# Comparing Similarity Learning with Taxonomies and One-Mode Projection in Context of the FEATURE-TAK Framework

Oliver Berg[3], Pascal Reuss[1,2](✉), Rotem Stram[1,3], and Klaus-Dieter Althoff[1,2]

[1] German Research Center for Artificial Intelligence, Kaiserslautern, Germany
reusspa@uni-hildesheim.de
[2] Institute of Computer Science, Intelligent Information Systems Lab, University of Hildesheim, Hildesheim, Germany
[3] Department of Computer Science, Technical University of Kaiserslautern, Kaiserslautern, Germany
http://www.dfki.de/
http://www.uni-hildesheim.de/

**Abstract.** This paper describes the learning of new similarity values for existing measures within the framework FEATURE-TAK. Maintenance of similarity measures is not easy, especially when having a semi-automated approach to relieve the knowledge engineer. Based on the extension of the vocabulary, the newly added values have to be integrated into the similarity measures with an initial similarity value to be useful. We describe the extension of the similarity measures with automated taxonomy extension and one-mode projections and present a comprehensive evaluation and comparison between the different approaches to highlight the advantages and short comings.

**Keywords:** Case-based reasoning · Similarity measures · Knowledge modeling · One-mode projection

## 1 Introduction

To solve occurring problems in areas like monitoring, maintenance and operation one requires knowledge of the situation and how to resolve the issue. Knowledge can be approximated through storing data of a past problem description together with context information and executed solutions, which forms a so-called case [1]. Case-based Reasoning (CBR) then tries to solve new problems through noticing similarities with previously solved problems and adapting their known solutions, as such modelling human reasoning [8]. This indicates that similarity of problem descriptions remains pivotal to finding adequate solutions. The quality of retrieved reference cases thus highly depends on the defined similarity measures. Exact similarity measures frequently rely on use case specific variables and expert understanding of how an application operates. This dependence on expert knowledge imposes restrictions to completeness and performance of any CBR implementation. To this end, the **F**ramework for **E**xtraction, **A**nalysis, and **T**ransformation of **U**nstructu**RE**d **T**extual **A**ircraft **K**nowledge (FEATURE-TAK) is being developed, which aims at automating knowledge acquisition, specifically in the aviation domain. From problem fault descriptions in free-text form as provided by aviation maintenance personnel, it retrieves keywords, phrases and synonyms/hypernyms for specific attributes to enrich the systems vocabulary and allow for a more refined context description. As of now, FEATURE-TAK employs keywords and synonym structure for local (attribute level) similarity approximation for individual attribute values, and sensitivity analysis for global (case-level) similarity approximation for similarity of complete case descriptions. Similarity of single attributes is aggregated and weighted to build up complete case similarities [9,10].

Local similarity operates on mostly non-numeric symbol attribute descriptions (phrases from text) as opposed to numeric values. This makes attribute similarity difficult to infer in a non-binary - not only attribute value equal to or unequal to value X - manner. The approach currently employs taxonomies, as such extracting dependent symbols to then infer level-based similarity properties. This assesses similarity only for related keywords, which greatly reduces the actual number of at-all-similar relations. In its current implementation it presents major implications regarding modelling assumptions and does not generally capture all attribute dependencies as it relies on synonyms and hypernyms. The compromise is intentional though, because synonym-hypernym-connections enable taxonomies in the first place and any taxonomy related implementation of similarity will likely continue to employ synonyms/hypernyms. Global similarity on the other hand weights single attributes through sensitivity analysis based on relevance for solving cases. This allows the system to build averages used to calculate the global similarity value for comparing different cases. As this presents a primarily novel technique, it relies on mostly project-specific configuration and uses fewer well-established procedures.

Overall case-similarity should take the total case description into account. These local measures are then being weighted and combined into a final global similarity, being in return utilized to rank the solutions and return the most likely solution back to the system user. Exact local measures, weighting and ranking implementations are an act of great balance, as it is both delicate and non-deterministic which dependencies are impacting problem transference and to what degree. Therefore, a projection-based similarity procedure [12,13] is being incorporated into the framework's local similarity assessment to add comparison between not directly related keywords. This Weighted One-Mode Projection (WOMP) has proven successful in prior experiments [12] but was not tested on real-world application data yet. This paper specifically displays the outcome of properly integrating it into FEATURE-TAK, evaluating performance inside the framework and act as a starting point to incorporate further concepts.

In Sect. 2 we describe briefly the framework FEATURE-TAK, the WOMP and its integration into the framework. Section 3 describes the evaluation setup and the results of the comparison between the taxonomy similarity learning and the one mode projection learning. Section 4 closes with a discussion and outlook.

## 2 Weighted One Mode Projection in FEATURE-TAK

This section gives a brief overview of the FEATURE-TAK framework and then describes the Weighted One Mode Projection and its integration in the framework. This transitions into Sect. 3 with the description of the evaluation setup, the results, and their interpretation.

### 2.1 FEATURE-TAK

Based on established procedures, namely the myCBR toolkit [4,6,11] and the agent-based SEASALT architecture [3], the FEATURE-TAK framework has been developed to support knowledge engineers in querying data that is organized in structured and unstructured format and with highly domain-specific information. Technical maintenance data is frequently available in attribute-value pairs, whereas logbook entries and feedback only exist in free-text format. Thus, a hybrid representation, combining attribute-value and textual representation, is implemented by accounting for attributes specified from meta-information surrounding a case description and incorporating information entities from text; further detail regarding input- and attribute data in [9]. FEATURE-TAK processes said free-text input, applying natural language processing (NLP) techniques alognside CBR to extract keywords, phrases and synonyms to comprise attribute values of a predefined case-structure. In addition to known CBR procedures, some novel automated knowledge transformation is added. The framework consists of five layers: Data-, Agent-, NLP-, CBR- and Interface Layer. Inside the agent layer, multiple agents provide functionality in form of designated tasks based on given input data and the required CBR data structure. The tasks are regrouped and separated into sub-steps. From a given free-text input file together with provided mapping-, abbreviation and white-/black-list files the data sets are transformed into the internal representation format. This process adapts dynamically to available input information. The mapping file is in the XML format and describes which information in the initial data are to be mapped to which attributes in the case structure. Exact transformation is based on the mapping information, which results in a case structure with precisely defined 71 - possibly sparse - attributes. [9] The agents in the framework are responsible for performing the tasks within FEATURE-TAK and are implemented as follows:

– The **Preprocessing Agent** (Task 0) is a prerequisite for the subsequent stages as it prepares the input data through part-of-speech (POS) tagging and abbreviation identification.

- The **Collocation Agent** (Task 1) extracts phrases as recurring word combinations based on standard English grammar and domain-specific patterns.
- The **Keyword Agent** (Task 2) extracts keywords via stop-word elimination, lemmatization and single word abbreviation replacement, returning the word's base form.
- **The Synonym Agent** (Task 3) identifies synonyms and hypernyms considering word context and -sense, thus utilizing the POS information and provided black- and whitelists.
- The **Vocabulary Manager** (Task 4) adds phrases, keywords and synonyms/hypernyms to the CBR system's vocabulary.
- The **Similarity Manager** (Task 5) sets similarity values for concepts, extending attribute similarity measures to compare overall cases with each other, and respectively utilizes taxonomies on top of generated phrases, keywords and synonyms/hypernyms to further infer attribute value proximity.
- The **ARM Agent** (Task 6) searches for association rules in word occurrences within and across data sets, where - depending on data set size and performance constraints - either the Apriori [2] or the FP-Growth [5] algorithm with a high confidence of 0.9 being used to only allow rules found to be true most of the time.
- The **Clustering Agent** (Task 7) generates a case from each corpus of input data with an associated cluster (based on aircraft type and component, where components are specified with a unique digit called Air Transport Association (ATA) chapter) to persist it into the case bases.
- The **Sensitivity Agent** (Task 8) finally generates global similarity measurement between complete cases by incrementally approximating weight vectors over the set of attributes for each case cluster, resulting in a global relevance weight matrix [14].

As such, starting from the sparse 71-attribute-representation with only directly extracted values set, the pre-processing agent is engaged and subsequently the agents are traversed as described above. Note that tasks 0–3 are executed in strict sequence on the previous task's respective output, and only then slight branching is being undergone. From there, with a concise vocabulary description, similarity between values of a given attribute is aggregated, whereas global case similarity through the sensitivity analysis operates on subsets of data in form of generated case clusters as well as internal attribute similarity approximation [9].

## 2.2 Integration of the Weighted One-Mode Projection

An alternative to manually described similarity matrices and also taxonomies is the more general usage of projections of bipartite graph structures. Complex network analysis is, generally speaking, an emerging field regarding modelling relationship. In this context, bipartite graphs present a particular type of graph that consists of two distinct populations of nodes between which, but not within which, nodes are connected. This type of graph can be nicely used to model real-world systems such as for example describe personal preference [16] or depict co-authorship in scientific publishing [7]. These scenarios all have two groups of nodes interacting with each other, following the example of scientific publishing, for example "authors" and "papers". To find relations between entities within one of the two populations, a simple One-Mode Projection (OMP) or if weighted a Weighted One-Mode Projection (WOMP) can be calculated. Following the co-authorship analogy above, this would find relations or "similarity" between authors based on their collaboration in scientific publications. Regarding how the projection is calculated, Fig. 1 provides a simple example: The idea is that for each element in one population $l_{\mathrm{a}} \in L$, one considers each other element $l_{\mathrm{b}} \in L$ that can be reached via one or more elements $r_{\mathrm{j}} \in R$ of the opposing population. Considering only reachability, one obtains a simple one-mode projection, while counting the number of common elements in R results in a weighted one-mode projection. We assume projections to be calculated on the set L for simplicity reasons, projecting R can be done by simply inverting symbols in the notation [15].

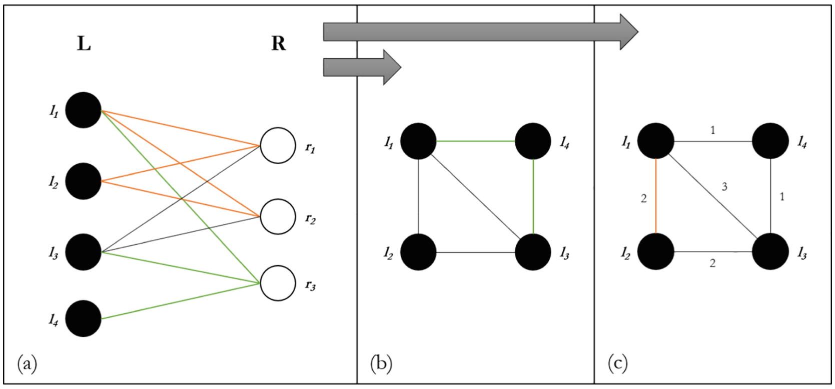


**Fig. 1.** Illustration of a bipartite graph and respective OMP/WOMP [12]

All of these approaches do however preserve the inherent projection problem of symmetry. Assuming symmetry in similarities cannot always be guaranteed to appy to real-world examples, as was stated in [12]. The so far discussed solutions all produce a single projected edge between any two entities $l_{\mathrm{a}}$, $l_{\mathrm{b}}$ in population L based on common properties through connections via a number of entities of the opposing population R. The resulting edge in the projection graph is not directed, $l_{\mathrm{a}}$ is as similar to $l_{\mathrm{b}}$ as is $l_{\mathrm{b}}$ to $l_{\mathrm{a}}$. Also, for any $r_{\mathrm{j}}$ that holds only a single connection to L, this connection cannot be properly represented in the projection graph's edge weights. Thus, the approach motivated by [12] is based on the idea introduced by [16] of nodes holding resources being distributed throughout the network. One assumes a bipartite graph with weighted edges, unweighted edges would correspond to all edges having a weight of 1.

The procedure then calculates resources to be distributed (visualized in Fig. 2). For each node $l_i \in L$ one aggregates all weights $w_{ij}$ of edges starting in $l_i$ and ending in all $r_j \in R$ to obtain the node's weight $W_i^L$ as its held aggregated resources (Fig. 2(a)). Similarly, $W_j^R$ is later calculated by aggregating all resource shares $w_{ij}^{L\rightarrow R}$ as they are propagated from $L \rightarrow R$ (Fig. 2(c)). Please note that notation is critical here: All $w_{ij}$, $w_{ij}^{L\rightarrow R}$ and $w_{ij}^{R\rightarrow L}$ refer to weights of the same edge connecting $l_i$ and $r_j$, with the former of the three considering the original weights as of the initial weighted bipartite graph, while the latter two are being calculated by the depicted procedure.

$$W_i^L = \sum_{j=1}^{|R|} w_{ij}, \quad w_{ij}^{L\rightarrow R} = \frac{w_{ij}}{W_i^L}, \quad W_j^R = \sum_{i=1}^{|L|} w_{ij}^{L\rightarrow R}, \quad w_{ij}^{R\rightarrow L} = \frac{w_{ij}^{L\rightarrow R}}{W_j^R} \tag{1}$$

To then obtain the final projection graph, one sums up all edges connecting any pair of nodes $(l_a, l_b)$ via nodes $r_j \in R$ of the opposing population. However, this shall result in a pair of directed weighted edges of weights $w_{ab}$ and $w_{ba}$. The difference in weights is achieved by not summing up all available edge weights between $l_a$ and $l_b$, but rather by following along the respective $w_{ab}^{L\rightarrow R}$ and $w_{ab}^{R\rightarrow L}$ according to the inherent direction ($a \Rightarrow b$ or $b \Rightarrow a$), in which $p_{ij} \in \{0, 1\}$ is used as indication of whether or not $l_i$ and $r_j$ share a connection. The resulting network is then the projection graph (not bipartite anymore) of the original (weighted) bipartite graph. This projection graph does not contain normalized edge weights. To obtain fixed weights in the interval $[0, 1]$, [12] proposes to utilize normalized weights $\hat{w}_{ab}^{L\rightarrow L}$ obtained by aggregating all incoming weights $w_{ab}^{L\rightarrow L}$ that lead into $l_b$ and normalize by dividing though the aggregate (called $w_{bb}^{L\rightarrow L}$).

$$w_{ab}^{L\rightarrow L} = \sum_{j=1}^{|R|} p_{aj} \cdot p_{bj} \cdot (w_{aj}^{L\rightarrow R} + w_{bj}^{R\rightarrow L}), \quad \hat{w}_{ab}^{L\rightarrow L} = \frac{w_{ab}^{L\rightarrow L}}{w_{bb}^{L\rightarrow L}} \tag{2}$$

With FEATURE-TAK operating on symbol attribute descriptions retrieved from free-text descriptions from maintenance operators, exact similarity rela-

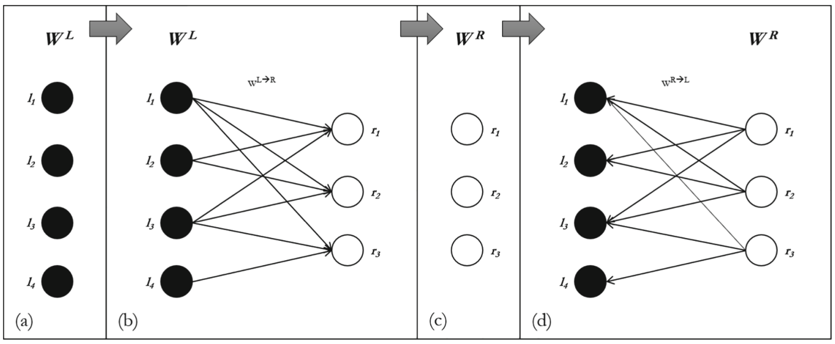


**Fig. 2.** Algorithmic workings of asym. WOMP with resource distribution [12]

tions between the retrieved keyword attribute values is non-trivial to infer. The goal of the current taxonomy-based similarity assessment algorithm is to position keywords alongside generated synonyms in a taxonomy structure that also builds on top of specific similarity values provided by domain experts and captures new concepts in this environment. The subsequently presented projection implementation has a different approach in that it specifically does not rely on manual similarity measures but rather derives relations between keywords by their co-occurrence in desired diagnosis results alone.

The projection operates on a bipartite graph of two populations:

- Population L, for which the one-mode projection graph is to be calculated, consists of keywords and synonyms being allowed values for a given symbol attribute description.
- Population R, which accounts for co-occurrence of to-be-projected attribute values, consists of all viable diagnoses (/ solutions) for the persisted case description.

A keyword/synonym $l_\mathrm{i}$ and diagnosis $r_\mathrm{j}$ are connected if there exists at least one case with diagnosis being $\mathrm{r}_j$ that also contains the keyword/synonym $l_\mathrm{i}$. The bipartite graph is weighted, such that the value of an edge represents the number of cases with the respective diagnosis in which the respective keyword also occurred. As such, the final projection graph strongly connects keywords together if and only if they co-occur in many cases with the same diagnoses. Keywords and synonyms may - and probably will - occur in multiple cases with possibly multiple different diagnoses and as such will be projected with the procedure of [12] according to how a keyword's "resources" are shared across multiple diagnoses. Similarly to taxonomies, one receives separate bipartite- and projection graphs for each of FEATURE-TAK's symbolic attributes.

Regarding nodes in population R, current data does not persist proper symbolic diagnosis classes. This is resolved by pattern-matching a case's *solution-recommendation* attribute, resulting in a class label out of the set of labels of "UNKNOWN", "FIX", "REPLACE", "RESET", "DISPATCH" and "MISC".

The implemented asymmetric weighted one-mode projection algorithm is being split up into two parts, approximately depicting Eqs. 1 and 2, respectively. Firstly, resources for L- and R-vertices as well as the edges are being composed, which is done in a bipartite graph object. Inside the bipartite graph object, a composeResources() function is being called with vertex- and edge-array-lists being completed. It then assigns resources to each vertex $l_i \in L$, edge $w_{ij}^{L \to R}$, vertex $r_j \in R$ and edge $w_{ij}^{R \to L}$ in precisely this order. Note that the depicted procedure resembles the algorithm as mathematically proposed rather closely. Secondly, the actual traversal of the bipartite structure according to Eq. 2 is done to obtain the projected resource distribution relations.

Inside FEATURE-TAK's architecture, local similarity assessment is done in the *SimilarityManager* (task 5) and as such can expect to utilize all transformed results from tasks 0 to 3 and an initially set (synonym) attribute and similarity value from task 4. The current taxonomy-based similarity procedure is implemented in a single function addSynonymSimilarity inside the task directory and engaged by the respective task (5) agent object in the framework. It ultimately manipulates the taxonomy structure for the newly added synonyms (where applicable) and outputs an integer count of how many synonyms got connected. Similarly, projections would result in a secondary addProjectionSimilarity function called by the task agent.

Concerning general implementation architecture, the projection objects and graph structures are as of now disconnected from the framework's overall structure as best they can. With one of the intended benefits of a projection-based similarity computation being that of a more holistic, loosely coupled execution of similarity computation, this becomes easier to achieve with a separate implementation. This does, however, not necessarily go as well regarding integration in the object- and file formats defined in the underlying myCBR-framework. The framework is already used for internal case representations and similarity function specifications as well as persisted attributes and project-file formats. As is, the projection part of the framework does not implement interfaces of myCBR.

## 3 Evaluation

### 3.1 Similarity Matrix Computation and Modelling Assumptions

The presented projection approach is tested by loading project file information from project-files into memory. Then for each (symbolic) problem description attribute (*system, status, function, location*) taxonomy similarity values are calculated for each attribute value pair and the bipartite graph is composed based on vertices created from symbolic attribute values and calculates similarity through projection methods to compute the similarity matrix across all allowed symbolic values for each similarity function (taxonomy, simple OMP, normalized asymmetric WOMP), finally persisting similarity matrices to CSV files. This reuses components of the myCBR framework as they had been used in the framework FEATURE-TAK as well, though it omits actually executing FEATURE-TAK itself. It rather operates in an "offline" mode, not relying on having to execute all agents and waiting for feedback which itself does not provide results to similarity measures. The framework FEATURE-TAK is meant to analyze maintenance data and enrich the underlying CBR system; inspecting similarity computation assumes the CBR system to be up-to-date to allow disregarding the multi-agent system toolchain and operate on the CBR data directly. This approach is valid as long as no intermediate FEATURE-TAK-specific information and data structures are required. Specifically, synonym information would be easily extractable from these data structures.

Regarding modelling diagnosis classes, the class labels are constructed by keyword-matching according to the depicted keywords in Table 1.

The attribute "sol_recommendation" is being checked for an exact match of a part of the string-value and if the word(s) occur within the string the corresponding class label is assigned. Slight exceptions to this procedure are *UNKNOWN* (which requires not part of the value but the complete value to fit the word) and *MISC* (which is assigned if none of the other class labels can be

**Table 1.** Diagnosis class labels as manually assigned to case instances

| Solution class label | String words |
|---|---|
| UNKNOWN | (*precisely*) "_unknown_", "none" |
| FIX | "fix", "de-ice" |
| REPLACE | "replace", "interchange", "swap", "change" |
| RESET | "reset", "re-power", "install new" |
| DISPATCH | "dispatch", "defer" |
| MISC | *(none of the above)* |

assigned, thus acting as a default label for attribute values not captured with the generated rules).

The generated output as of the implementation of similarity computation based on respective in- and outputs consists of a total of 20 CSV files. For each of the four problem fault description attributes ("function", "location", "status", "system") as well as for each similarity computation method (taxonomy, simple- and weighted OMP, stock- and normalized asymmetric WOMP) a separate similarity matrix was generated. Note that the only formal similarity matrices are the ones of taxonomy and normalized asymmetric WOMP similarity, as no non-normalized matrix representation allows for entries in the diagonal to have similarities of 1 as they do not contain normalized values. As projections by themselves do not capture self-similarities of attributes, these needed to be added after computations have already been executed. All non-normalized quasi-similarity matrices do, however, show the intermediate relations depicted throughout the projection process, which helps quantify the projection performance on the given case instances. The taxonomy similarity matrices shall provide a baseline against which to compare projection similarity measures.

## 3.2 Evaluation Results

As can be seen in Table 2, which shows excerpts of the "location" attribute taxonomy similarity values, taxonomies in FEATURE-TAK produce similarity matrices which hold similar non-zero entries across rows and columns.

The location attribute was chosen as an example due to the feasible amount of 75 attributes as well as it being well populated with taxonomy similarity measures other than 1 and 0.8 (synonym relations). For distributions of (non-)zero entries and other characteristics across similarity matrices, see Table 4.

Considering the same attribute matrix but with entries calculated via projections the "location" attribute produces only two similarity values, thus Table 3 shows another excerpt of the projection similarity matrix that contains more non-zero entries. Note that the two attributes "side" and "forward": Both are approximately - but not exactly - equally similar to each other ($\text{sim}_{side \rightarrow forward} = 0.015$, $\text{sim}_{forward \rightarrow side} = 0.017$). Compare this however to the attributes "aft" and "forward": With $\text{sim}_{forward \rightarrow aft} = 0.004$, forward is much less similar to

**Table 2.** Excerpt of prob_fault_description_location_taxonomy.csv

| Attribute value | d | d | w | D | f | w | u | u | u | u | c | i | c | b | l | l | m | c | b | m |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| door | 1 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.8 | 0.8 | 0.8 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.5 | 0 | 0.8 |
| deck | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| washstand | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Door Right | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| fore | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| washbowl | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| upper [...] door #1 | 0.8 | 0 | 0 | 0 | 0 | 0 | 1 | 0.8 | 0.8 | 0.8 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.5 | 0 | 0.8 |
| upper [...] door #2 | 0.8 | 0 | 0 | 0 | 0 | 0 | 0.8 | 1 | 0.8 | 0.8 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.5 | 0 | 0.8 |
| upper [...] door #3 | 0.8 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.8 | 1 | 0.8 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.5 | 0 | 0.8 |
| upper [...] door #4 | 0.8 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.8 | 0.8 | 1 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.5 | 0 | 0.8 |
| cabin work station | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.5 | 0.5 | 0.5 | 0.5 | 1 | 0 | 0 | 0 | 0 | 0 | 0.5 | 0.5 | 0 | 0.5 |
| incline | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| cargo | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| bathroom | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| lav | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| level | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| main [...] door #1 | 0.8 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.8 | 0.8 | 0.8 | 0.5 | 0 | 0 | 0 | 0 | 0 | 1 | 0.5 | 0 | 0.8 |
| cookhouse | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.5 | 1 | 0 | 0.5 |
| basin | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| main [...] door #4 | 0.8 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.8 | 0.8 | 0.8 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0.8 | 0.5 | 0 | 1 |

aft than the other way around ($\text{sim}_{aft \rightarrow forward} = 0.014$). This can be explained with forward being more closely related to other more impactful attributes (like "side"and "right"), while aft distributes its similarity over mostly shared "main deck left hand door #[1,3,4]" attributes.

Considering all the different matrix similarity representation formats, the different produced matrices show how the similarity computation proceeds across multiple stages. For illustration purposes, Fig. 3 shows original taxonomy similarity matrix, WOMP and both stock- and normalized asymmetric WOMP similarity matrices, with coloring indicating strength of similarity values. The matrices are excerpts of the problem fault description status attribute example, which is more densely populated with also different WOMP values (see Table 4 for more detail). All matrices apply on the same attribute pairs.

Figure 3 illustrates that distinct different relations exist between taxonomy- and projection measures. While the taxonomy relies on nodes modeled by domain expert and taxonomy similarities, the projection builds up over multiple stages from the symmetric WOMP of the bipartite graph, which is then traversed asymmetrically and normalized. Note that the asymmetric (non-normalized) WOMP is not computed from the simpler WOMP directly, but the properties as of the symmetric WOMP carry over into the asymmetric representation.

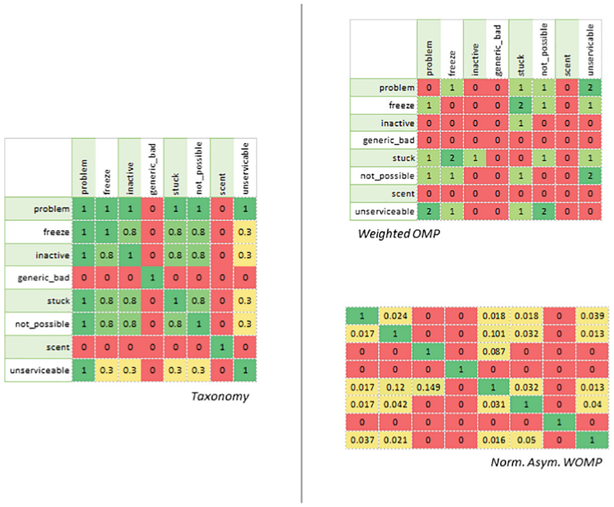
**Table 3.** Projected normalized asymmetric similarity values

| Attribute value | m | m | m | s | m | m | s | w | u | u | f | u | c | r | s | s | a |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| main [...] #1 | 1 | 0 | 0.015 | 0 | 0 | 0.014 | 0 | 0 | 0 | 0 | 0.009 | 0 | 0 | 0 | 0 | 0 | 0.012 |
| main [...] #2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| main [...] #3 | 0.020 | 0 | 1 | 0 | 0 | 0.020 | 0 | 0 | 0 | 0 | 0.014 | 0 | 0 | 0 | 0 | 0 | 0.017 |
| side | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0.015 | 0 | 0 | 0.016 | 0 | 0 | 0 |
| middle | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| main [...] #4 | 0.020 | 0 | 0.020 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0.014 | 0 | 0 | 0 | 0 | 0 | 0.017 |
| side of meat | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| washbasin | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| upper [...] #2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| upper [...] #3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| forward | 0.004 | 0 | 0.005 | 0.017 | 0 | 0.005 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0.015 | 0 | 0 | 0.004 |
| upper [...] #4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| crew rest [...] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| right | 0 | 0 | 0 | 0.019 | 0 | 0 | 0 | 0 | 0 | 0 | 0.015 | 0 | 0 | 1 | 0 | 0 | 0 |
| sanitary | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| storey | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| aft | 0.020 | 0 | 0.020 | 0 | 0 | 0.020 | 0 | 0 | 0 | 0 | 0.014 | 0 | 0 | 0 | 0 | 0 | 1 |

The displayed taxonomy excerpt contains 1.0 entries on the diagonal indicating self-similarity, but also in other non-diagonal entries where attributes have been connected via parent nodes of value 1.0 (e.g. "problem" and "inactive"). The 0.8 values indicate similarity between synonyms (e.g. "inactive" and "stuck"). The entries of value 0.3 are connected via weaker parent nodes, resulting in smaller similarity (e.g. "inactive" and "unserviceable").

On the projection side of things, one can observe that more entries are set to 0 (attributes not similar). Few entries have values (number of common neighbors in R-population) greater than 1. The continued asymmetric projection still captures higher values for entries with values greater than 1 in the previous weighted projection, where the asymmetry and weighting across complete attribute similarity values introduce further deviation. The final normalized asymmetric WOMP then contains entries generally being very small in similarity value. Relations from the previous non-normalized asymmetric WOMP largely persist, but are heavily scaled down, which gets further amplified for larger similarity matrices. Manually inserting self-similarity properties through 1.0 entries in the diagonal appear disproportional to other similarity values. As Fig. 3 shows, though formats of both taxonomy- and projection similarity build on fundamentally similar ideas, they yield different values through different ways of reasoning. This leaves open the question of how well these different measures are comparable not from a conceptual standpoint but from an application and data-oriented one.

To additionally compare not only representations across the same attribute, but also overall shape and how many actual similarity relations are contained



| | problem | freeze | inactive | generic_bad | stuck | not_possible | scent | unservicable |
|---|---|---|---|---|---|---|---|---|
| problem | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| freeze | 1 | 1 | 0.8 | 0 | 0.8 | 0.8 | 0 | 0.3 |
| inactive | 1 | 0.8 | 1 | 0 | 0.8 | 0.8 | 0 | 0.3 |
| generic_bad | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| stuck | 1 | 0.8 | 0.8 | 0 | 1 | 0.8 | 0 | 0.3 |
| not_possible | 1 | 0.8 | 0.8 | 0 | 0.8 | 1 | 0 | 0.3 |
| scent | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| unserviceable | 1 | 0.3 | 0.3 | 0 | 0.3 | 0.3 | 0 | 1 |

*Taxonomy*

| | problem | freeze | inactive | generic_bad | stuck | not_possible | scent | unservicable |
|---|---|---|---|---|---|---|---|---|
| problem | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 2 |
| freeze | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 1 |
| inactive | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| generic_bad | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| stuck | 1 | 2 | 1 | 0 | 0 | 1 | 0 | 1 |
| not_possible | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 2 |
| scent | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| unserviceable | 2 | 1 | 0 | 0 | 1 | 2 | 0 | 0 |

*Weighted OMP*

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0.22 | 0 | 0 | 0.22 | 0.22 | 0 | 1.06 |
| 0.36 | 0 | 0 | 0 | 1.25 | 0.39 | 0 | 0.36 |
| 0 | 0 | 0 | 0 | 1.09 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.36 | 1.12 | 0.6 | 0 | 0 | 0.39 | 0 | 0.36 |
| 0.36 | 0.39 | 0 | 0 | 0.39 | 0 | 0 | 1.08 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.8 | 0.2 | 0 | 0 | 0.2 | 0.61 | 0 | 0 |

*Asym. WOMP*

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0.024 | 0 | 0 | 0.018 | 0.018 | 0 | 0.039 |
| 0.017 | 1 | 0 | 0 | 0.101 | 0.032 | 0 | 0.013 |
| 0 | 0 | 1 | 0 | 0.087 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0.017 | 0.12 | 0.149 | 0 | 1 | 0.032 | 0 | 0.013 |
| 0.017 | 0.042 | 0 | 0 | 0.031 | 1 | 0 | 0.04 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0.037 | 0.021 | 0 | 0 | 0.016 | 0.05 | 0 | 1 |

*Norm. Asym. WOMP*

**Fig. 3.** Comparing similarity matrix representations of the "status" attribute

in the presented similarity matrices, Table 4 shows an overview of how many entries of a particular type of similarity value are in different similarity matrices across multiple attributes. For all four problem fault description attributes, the taxonomy and asymmetric projection matrices are covered as well as the intermediate symmetric weighted projection. The numbers for the asymmetric projection holds for both the normalized and not normalized matrices, thus it is only listed once. The symmetric projection is included to give an intuitive understanding of how many occurrences of attributes overlap in the resulted projection, to which the simple non-weighted projection does not provide additional insight, thus it is excluded from the overview.

Inspected matrix properties are the number of total entries and non-zero (as similarity is always positive this depicts all entries of $\mathrm{sim}_{a \to b} > 0$) as well as non self-similar values (thus excluding entries d on the diagonal). For taxonomies it is also interesting how many non-intrinsically related similarity values - in the sense of values not specifically assigned by domain experts directly or the framework FEATURE-TAK implicitly through synonym structures - exist, thus only entries of similarity 0.8 are excluded further from the measure to exclude synonym relations. This does also neglect valid taxonomies just happening to return a similarity of 0.8 without the prior use of synonym constructs, but after manually inspecting the taxonomy similarity table and operation of FEATURE-TAK, this possibility was estimated to be unlikely. Note that domain experts specify intermediate nodes, which as such not necessarily depict attribute values expected to occur in real cases, and as such they needed not be inspected separately from formally calculated similarity values. For the intermediate symmetric weighted projection, an important measure is that of similarity relations depicted through values greater than 1, because these correspond to multiple

**Table 4.** Similarity matrices statistics across attributes

| Attribute | Taxonomy | N. Asym. WOMP | Symmetric WOMP |
|---|---|---|---|
| *Function* (65 × 65, 4225 entries) | >0: 577<br>>0, ¬d: 512<br>>0, ¬d, ¬0.8: 0 | >0: 133<br>>0, ¬d: 68 | >0: 68<br>>0, >1: 6 |
| *Location* (75 × 75, 5625 entries) | >0: 967<br>>0, ¬d: 892<br>>0, ¬d, ¬0.8: 498 | >0: 177<br>>0, ¬d: 102 | >0: 102<br>>0, >1: 0 |
| *Status* (137 × 137, 18769 entries) | >0: 4151<br>>0, ¬d: 4014<br>>0, ¬d, ¬0.8: 3082 | >0: 669<br>>0, ¬d: 532 | >0: 532<br>>0, >1: 66 |
| *System* (979 × 979, 635209 entries) | >0: 22827<br>>0, ¬d: 22030<br>>0, ¬d, ¬0.8: 21386 | >0: 22373<br>>0, ¬d: 21576 | >0: 21576<br>>0, >1: 62 |

common neighbors across attributes pairs, which in return allows to draw conclusions towards respective influences on the distribution of similarity in the normalized asymmetric projection values.

As seen in the left-most column of Table 4, the different attributes are very different in size with problem-fault-description-system being by far the largest of the four and the others being approximately of the same size with just - status being noticeably larger. As can be seen in the second and third column for taxonomy- and (normalized) asymmetric WOMP respectively, the attributes are more or less sparse depending on the type of values considered. Taxonomy values are largely zero with function, location, status and system having 14%, 17%, 22% and 4% non-zero entries, respectively. Compare this to 3%, 3%, 4% and 4% non-zero values for the asymmetric projection, which is considerably more sparse (except for the largest "system"-matrix). Though "status" is denser than "location" still being denser then "function" with taxonomies, this gets reduced to an even 3–4% across all attributes for projections. This drop is even more severe considering not only non-zero values but also values not equal to 0.8. Here, taxonomies maintain 0%, 9%, 16% and 3% similarity values with "location" keeping many-, while "status" and "system" keep most values. Problem fault description "function" only contains values of self- or synonym-similarity.

Considering weighted projections, both "function" and "status" hold a fair share of values larger than one, which allows for a more varying degree of similarity in the asymmetric projections, as opposed to larger matrices such as with "system" with fewer overlapping neighbors. In order to compare similarity measures of projections to their established taxonomy counterparts, a Mean Squared Error (MSE) estimation summing up the squared difference in values for all entries in both matrices was intended to be done, comparing the overall deviation in values between the taxonomy similarity matrix and the normalized asymmetric WOMP similarity matrix of each attribute separately. This however resulted in incredibly large deviations, because those similarity matrices do not overlap in most - if not all - non-zero values. Upon closer inspection as to why the values mostly do not overlap, it was found that the synonym structure as well as keywords related over said synonyms frequently happen to be conceptually related, without being used interchangeably in cases with similar solutions. What this means is that the taxonomies capture (pre-)defined classes of concepts, while projections rather capture concepts of different keywords used in similar situations in a scenario more closely related to the solution procedure of the case (as they get connected over solution classes).

## 4 Discussion and Outlook

The evaluation addresses inadequate symmetric self-similarity and small similarity values for sparse data as major issues. Regarding symmetric self-similarity, projections do not yield similarities of 1 for attributes to themselves. After having normalized all other non-self similarities, the similarities of attributes to themselves can be set to 1 manually, but they are not a result of the projection procedure itself, because the projection accounts only for co-occurrences of attribute values. The procedure of traversing the right-population has as consequence that any traversal is related to the resource being distributed to different solution classes, thus even adjusting for traversing all edges of the bipartite graph from a left-node to itself results in related twisted aggregated values due to $W^{\mathrm{R}}$ being typically non-zero, which is correlated to $w^{R \to L}$. While taxonomy similarity has attribute self similarity as inherent property, projections do not. This technically violates the fundamental notion of similarity, and requires manual adaptation if used on its own. Furthermore, as could be seen in Fig. 3, the different intermediate similarity matrices of symmetric-, asymmetric non-normalized- and asymmetric normalized WOMP have very different scaling across similarity values. When compared with similarity values in the taxonomy similarity matrix being nicely distributed over the interval [0, 1], the values of the final normalized asymmetric WOMP are incredibly small and - not counting self-similarity - rarely exceed a similarity of 0.1. This trait, however, is not shared across the intermediate representations, where the non-normalized matrix contains what seems like more adequate values, which can be larger than 1. Note that the basic normalization idea of [12] still applies, but it seems to not scale well in application, especially with a small number of connections between the populations of the bipartite graph as a result of a low number of cases.

A solution to this could be a logarithmic function, which maps normalized projection-based similarity values to a range incorporating also similarities closer to 1. Compared to a baseline of no mapping, this logarithmic mapping would ensure that for small projection similarities their output similarity value in the similarity matrix would be larger - e.g. attain a value of 1 for a value of 1 in the normalized projection similarity and setting larger values to exactly 1. The steepness of the initial logarithmic incline for very small values and how quickly the maximum similarity should be attained depends on the number of input cases and distribution of attributes and attribute co-occurrences within the attribute descriptions. A further study inspecting different parameters and their influence would be worthwhile.

With projections being more of an extension to - rather than a substitution of - taxonomy relations, under the given analysis and computations it seems that a combined application of projections as well as taxonomies is more likely to succeed as compared to utilizing projections on their own. How exactly both procedures get interconnected in the application architecture of FEATURE-TAK and how nicely the different components extend already implemented functionality was sketched but can at this point only be estimated. The idea of combining projections with taxonomies works intuitively from an ideological point of view - with both incorporating expert knowledge in a network structure, while resulting in different sets of similarity relations - where the combination procedure requires further planning.

Possible future work includes incorporating projections more closely into FEATURE-TAK's workflow and implementing projection similarity measures in myCBR as the underlying CBR framework. The maintenance procedure for continuous operation of FEATURE-TAK allowing scaling across longer framework runs is considerably easier for taxonomies than for projections, but trade-offs in complexity can be expected by batch-updating the bipartite graph.

## References

1. Aamodt, A., Plaza, E.: Case-based reasoning: foundational issues, methodological variations and system approaches. AI Commun. **7**(1), 39–59 (1994). http://www.idi.ntnu.no/emner/it3794/lectures/papers/Aamodt_1994_Case.pdf
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pp. 487–499 (1994)
3. Bach, K.: Knowledge acquisition for case-based reasoning systems. Ph.D. thesis, University of Hildesheim (2013). Dr. Hut Verlag München
4. Bach, K., Althoff, K.-D.: Developing case-based reasoning applications using myCBR 3. In: Agudo, B.D., Watson, I. (eds.) ICCBR 2012. LNCS (LNAI), vol. 7466, pp. 17–31. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32986-9_4
5. Borgelt, C.: An implementation of the FP-growth algorithm. In: Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations, pp. 1–5. ACM (2005)
6. Hundt, A., Reuss, P., Sauer, C.S., Roth-Berhofer, T.: Knowledge modeling and maintenance in myCBR3. In: Proceedings of the FGWM Workshop at Learning, Knowledge, Adaptation Conference 2014. Springer, Heidelberg (2014)
7. Lambiotte, R., Ausloos, M.: N-body decomposition of bipartite author networks. Phys. Rev. E **72**, 066117 (2005)
8. de Mantaras, R.L.: Case-based reasoning. In: Advanced Course on Artificial Intelligence, pp. 127–145. Springer, Heidelberg (1999)

9. Reuss, P., Stram, R., Althoff, K.-D., Henkel, W., Henning, F.: Knowledge engineering for decision support on diagnosis and maintenance in the aircraft domain. In: Nalepa, G.J., Baumeister, J. (eds.) Synergies Between Knowledge Engineering and Software Engineering. AISC, vol. 626, pp. 173–196. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-64161-4_9
10. Reuss, P., et al.: FEATURE-TAK - framework for extraction, analysis, and transformation of unstructured textual aircraft knowledge. In: Goel, A., Díaz-Agudo, M.B., Roth-Berghofer, T. (eds.) ICCBR 2016. LNCS (LNAI), vol. 9969, pp. 327–341. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47096-2_22
11. Stahl, A., Roth-Berghofer, T.R.: Rapid prototyping of CBR applications with the open source tool myCBR. In: Althoff, K.-D., Bergmann, R., Minor, M., Hanft, A. (eds.) ECCBR 2008. LNCS (LNAI), vol. 5239, pp. 615–629. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85502-6_42
12. Stram, R., Reuss, P., Althoff, K.-D.: Weighted one mode projection of a bipartite graph as a local similarity measure. In: Aha, D.W., Lieber, J. (eds.) ICCBR 2017. LNCS (LNAI), vol. 10339, pp. 375–389. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61030-6_26
13. Stram, R., Reuss, P., Althoff, K.-D.: Dynamic case bases and the asymmetrical weighted one-mode projection. In: Cox, M.T., Funk, P., Begum, S. (eds.) ICCBR 2018. LNCS (LNAI), vol. 11156, pp. 385–398. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01081-2_26
14. Stram, R., Reuss, P., Althoff, K.-D., Henkel, W., Fischer, D.: Relevance matrix generation using sensitivity analysis in a case-based reasoning environment. In: Goel, A., Díaz-Agudo, M.B., Roth-Berghofer, T. (eds.) ICCBR 2016. LNCS (LNAI), vol. 9969, pp. 402–412. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47096-2_27
15. Xu, K., Wang, F., Gu, L.: Behaviour analysis of internet traffic via bipartite graphs and one-mode projecions. IEEE/ACM Trans. Netw. (TON) **22**, 931–942 (2014)
16. Zhou, T., Ren, J., Medo, M., Zhang, Y.C.: Bipartite network projection and personal recommendation. Phys. Rev. E **76**, 046115 (2007)