or VSM)

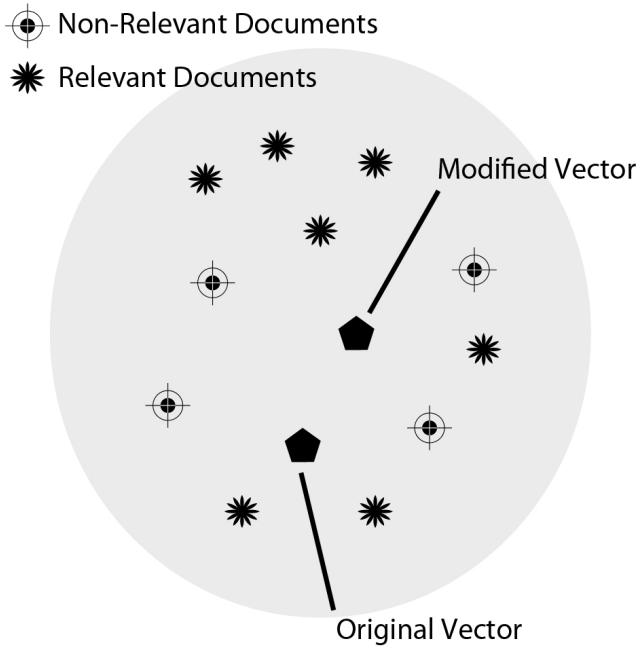
The hybrid method that combines complementary techniques outperforms (e.g., RTM + JS) stand-alone IR methods as well as any other combination of non-complementary methods with a statistically significant margin (e.g., VSM + JS)

User feedback analysis

User (or relevance) feedback: a link classified as either correct or false positive by the software engineer

This information can be used to train the IR-based traceability recovery tool

This will produce more accurate results at next usage



Feedback in VSM

Term re-weighting using Standard Rochio

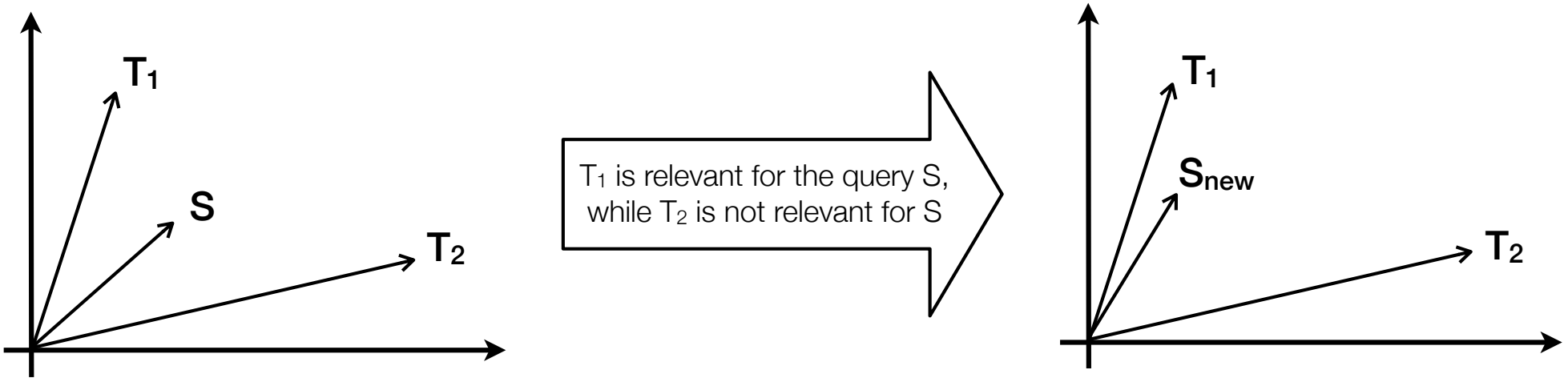
The user classifies some links (feedbacks)

Feedbacks are used to modify the source artefact vectors of classified links (query vectors)

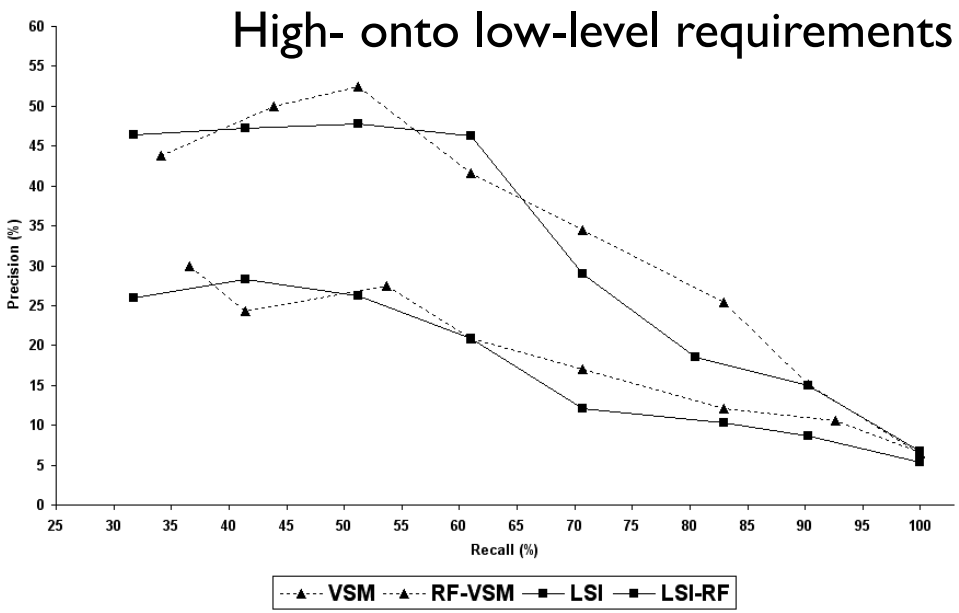
Each query vector is modified as follows

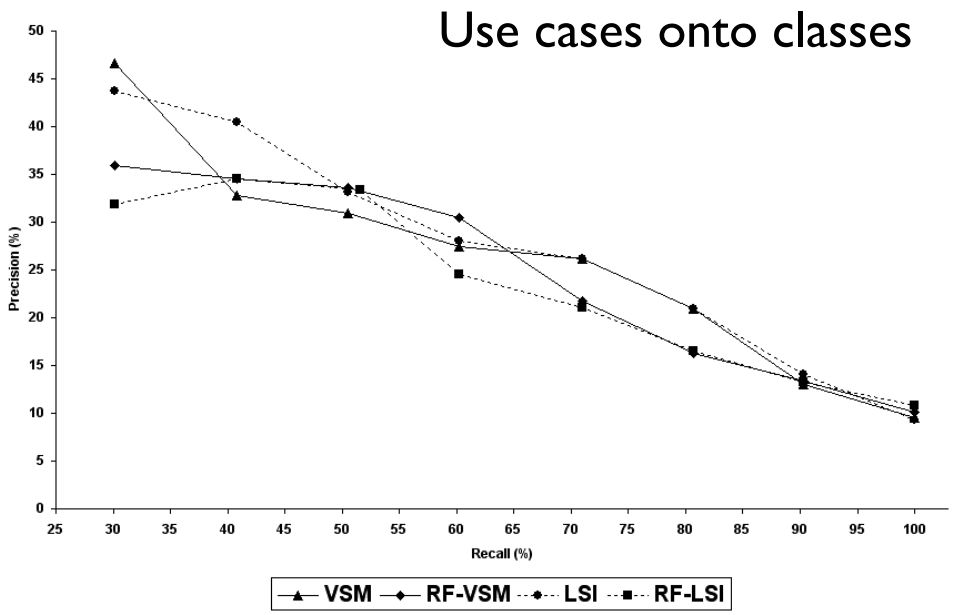
The terms shared with relevant target artefacts increase their weight in the new query vector

The terms shared with non relevant target artefacts decrease their weight in the new query vector



Feedback Analysis on Action





Feedback analysis is not a silver bullet for the problem of recovering all correct links

When the accuracy of the IR method is already good, feedback do not provide any improvement at all

Andrea De Lucia, Rocco Oliveto, Paola Sgueglia: Incremental Approach and User Feedbacks: a Silver Bullet for Traceability Recovery. ICSM 2006: 299-309

What works, what doesn't

IR engine		No clear winner Probably the esiest is the best
Advanced artefact indexing (key-phrases, project glossary, thesaurus)		Not easy to perform Not applicable to all projects
Indexing only the nouns		Could depend by the accuracy of POS tagger
Smoothing filter		Easy to apply Good improvement in term of accuracy
Combination of IR methods		Combination of orthogonal method is worthwhile

There is Still the “2nd Limitation”

Recovering all correct links is in general impractical!

None of these enhancing strategies is a silver bullet for the problem of recovering all correct links, as this still implies a high effort required to discard false positives

Are IR-based traceability recovery tools useful?

Evaluation of IR-based Recovery Tools

Evaluate the impact of the traceability recovery tool on the tracing performances of the software engineer (accuracy and time)

Context…

20 Master students (Exp I) and 12 Master students (Exp II) of the University of Salerno

Students were grouped in Low and High Ability

They had to perform two traceability recovery tasks on a repository of a completed software project

T1: recovering links between use cases and classes

T2: recovering links between UML diagrams and test cases

Tasks were performed in two separate laboratory sessions with or without ADAMS Re-Trace

Andrea De Lucia, Rocco Oliveto, Genoveffa Tortora: Assessing IR-based traceability recovery tools through controlled experiments. Empirical Software Engineering 14(1): 57-92 (2009)

Results of the Experiments

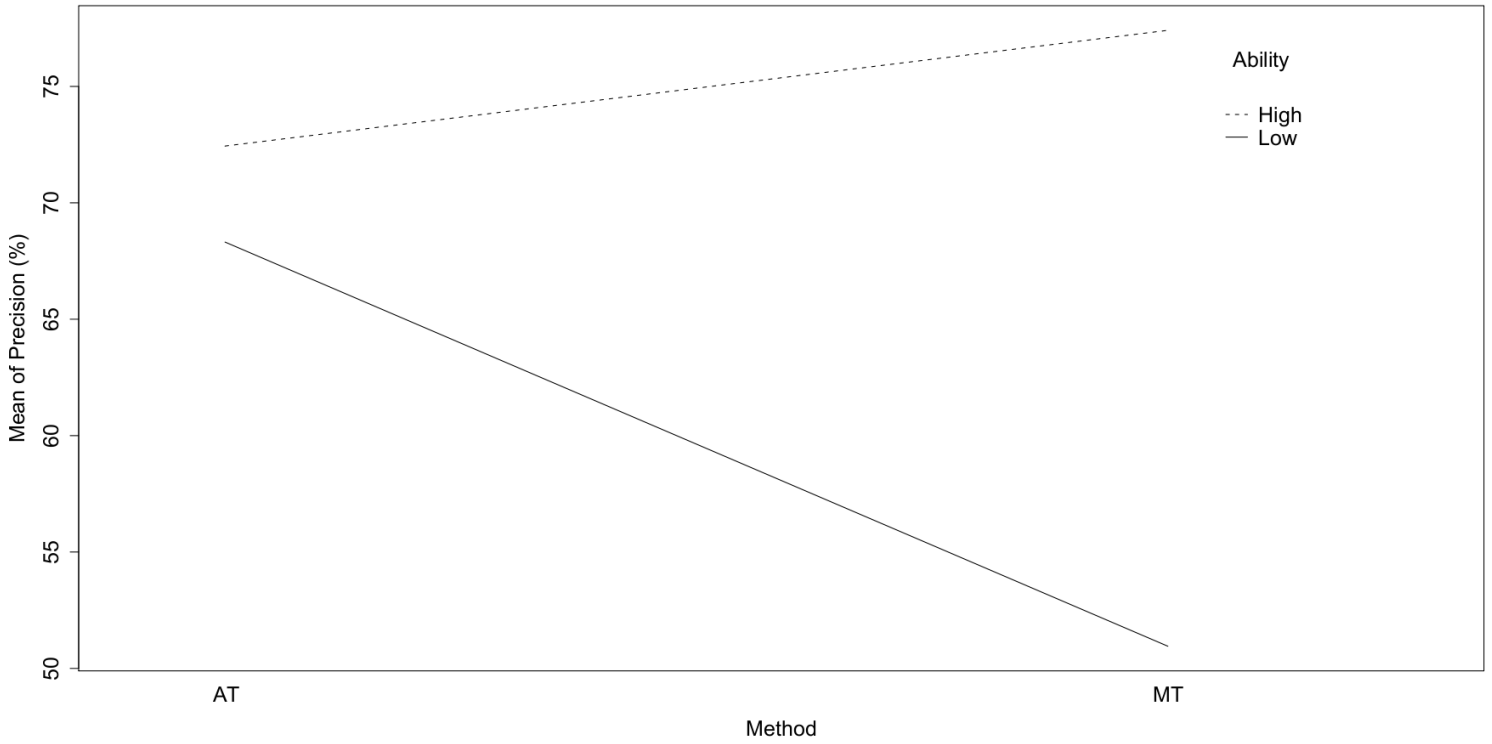
Table 6 Wilcoxon paired test *p*-values and descriptive statistics of differences (by subject)

	Variable	Median	Mean	Std. Dev.	*p*-value	% of Positive effect
Exp I	Time	−36.00	−46.30	33.78	**7.1e-05**	95.00
	F-measure	11.50	8.92	20.37	**0.048**	65.00
	Recall	−7.95	−0.90	26.47	0.820	40.00
	Precision	18.57	13.10	32.17	**0.045**	65.00
Exp II	Time	−60.25	−49.00	−19.06	**0.001**	100.00
	F-measure	−5.09	0.53	14.26	0.484	33.33
	Recall	12.34	7.69	20.05	0.098	58.33
	Precision	−5.62	−5.31	19.50	0.867	41.67
All	Time	−44.00	−49.06	29.01	**6.1e-07**	96.88
	F-measure	5.69	5.74	18.54	0.064	53.13
	Recall	−3.30	2.32	24.29	0.430	46.88
	Precision	4.62	6.20	29.18	0.129	56.25

Exp I: statistical significant reduction of time and tracing errors (precision) made by subject

Exp II: only the reduction of time is statistically significant

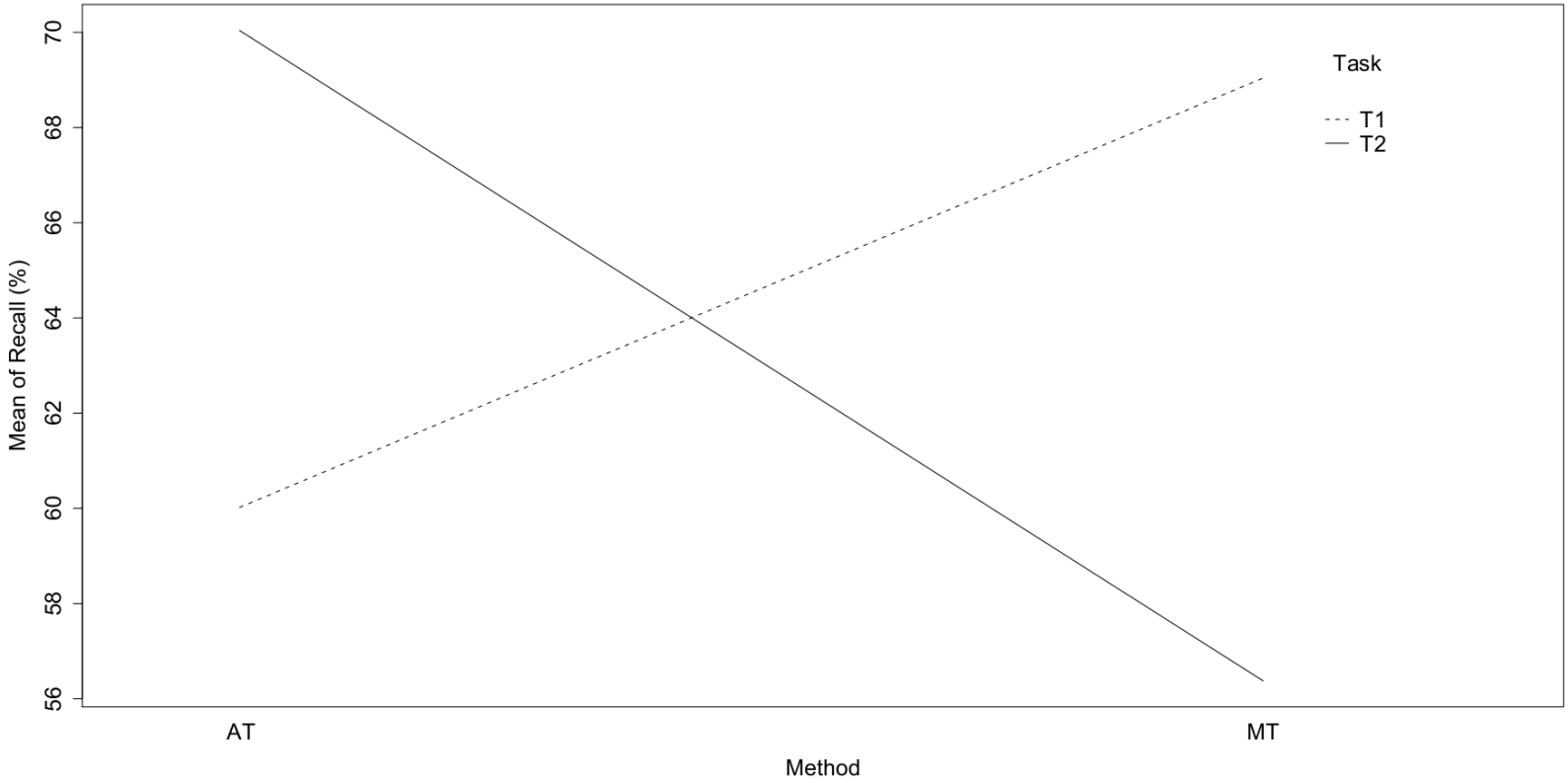
Effect of Ability



The tool helps to reduce the gap between high and low ability subjects

The tool (sometimes) represents a bias for the subject who in the doubt trace a link suggested by the tool

Effect of Task

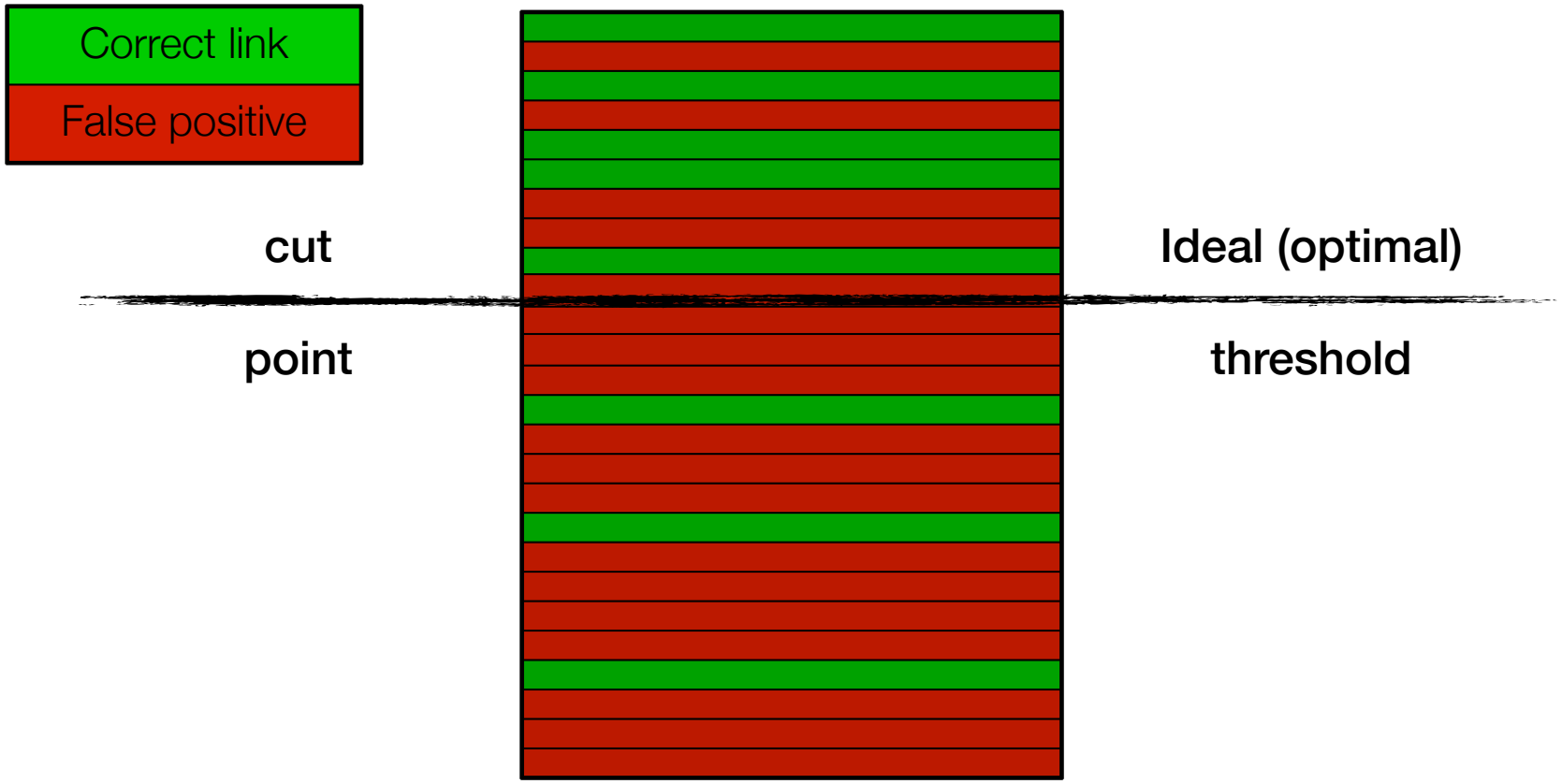


The retrieval accuracy of the IR engine is an influencing factor

The accuracy of the IR engine is better for task T2 (tracing interaction diagrams onto test cases) than for task T1 (tracing use cases onto code classes)

Such a situation is also reflected in the perceived usefulness of the tool

How to use IR-based Traceability Recovery Tools



Automatic tracing should be combined with manual tracing!

Incremental traceability recovery

Source_1	Target_2	95.4%	
Source_3	Target_4	92.1%	90%
Source_1	Target_1	85.6%	
Source_2	Target_2	83.2%	
Source_3	Target_3	81.2%	80%
Source_1	Target_3	79.0%	
Source_3	Target_2	77.5%	
Source_2	Target_4	72.3%	70%
Source_2	Target_3	53.2%	
Source_3	Target_1	43.9%	
Source_2	Target_1	38.7%	
Source_1	Target_4	23.6%	

Correct link

False positive

Link classification

Source_1	Target_3	79.0%
Source_3	Target_2	77.5%
Source_2	Target_4	72.3%

The software engineer decides to stop the process as the effort to discard false positives is becoming to high. Probably he does not retrieve all correct links!

Drawback of the Incremental Process

Threshold used to stop the process plays an important role

- The lower the threshold the higher the number of correct links retrieved

Better recall could be achieved by providing the software engineer with the full ranked list...

- But, the ranked list contains a high density of correct links in the upper part and a low density of such links in the lower part

Thus

- Full ranked list (“one-shot”) method might result in a better recall
- But in a worse precision (more tracing errors) with respect to the incremental process

Incremental vs “One shot” (1)

Evaluate the impact of the traceability recovery process adopted on the tracing performances of the software engineer (accuracy and time)

Context…

20 Master students of the University of Salerno

They had to perform two traceability recovery tasks on a repository of a completed software project

T1: recovering links between use cases and classes

T2: recovering links between UML diagrams and test cases

Tasks were performed in two separate laboratory sessions using the two different approaches (incremental and one-shot)

Tasks were performed using ADAMS Re-Trace

Andrea De Lucia, Rocco Oliveto, Genoveffa Tortora: IR-Based Traceability Recovery Processes: An Empirical Comparison of "One-Shot" and Incremental Processes. ASE 2008: 39-48

Incremental vs “One shot” (2)

	Med.	Mean	St. Dev.	p-value	Effect size	% of Pos. Effect
time “one-shot” vs inc.	1.00	0.65	12.63	0.589	-	45.00
recall “one-shot” vs inc.	5.47	6.95	17.63	**0.035**	**0.28**	75.00
precision inc. vs “one-shot”	11.65	9.81	20.38	**0.032**	**0.57**	70.00

Wilcoxon test p-values and descriptive statistics of differences by subject

Better recall using the one-shot approach

Better precision using the incremental approach

Incremental vs “One shot” (3)

Analising the effect size...

- on recall, only small effect (0.28)
- on precision, a medium effect (0.57)

Deeper analysis of the results

- Threshold used to stop the incremental traceability recovery
- On average, a low threshold, i.e., about 40%
- Two outliers stop: they stopped the process at 65% of similarity

New analysis of the results discarding the outliers

- No influence on recall (p-value = 0.064)
- Influence on precision (p-value = 0.023)

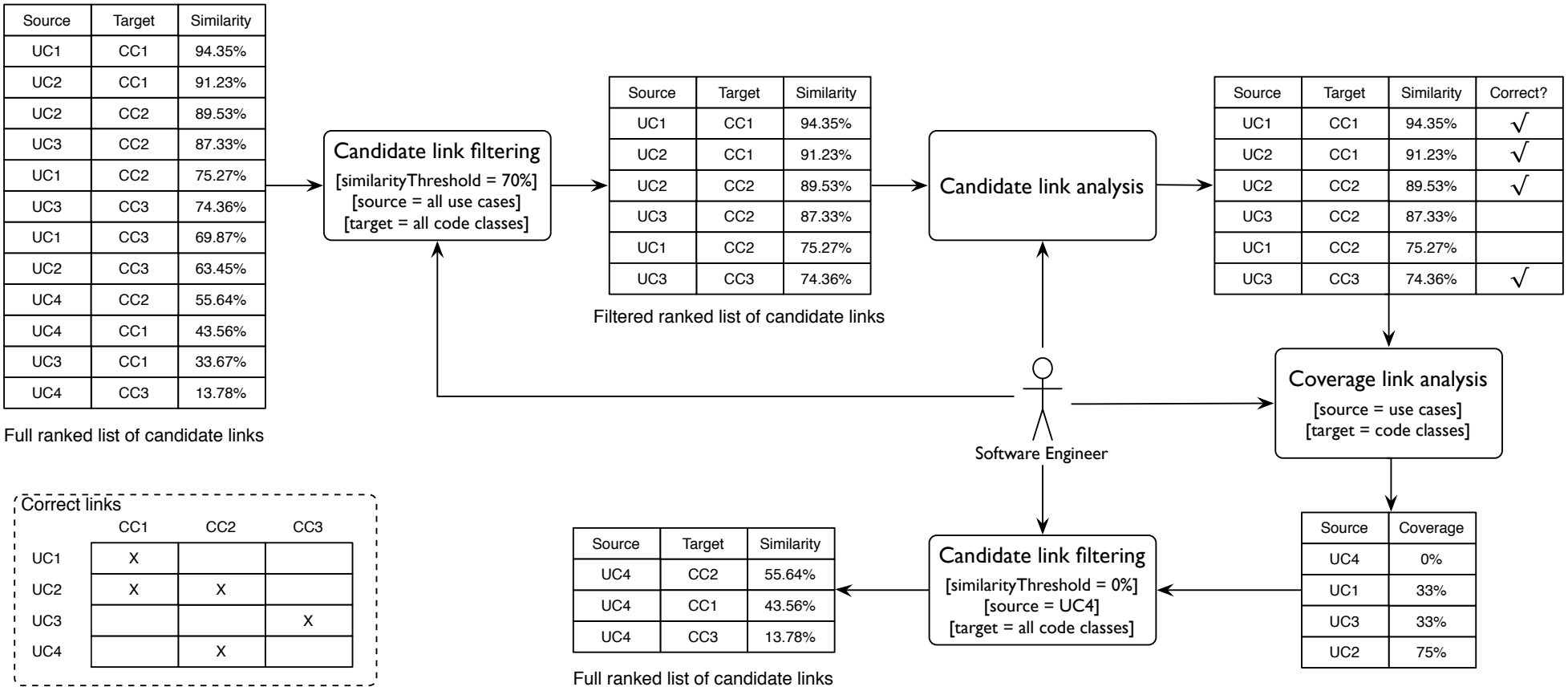
Coverage Link Analysis: a Two Steps Approach

First step: the software engineer performs an incremental massive traceability recovery between a set of source artefacts and a set of target artefacts

During this step she traces as many links as possible keeping low the effort to discard false positives

Second step: she uses a coverage link analysis aiming at identifying source artefacts poorly traced and guiding focused fine-grained traceability recovery sessions to recover links missed in the first step

Example



How to enrich the set of correct links with coverage analysis

Benefits of Coverage Link Analysis (1)

Evaluate the impact of coverage analysis on the tracing performances of the software engineer (recall and precision)

Context…

30 Master students of the University of Salerno

They had to perform two traceability recovery tasks on a repository of a completed software project

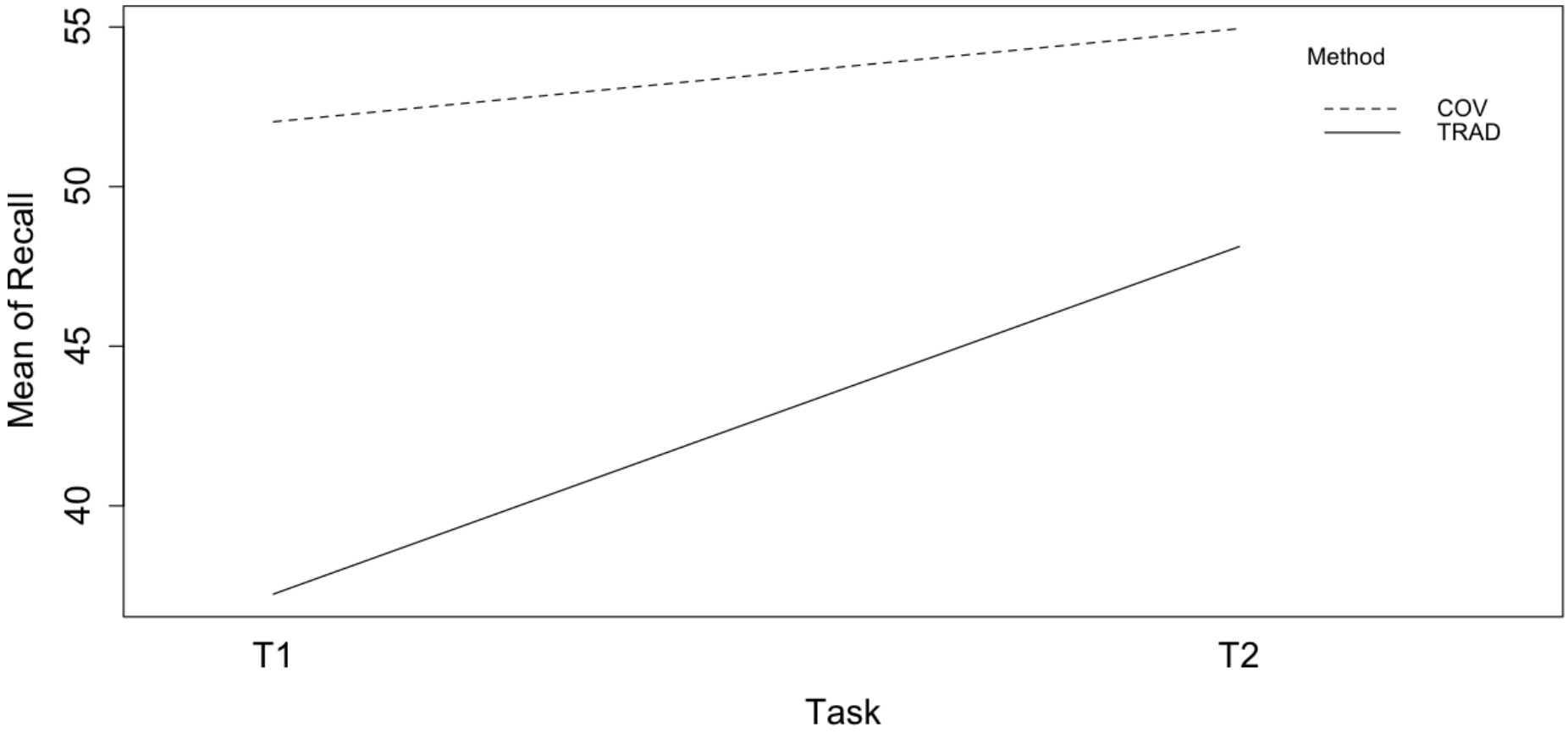
T1: recovering links between use cases and classes

T2: recovering links between UML diagrams and test cases

Tasks were performed in two separate laboratory sessions using ADAMS Re-Trace with and without the coverage analysis feature

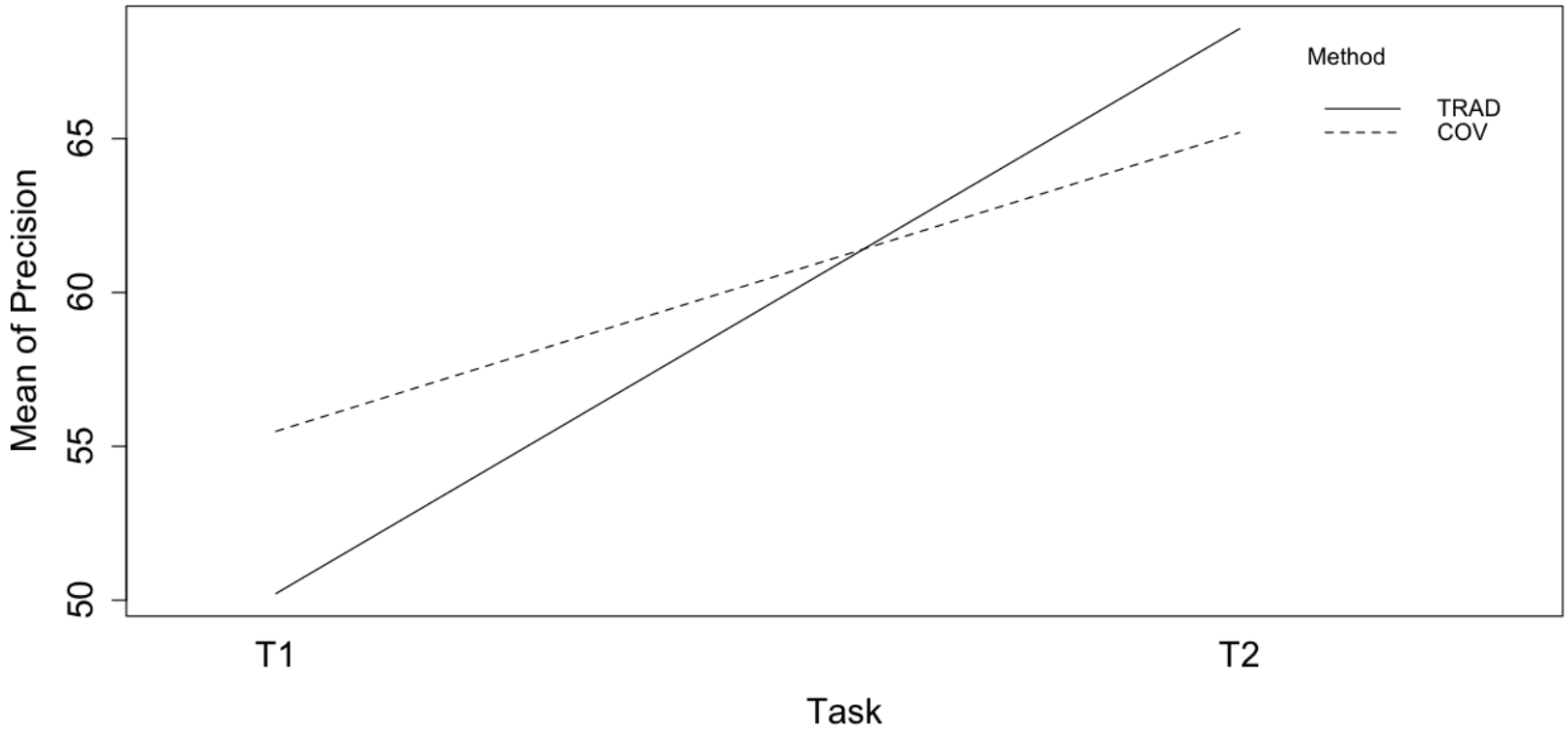
Andrea De Lucia, Rocco Oliveto, Genoveffa Tortora: The role of the coverage analysis during IR-based traceability recovery: A controlled experiment. ICSM 2009: 371-380
Gabriele Bavota, Andrea De Lucia, Rocco Oliveto, Genoveffa Tortora: Enhancing Software Artefact Traceability Recovery Processes with Coverage Analysis. Information and Software Technology.

Benefits of Coverage Link Analysis (2)



Better recall (always) when using coverage analysis!

Benefits of Coverage Link Analysis (3)



Better precision when tracing use cases onto code classes, but worst precision when tracing UML diagrams onto test cases

Some features were not implemented and thus not tested. This means that several UML diagrams have 0% as coverage (subjects were misguided)

Challenges in IR-based traceability recovery

There is still room for improving the accuracy of IR-based traceability? Is it possible to combine textual information with other information?

Is also possible to extract the semantics of a link? In other words, it is possible to label with words (or a sentence) a link in order to improve its comprehensibility?