UNIV. OF MILANO
COMPUTER SCIENCE DEPT.

NETT
EUROPEAN PROJECT

Toward a human-centric intelligent didactical recommender system

Nett Project

Authors:
Bruno APOLLONI
Nett Project TEAM

Affiliation:
Univ. MILANO
Univ. MILANO

January 23, 2014

Contents

1 Preliminaries, notation, preprocessing, and dataset

Exploiting the nowadays wide availability of learning resources repositories on the internet, the aim of the proposed algorithm is to support teachers to retrieve and select effectively and efficiently learning materials appropriate for their educational purposes. Such repositories contain information on existing courses in terms of both metadata describing their main features, and suitable judgments made by experts/learners evaluating their efficacy and organization.

In our acceptation, let us denote with c_i a generic course, for $i = 1, \ldots, n_c$, where $c_i = (\ell_{i1}, \ldots, \ell_{in_{\ell_i}})$ is composed of a set of n_{ℓ_i} modules (possibly depending on i). You can think to a module as a well-defined part of a course, like a set of lessons where a given topic is introduced, explained in details, possibly with the help of examples and exercises. In turn, each ℓ_{ij} is composed of a set of didactic resources $\{r_{ij1}, \ldots, r_{ijn_{r_{ij}}}\}$, the latter representing lectures, videos, exercises, and

symbol	description
c_i	i-th course
n_c	number of courses in the repository
ℓ_{ij}	j-th module of the i-th course
n_{ℓ_i}	number of modules in the i-th course
r_{ijk}	k-th resource of the j-th module in the i-th course
$n_{r_{ij}}$	number of resources in the j-th module of the i-th course
m_{ijkl}	l-th metadata associated to resource k of module j of course i
n_m	constant number of metadata

Table 1: Notation used throughout this report.

all available material related to the topics covered by the module ℓ_j. In order to abstract courses, modules, and resources, we associate to the latter a set of metadata[1] $(m_{ijk1}, \ldots, m_{ijkn_m})$ providing a set of n_m attributes. Table 1 summarizes the notation used throughout this report.

Each metadata is composed of a pair $\langle key, value \rangle$ where key is an abbreviation of the metadata, and $value$ an item chosen in a suitable domain. The complete list of metadata describing learning objects together with the values each key may assume is reported in Table 2. Note that, while Language, Resourcetype, Category, Difficulty, Format, and Time metadata are nominal variables, i.e. their domain is an enumeration of linguistic terms, Age identifies in a interval within the range $[6, 36]$ years, and the four Skills (s_req_skill, d_req_skill, s_acq_skill, d_acq_skill) share the same linguistic terms which are to be interpreted as multi-value attributes, in the sense that each resource may be tagged with more skill values. Moreover, the set of these linguistic terms depends on the course Category, which is reported in italics on the top of the related terms. Finally, Keywords will contain a set of English terms chosen by the user to provide an in-depth description of the resource.

[1] Such metadata are inherited by the respective module and course. Moreover, in the case some metadata is attached directly to either a module or a course, all the resources belonging to the same module and modules constituting the same course will share the same metadata.

key	*values*
Language	English, Italian, Bulgarian, Turkish
Resourcetype	Exercise, Simulation, Questionnaire, Diagram, Figure, Graph, Index, Slide, Table, Narrativetext, Exam, Experiment, Problemstatement, Self-assessment, Lecture
Category	EntrepreneurialVision, PersonalDevelopment, CommunicationSkills, EconomicSkills, TechnicalSkills
Difficulty	Veryeasy, Easy, Medium, Difficult, Verydifficult
Age	$[6, 36]$
Format	Video, Slide, Text, Audio
Time	30minutes, 60minutes, 90minutes, +120minutes
	EntrepreneurialVision
s_req_skill d_req_skill s_acq_skill d_acq_skill	Proactivity, Entrepreneurialbehaviorsandattitudes, Leadership, Self-evaluation, Self-organization, Innovativethinking, Creativethinking, OpportunitiesManagement, Abilitytopromoteinitiatives, ManagementSkills, RiskManagement
	PersonalDevelopment
s_req_skill d_req_skill s_acq_skill d_acq_skill	InterpersonalRelations, ConflictManagement, Teamworking, CareerPlanning, JobSearchSkills, PeopleManagement, TrainingandProfessionalDevelopment, Motivation, PeopleandPerformanceEvaluationSkills, Responsibility
	CommunicationSkills
s_req_skill d_req_skill s_acq_skill d_acq_skill	CommunicationsBasics, CommunicationEthics, InformationManagement, DataManagement, InformationTechnologyBasics, ProductandServiceMarketing, MarketingInformationManagement, StrategicMarketingPlanning
	EconomicSkills
s_req_skill d_req_skill s_acq_skill d_acq_skill	BusinessBasics, BusinessAttitudes, DecisionMaking, EconomicCulture, FinancialBasics, TreasuryManagement, Accounting, EnterpriseModeling, DistributionChannelsManagement, PurchasingManagement, OperationsManagement
	TechnicalSkills
s_req_skill d_req_skill s_acq_skill d_acq_skill	ComputerSkills, ITBasics, ITApplicationsBasics, ElectronicSystemToolsBasics, PaintingSW, CalculationSW, ProjectManagementSW, DocumentManagementSW, PlanningandControlSW, SimulationSW, AccountingSW, CommunicationSW
Keywords	a set of English terms suitably chosen by the user

Table 2: Metadata describing learning objects.

In order to use the machine learning techniques introduced in the next section, we decided to pre-process the metadata Age and the various Skills, so as to recover, through a fuzzification-defuzzification procedure, an enumeration of linguistic terms along the lines of the nominal metadata:

Age. We firstly introduced six linguistic variables: {Young, Teenager, Mature, University, PhD, Adult} corresponding respectively to the age intervals: $\{[6, 12], [10, 16], [15, 21], [19, 26], [24, 30], [28, 37]\}$ described through suitable trapezoidal fuzzy sets. Then to each age interval instantiating the metadata Age in the dataset, we associated a pair of linguistic variables corresponding to the two fuzzy sets where the interval extremes have maximum membership. For instance, if the Age value is $[8, 18]$ we translate it in Young-Mature. If both extremes fall in the same set, then only a linguistic variable will be used;

Skills. Firstly we extended the various Skills metadata by forcing the user to insert a strength coefficient for each linguistic term, i.e. an integer ranging from 0 (meaning absence of the corresponding term) to 10. Then we fuzzified this value through a narrower set of linguistic terms: {Absent, Low, Medium-Low, Medium, Medium-High, High}, through the following mapping:

$$0 \to \text{Absent}, \{1, 2\} \to \text{Low}, \ldots, \{9, 10\} \to \text{High}$$

Each record of the dataset is therefore composed of all the metadata associated to a given resource r_{ijk} (with the exception of the Category attribute which may be deduced by the Skills metadata) together with the course and module *id*s the resource belongs to (in terms of i, j, k respectively). Table 3 reports an excerpt of the dataset used in this report both before and after the pre-processing phase.

(a) before the pre-processing phase.

Cou.	Mod.	Res.	Language	Resourcetype	Category	Difficulty	Age	Format	Time	s_req_skill		...	d_req_skill	s_acq_skill	d_acq_skill	Keywords	Judg.
										CB.	CE.						
1	1	1	Italian	Lecture	Com.Skills	Medium	[21, 28]	Slide	60min.	2	0	...	...	...	...	{a, b}	6
1	1	2	English	Slide	Com.Skills	Medium	[21, 28]	Text	90min.	2	0	...	...	...	...	{q, a}	6
1	1	3	Italian	Exercise	Com.Skills	Medium	[21, 28]	Slide	60min.	2	0	...	...	...	...	{r, b, a, d}	6
1	1	4	English	Diagram	Com.Skills	Medium	[21, 28]	Text	60min.	2	0	...	...	...	...	{a, q}	3
1	2	1	English	Slide	Com.Skills	Difficult	[21, 28]	Slide	90min.	2	0	...	...	...	...	{p, a}	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2	1	2	Italian	Lecture	Com.Skills	Medium	[21, 35]	Slide	90min.	0	4	...	...	...	...	{d, q}	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

(b) after the pre-processing phase.

Cou.	Mod.	Res.	Language	Resourcetype	Difficulty	Age	Format	Time	s_req_skill		...	d_req_skill	s_acq_skill	d_acq_skill	Keywords	Judg.
									CB.	CE.						
1	1	1	Italian	Lecture	Medium	Univ-PhD	Slide	60min.	Low	Absent	...	...	...	...	{a, b}	6
1	1	2	English	Slide	Medium	Univ-PhD	Text	90min.	Low	Absent	...	...	...	...	{q, a}	6
1	1	3	Italian	Exercise	Medium	Univ-PhD	Slide	60min.	Low	Absent	...	...	...	...	{r, b, a, d}	6
1	1	4	English	Diagram	Medium	Univ-PhD	Text	60min.	Low	Absent	...	...	...	...	{a, q}	3
1	2	1	English	Slide	Difficult	Univ-PhD	Slide	90min.	Low	Absent	...	...	...	...	{p, a}	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2	1	2	Italian	Lecture	Medium	Univ-Adult	Slide	90min.	Absent	Medium-Low	...	...	...	...	{d, q}	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Table 3: Excerpt of the dataset used throughout this report.

2 Architecture of the proposed multi-phase Recommender System

The main idea supporting the proposed human-centric algorithm is to guide the teacher in the selection of the didactic resources more suitable to her purposes through several phases, each one aimed at reducing the huge number of available resources. In such a way, the teacher may incrementally focus only on those subsets of resources really pertaining to her courses without being frustrated in browsing the whole dataset.

Due to: i) the heterogeneity of the various courses, modules and lessons; ii) the huge amount of resources appearing in the dataset; and iii) the unknown preference of the teacher in terms of topics covered by the course she is preparing (which are to be inferred through a agnostic paradigm), we split the recommender system functionality into three main parts, according to the schema reported in Algorithm 1. In all phases, it emerges the human-centric feature of the proposed

Algorithm 1 Schema of the proposed Recommender System.

```
1: procedure CREATECOURSE(dataset)
2:     repeat
3:         1. Select the most informative part of the course
4:         2. Choose the metadata mostly relevant to the selected part
5:         3. Select the resources filling the selected part
6:     until the course is complete
7: end procedure
```

Recommender System. In fact, the teacher plays a primary and active role in the design of her course. The system itself is asked to assist the teacher in all her choices, suggesting the most suitable moves she may take at each step of the algorithm. The three phases have been designed exactly to tackle the three main features of the task in question: i) a course is usually composed by many parts; some

of them are indispensable for its success, some others stay in the background (for instance the course introduction and conclusion); aim of the first phase is to automatically discover which are the most relevant parts of a course; ii) the teacher is the only actor which knows the topics of the course; the goal of the second phase is to infer these topics by guiding the teacher in the selection of the most suitable metadata, in the meanwhile avoiding her frustration and dissatisfaction; and iii) the repository may contain a huge amount of resources; aim of the third phase is to help the teacher to select the resources which better suit the course topics in a small number of steps. In the next sections we will deeply investigate the three phases.

2.1 Choice of the most informative part

An entire course may take a long time in order to be fully accomplished. For this reason, and in order to guarantee a flawless organization of it, a course is usually split in several parts. Some of these are more informative than others: for instance it is less probable that both introductory and conclusive parts of a course will provide a substantial contribution to the teaching process. The aim of the first phase is to split each course appearing in the dataset in a same number n_{part} of parts, suitably chosen by the teacher, and to select that part which proves to be more informative under a suitable criterion.

2.1.1 Parts as aggregates of modules

As the number n_{ℓ_i} of modules constituting the i-th course is not fixed within the entire dataset, we have to organize modules in parts, in a number n_{part} which is constant all over the dataset and arbitrarily chosen by each teacher. As such splitting is not unique, we force univocity by imposing the following three intuitive

constraints:

1. each part should contain a same number of modules, equal to $\lceil M/P \rceil$;
2. in order this to be feasible for all pairs $(n_{\ell_i}, n_{\text{part}})$, each module may belong to more than one part;
3. this *overlap* (number of modules shared among more parts) should be minimized.

By forcing each part to contain exactly $\lceil M/P \rceil$ modules, we avoid situations where, for instance, a module approaching the end of a course is inserted in a introductory part. This kind of behavior may be observed in case we decide to relax constraint 1 by letting each part containing a same *arbitrary* number of modules, and strengthen constraint 3 by imposing a same overlap of modules among the parts. Think for instance to the case in which n_{ℓ_i} and n_{part} are coprime, as in these two simple examples: i) $n_{\ell_i} = 9, n_{\text{part}} = 5$ would give rise to the following partition: $\{\{1,2,3,4,5\},\{2,3,4,5,6\},\{3,4,5,6,7\},\{4,5,6,7,8\},\{5,6,7,8,9\}\}$; ii) even worst, $n_{\ell_i} = 13, n_{\text{part}} = 7$ would give rise to: $\{\{1,2,3,4,5,6,7\},\{2,3,4,5,6,7,8\},\{3,4,5,6,7,8,9\},\{4,5,6,7,8,9,10\},\{5,6,7,8,9,10,11\},\{6,7,8,9,10,11,12\},\{7,8,9,10,11,12,13\}\}$. Module number 5 in the first example and module number 7 in the second one belong to all parts, which makes such partitioning useless. By forcing each part to contain exactly $\lceil M/P \rceil$ modules, which in both examples evaluates to 2, we obtain the following two more significant partitions: i) $\{\{1,2\},\{3,4\},\{4,5\},\{6,7\},\{8,9\}\}$; and ii) $\{\{1,2\},\{3,4\},\{5,6\},\{7,8\},\{9,10\},\{10,11\},\{12,13\}\}$. When $P > M$, the same constraints guarantee a final partition where each part is a singleton (it contains exactly one module). For instance, by inverting the values of n_{ℓ_i} and n_{part} in example 1, we obtain: $\{\{1\},\{1\},\{2\},\{2\},\{3\},\{3\},\{4\},\{5\},\{5\}\}$.

The code of the proposed procedure, written in the functional programming language Mathematica[2], is reported for completeness in Program 2.1. Considering both Mathematica abstruse syntax and the technicalities involved in the partition generation, we avoid explaining in details the pseudocode. Rather, we remember that this procedure satisfies the aforementioned constraints; moreover it makes use of random instructions to select the various overlaps, whereas some indeterminacy arises.

2.1.2 Selection of the most informative part

Once subdivided all the courses within the datasets in a same number n_{part} of parts, the next step consists in selecting the most informative one under a suitable criterion. Of course, in selecting such criterion we are limited to the sole information contained in the dataset, i.e. the metadata describing each resource together with the judgment evaluating it. The chosen criterion is based on the following principle:

> The more the metadata associated to a resource are significant, and the more they contain information on the related judgment.

The idea is then to split the dataset in n_{part} subsets, each one containing all the records associated to a given part, and subsequently apply to each subset a machine learning algorithm with the aim of inferring the judgment of each resource on the basis of its metadata[3]. The part whose prediction is more accurate (having therefore a high number of hits in the recall phase) is declared as the most informative one: in other words it will be the candidate for the next phase.

[2] `http://www.wolfram.com/mathematica/`

[3] From here on, we will exclude the keywords from the set of metadata, being them too specific to the tagged resource.

Program 2.1 Mathematica code to generate a partition of M modules in P parts.

```
getPartition[M_, P_] := Block[
    (* local variables *)
    {x = P Ceiling[M/P] - M, a, b},
    If[M >= P,
        (* then *)
        {a, b} = {Floor[x/(P - 1)], Mod[x, P - 1]};
        a = Prepend[Accumulate[Table[a, {P - 1}] + If[b == 0, Table[0, {P - 1}], Total[UnitVector[P - 1, #]
            & /@ RandomSample[Range[P - 1], b]]]], 0];
        Table[Take[Range[M], {1 + i Ceiling[M/P] - a[[i + 1]], (i + 1) Ceiling[M/P] - a[[i + 1]]}], {i, 0,
            P - 1}],
        (* else *)
        {a, b} = {Floor[P/M], Mod[P, M]}; {#} & /@ Sort[Flatten[Append[Table[Range[M], {a}], If[b > 0,
            RandomSample[Range[M], b], {}]]]]
    ]
]
```

As machine learning algorithm we focused on C5.0[4], a well-known tool producing either decision trees or rule sets coupling attributes (in our case metadata) with suitable labels (judgments). In Figure 1 we report a simple decision tree: we used a small subset of metadata for the sake of clarity, remembering that the tree produced by employing all metadata may be composed by more than one thousand of nodes. Each path from the root to a leaf describes which conditions a resource should satisfy in order it to be evaluated through the judgment labeling the leaf. For instance, the first path reads as follows: "*if* a resource has been prepared in Italian and has a textual form, *then* it will be evaluated with 7". It is important to note that each resource satisfies exactly one path from the root to the leaf. However, due to both the criteria used by C5.0 to generate the tree and the pruning technique employed to avoid overfitting (in other words to retain only the ground structural information contained in the dataset, ignoring therefore details covering the single resources), not all resources belonging to the dataset and used to generate the tree will be labeled with the right judgment. In this way, however, the predictive power of the tree when queried on new resources (not used to generate the tree itself) will increase. Not only, the labeling errors on the training set should even increase when the tree is transformed in the corresponding rule set, like the following:

[4]freely downloadable from `http://www.rulequest.com/see5-info.html`

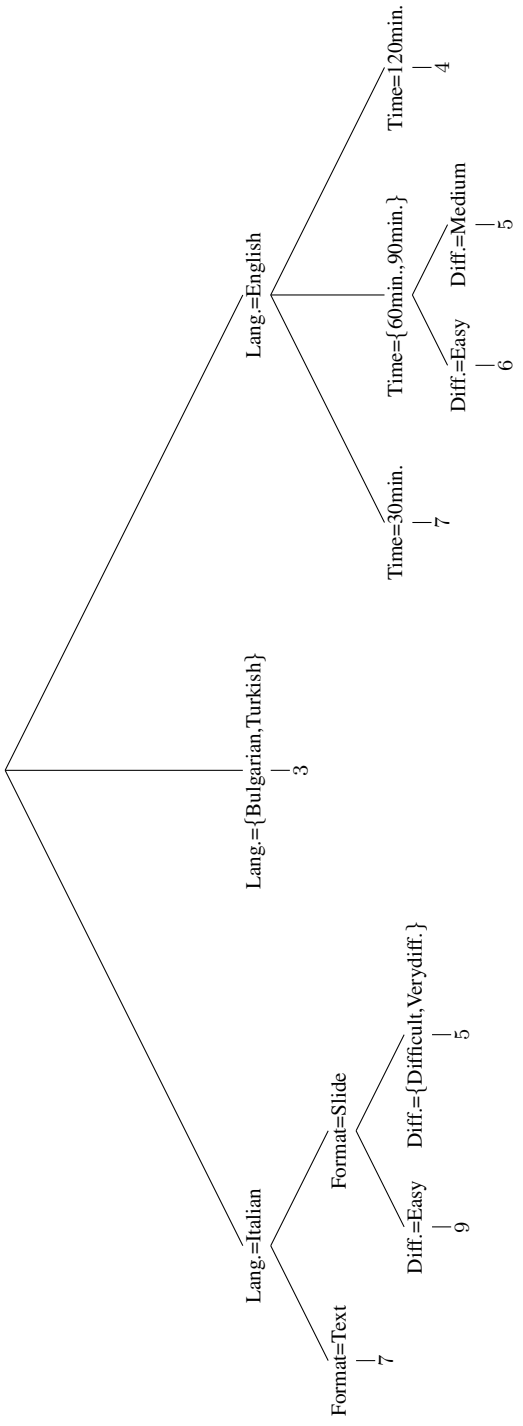

Figure 1: A simple decision tree generated by C5.0.

$$\begin{aligned}
\text{Language} = \text{Italian} \wedge \text{Format} = \text{Text} &\rightarrow 7 \\
\text{Language} = \text{Italian} \wedge \text{Format} = \text{Slide} \wedge \text{Difficulty} = \text{Easy} &\rightarrow 9 \\
\text{Language} = \text{Italian} \wedge \text{Format} = \text{Slide} \wedge \text{Difficulty} = \{\text{Difficult}, \text{VeryDifficult}\} &\rightarrow 5 \\
\text{Language} = \{\text{Bulgarian}, \text{Turkish}\} &\rightarrow 3 \\
\text{Language} = \text{English} \wedge \text{Time} = \text{30minutes} &\rightarrow 7 \\
\text{Language} = \text{English} \wedge \text{Time} = \{\text{60min.}, \text{90min.}\} \wedge \text{Difficulty} = \text{Easy} &\rightarrow 6 \\
\text{Language} = \text{English} \wedge \text{Time} = \{\text{60min.}, \text{90min.}\} \wedge \text{Difficulty} = \text{Medium} &\rightarrow 5 \\
\text{Language} = \text{English} \wedge \text{Time} = \text{120minutes} &\rightarrow 4
\end{aligned}$$

This is because C5.0 implements some heuristics having as goal exactly the inference of easily interpretable rules, hence short (i.e. more general) and in a low number.

This brief and in any case non exhaustive description of C5.0 helps to understand the strategy used to select the *best*, i.e. most informative, part. The idea is to run C5.0 on the portion of the dataset related to the single parts, obtaining a decision tree or its companion rule set, and then test the single predictors on the whole dataset, choosing as most informative the one having highest accuracy. In other words, the most informative part is the one whose resources, through their metadata, have the highest discriminative power w.r.t. their judgment.

2.2 Choice of the most informative metadata

A free benefit of the selection of the most informative phase is the individuation of a subset of modules/resources which may be profitably employed by the teacher to organize her course. Naturally, this subset contains a huge variety of resources, possibly not all pertinent to the topics the teacher is interested in. Aim of this phase is twofold: on the one hand select those metadata *causally* associated with

well-evaluated courses (having therefore a high judgment); on the other hand, let the teacher select the sole metadata covering her needs, narrowing at most her choice basin.

While the second goal may be easily achieved through iterative queries raised to the teacher, the selection of those metadata of utmost importance in the evaluation of the best resources requires again the use of machine learning techniques. In principle, we might use once again the rule set generated by C5.0, isolating those metadata most frequently appearing in the premises of rules leading to good judgment. Unfortunately, the metadata contained in these rules have been selected on the basis of their mutual information with the consequent judgment; this means that it is the judgment itself which plays a primary role in the choice of the metadata. In other words, only those metadata strongly discriminating the various judgments are selected in the rule set generated by C5.0. *Vice versa*, we are interested here in finding some regularities in the metadata emerging from those resources having a good judgment (let's say higher than 7); a task which is usually performed by clustering algorithms. As we further require a *causal* relation between the metadata and the goodness of a resource, we decided to use a technique called *association rule learning*.

It is a popular and highly investigated method for discovering interesting relations among variables in large databases, which works through the identification of strong rules using different measures of interestingness. Formally, letting $I = \{i_1, \ldots, i_n\}$ be a set of attributes called *items*, and $T = \{t_1, \ldots, t_m\}$ a set of transactions, called *transaction set*, where $t_i \subseteq I$, the aim of this technique is to discover rules of the forms $P \rightarrow C$, with $P, C \subseteq I$ and $P \cap C = \emptyset$, corresponding to the most frequent associations in T. To make an example, let $I =$\{good_course, language_Italian, difficulty_easy, time_90min\} and let the transaction set T be defined as follows:

i	t_i
1	{good_course, language_Italian}
2	{difficulty_easy}
3	{time_90min}
4	{good_course, language_Italian, difficulty_easy}
5	{language_Italian}

An illustrative rule could be for instance $\{\text{language\_Italian} \wedge \text{difficulty\_easy}\} \rightarrow \{\text{good\_course}\}$, meaning that if a resource is written in Italian and it is not difficult, then the resource itself is part of a good course.

Concerning the measures of interestingness at the basis of the rule discovering, we decided to use one of the simplest metrics: the *support*. Formally, the support $\text{supp}_\text{T}(X)$ of an item set X is defined as the proportion of transactions in the dataset T which contain X. The support $\text{supp}_\text{T}(P \rightarrow C)$ of the rule $P \rightarrow C$ is defined as the support of the set $P \cup C$. Of course, the higher the support of a rule, and the higher will be the probability that it will be tagged as interesting. The choice of this metric instead of more informative ones (such as confidence, lift, leverage, conviction, ...) was dictated by the peculiarity of our transaction set. In fact, while the item set I is composed of all possible values each metadata may assume, together with the attribute good_course, each transaction $t_i \in T$ contains those metadata related to the i-th resource, providing the latter has a judgment greater or equal to a given threshold (by default set equal to 7, in any case arbitrarily customizable by the teacher). From this set we removed each skill having value equal to Absent, as its meaning is straightforwardly the lack of this skill. Now, as we selected only good courses, all transactions will contain the attribute good_course. Moreover, being interested in rules having the form $\{m_1, \ldots, m_k\} \rightarrow$ good_course, where m_is are suitable metadata, all metrics with the exception of the support will have a constant value equal to 1 within the entire transaction set, proving therefore uninformative.

i	t_i
1	language_Italian, Resourcetype_Lecture, Difficulty_Medium, Age_Univ-PhD, Format_Slide, Time_60min, s_req_skill_CB_Low, ~~s_req_skill_CE_Absent~~, ..., good_course
2	language_English, Resourcetype_Slide, Difficulty_Medium, Age_Univ-PhD, Format_Text, Time_90min, s_req_skill_CB_Low, ~~s_req_skill_CE_Absent~~, ..., good_course
3	language_Italian, Resourcetype_Exercise, Difficulty_Medium, Age_Univ-PhD, Format_Slide, Time_60min, s_req_skill_CB_Low, ~~s_req_skill_CE_Absent~~, ..., good_course
4	...

Table 4: Example of transaction set given in input to Apriori algorithm.

Coming back to the lead example shown in Figure 3 and choosing as threshold the value 6, we obtain the transaction set in Table 4, where the strikethrough text corresponds to erased metadata.

Many algorithms for generating association rules were presented over time. We decided to use one of the best known, Apriori[5] which uses a breadth-first search strategy to count the support of item sets and uses a candidate generation function which exploits the downward closure property of support, making it extremely scalable.

Given in input a dataset as the one listed in Table 4, Apriori returns a set of rules together with their support. We retain only those rules having as consequence the attribute good_course and sort them in decreasing order w.r.t. their support. A typical rule outputted by Apriori is, for instance,

$$0.78 \quad \{\text{language\_Italian, s\_req\_skill\_CB\_High, s\_acq\_skill\_CE\_Medium}\} \rightarrow \text{good\_course}$$

where 0.78 denotes its support. Starting from rules having highest support, we iteratively propose to the teacher the metadata appearing in the antecedent of the rules. She is free to select those metadata related to the topics of the course she is

[5]Many implementations of Apriori have been developed since its first publication; see for instance `http://www.borgelt.net/apriori.html`

called to prepare, until satisfied with the current item set. Of course, at each new iteration a filter is applied so as to propose to the teacher only those metadata she has never processed formerly.

2.3 Selection of the resources constituting the new course

Thanks to the first two phases we were able to strongly downsize the set of resources available to the teacher. However, the metadata used in the previous phase alone are too broad to exactly identify the main topics of interest for the organization of the new course. In fact, we have not yet played all our available cards, having till now ignored the keywords, attributes associated to each resource which, although too specific and arbitrary for the goals imposed in the previous phases, represent now a rich source of information thanks to their circumscribed discriminative power. Aim of this phase is to guide the teacher in the selection of those keywords pertaining to her course, yet avoiding to flood her with a lot of alternatives, hence optimizing her search. As we have no information of which keywords are those sought by the teacher, we decided to use an orthogonal approach w.r.t. the one adopted in the previous phases. In order to shorten the search time, we should in fact identify those sets of keywords which partition the available resources *most uniformly*. In this way, by iteratively proposing to the teacher the polychotomy induced by the optimized set of keywords and awaiting for her selection, the set of interesting keywords will be discovered in a minimal number of cycles. To better illustrate the idea, let introduce this simple example. Consider a dataset composed of 8 resources $\{r_1, \ldots, r_8\}$ each tagged with some keywords belonging to the set $\{a, b, c, d, e\}$. An easy way to visualize this association comes through the binary matrix reported in Table 5. where a 1 in correspondence with resource r_i and keyword k_j means that k_j is an attribute of r_i. Suppose now that

!ht

	a	b	c	d	e
r_1	0	0	1	0	0
r_2	0	0	1	0	0
r_4	0	1	1	0	0
r_6	1	0	1	0	0
r_8	1	1	1	0	0
r_3	0	1	0	0	0
r_7	1	1	0	0	1
r_5	1	0	0	1	1

Table 5: Binary representation of the resource-keyword association

we choose to follow the opposite strategy, i.e. proposing to the teacher the most discriminative (less uniform and more heterogeneous) resource set, awaiting then for her feedback. In Table 6 we show a possible scenario obtained through the adopted policy. Therein we list in each column: i) the number of interaction step;

step	**keyword set**	**teacher choice**	**selected resource**
1	$\{d\}$	$\{\overline{d}\}$	$\{r_1, r_2, r_3, r_4, r_6, r_7, r_8\}$
2	$\{e\}$	$\{\overline{e}\}$	$\{r_1, r_2, r_3, r_4, r_6, r_8\}$
3	$\{c\}$	$\{c\}$	$\{r_1, r_2, r_4, r_6, r_8\}$
4	$\{b\}$	$\{\overline{b}\}$	$\{r_1, r_2, r_6\}$
5	$\{a\}$	$\{\overline{a}\}$	$\{r_1, r_2\}$

Table 6: Interaction with the teacher when adopting the incorrect policy.

ii) the set of keywords chosen by the policy, which is then exploded in all the possible assignments (for instance, starting from the keyword set $\{a, b\}$, the teacher is presented with the set $\big\{\{a, b\}, \{a, \overline{b}\}, \{\overline{a}, b\}, \{\overline{a}, \overline{b}\}\big\}$, where $\overline{k}_i$ means that the teacher is not interested in the keyword k_i); iii) the choice of the teacher, on the basis of her interests; and iv) the underlying resources selected by the teacher, compatible with her choice.

Let us now use the keyword set selection strategy (effectively employed in the proposed algorithm) which partitions the available resources *most uniformly*. Table 7

summarizes the teacher interaction step until she is satisfied with her keyword selection. Of course, this is a prototypical example taken on purpose to the extreme,

step	keyword set	teacher choice	selected resource
1	$\{a, b, c\}$	$\{\overline{a}, \overline{b}, c\}$	$\{r_1, r_2\}$

Table 7: Interaction with the teacher when adopting the correct policy.

in order to show the drastic effects on the interaction time a wrong policy may lead to.

The pseudocode of the proposed algorithm aimed at selecting the best keyword set at each interaction with the teacher is reported in Algorithm 2. In input the keyword set K, the binary matrix M describing the resource-keyword associations (see Table 5), and the maximum number $\max_k$ of keywords a keyword set is allowed to contain (parameter chosen by the teacher), it evaluates each possible subsets of at most $\max_k$ keywords, returning the one having maximum uniformness. The latter is evaluated through the function EVALUATE which in the pseudocode is based on the computation of the normalized entropy[6] of the partition induced by the candidate keyword set on the resources (rows of matrix M). In fact, the more this partition is balanced and the more its entropy will be high. We used the normalized entropy (obtained by dividing the entropy by its maximum value achieved with exactly i keywords) because it allows a comparison with entropic quantities computed on different keyword set sizes.

Note that each time the teacher selects a keyword set, this will be removed from the binary matrix M before calling again Algorithm 2. Moreover, the whole computation ends when the teacher is satisfied with all the previously selected keywords.

[6]Actually, we might have used different indices of uniformness, like the *diversity coefficient* proposed by Gini.

Algorithm 2 Selection of the most uniform keyword set.

procedure KEYWORDSETSELECTION($K, M, \max_k$)
  $best \leftarrow -1$
  $kSet \leftarrow \emptyset$
  for $i \leftarrow 1, \max_k$ **do**
    for all $S \subseteq \mathcal{P}_i(K)$ **do**
      $eval \leftarrow$ EVALUATE(S, M, i)
      if $eval > best$ **then**
        $eval \leftarrow best$
        $kSet \leftarrow S$
      end if
    end for
  end for
end procedure

function EVALUATE(S, M, i)
  $entr \leftarrow 0$
  for all assignments A of S **do**
    $c \leftarrow$ fraction of resources in M satisfying A
    $entr \leftarrow entr - c \log c$
  end for
  return $\frac{entr}{\log 2^i}$
end function