Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Leveraging crowd skills and consensus for collaborative web-resource labeling

Silvana Castano, Alfio Ferrara, Stefano Montanelli *

Department of Computer Science, Università degli Studi di Milano, via Comelico 39, 20135 Milan, Italy

HIGHLIGHTS

- A three-stage approach called CLabel is proposed for enforcing collaborative web-resource labeling.
- CLabel is characterized by the disciplined combination of automatic tools/techniques and crowdsourcing.
- CLabel is capable of dealing with complex crowdsourcing processes where multiple task typologies, including create-questions, are required.
- CLabel leverages on crowd preferences and consensus for capturing the different interpretations that can be associated with a considered web resources.

ARTICLE INFO

Article history:
Received 24 April 2017
Received in revised form 14 November 2017
Accepted 22 December 2017
Available online xxxx

Keywords:
Crowdsourcing
Consensus-based web-resource labeling
Task design

ABSTRACT

In this paper, we propose a three-stage approach called CLabel for enforcing collaborative web-resource labeling in form of a crowdsourcing process. In CLabel, the results of both crowdsourcing and automated tasks are combined into a coherent process flow. CLabel leverages on crowd preferences and consensus, for capturing the different interpretations that can be associated with a considered web resource in form of different candidate labels and for selecting the most agreed candidate(s) as the final result. CLabel succeeds to be particularly appropriate for application to labeling problems and scenarios where human feelings and preferences are decisive to select the answers (i.e., labels) supported by the majority of the crowd. Moreover, CLabel succeeds in providing label variety when multiple labels are required for a suitable resource annotation, thus avoiding duplicate or repetitive labels.

A real case-study of collective web-resource labeling in the music domain is presented, where we discuss the task/consensus configuration and obtained labels as well as the results of two specific tests, respectively devoted to the analysis of label variety, and to the comparison of CLabel results against a reference classification system, where music resources are labeled using predefined categories based on a mix of social-based and expert-based recommendations.

1. Introduction

In the recent years, crowdsourcing has gained a growing popularity in many application domains concerned with resource labeling, like for example item classification [1], argument discovery [2], and entity linking [3], where the use of automatic tools/procedures is not completely effective nor possible or satisfactory, and the contribution of human workers becomes decisive for improving the quality of final results. According to a widely-accepted definition proposed in [4], *crowdsourcing* is defined as "a type of participative on-line activity in which an individual, [. . .], proposes to a group of workers of varying knowledge, heterogeneity, and number, via a flexible open call, the voluntary undertaking of a task".

By focusing on crowdsourcing for resource labeling, the idea is to use *decide-question* tasks where the crowd workers are asked to choose the preferred label among a set of candidate tags generated through some automated procedure [1,5,6]. The aim is to rely on human understanding for selecting the most representative label(s) of a resource among those given in the task as possible choices. However, solutions in this field are negatively affected by two main limitations. First, the automatically-generated tags submitted to the crowd for evaluation are usually inadequate either due to poor metadata and/or description of the resource to label, that causes non-featuring, generic tags to be proposed to the crowd for choice. Second, the inability of automated procedures to be creative and original produces a set of repetitive tags not capturing the possibly-different resource interpretations.

* Corresponding author.
E-mail addresses: silvana.castano@unimi.it (S. Castano), alfio.ferrara@unimi.it (A. Ferrara), stefano.montanelli@unimi.it (S. Montanelli).
URLs: http://islab.di.unimi.it/castano (S. Castano), http://islab.di.unimi.it/ferrara (A. Ferrara), http://islab.di.unimi.it/montanelli (S. Montanelli).

https://doi.org/10.1016/j.future.2017.12.024
0167-739X/

Furthermore, resource labeling solutions using crowdsourcing require a mechanism for assessing the reliability of worker answers to ensure that the label chosen for a resource can be *really* considered as the most representative among those available. The notion of *consensus* is frequently employed in crowdsourcing applications and systems as a solution for quality assessment of task results, either based on an explicit or implicit notion of worker agreement. Usually, consensus is described as an emerging property that could only be obtained by combining together multiple contributions provided by workers [7,8]. In most of the existing work, the focus is mainly on defining technical solutions aimed at ensuring a "fair" consensus calculation among the workers involved in the execution of a labeling task. However, for effective design of labeling tasks, the key point is not only to enforce adequate flexibility in configuring the relevant task features (such as the number of workers to involve and the consensus mechanism to exploit for resulting label evaluation), but also the capability to support articulated crowdsourcing processes where tasks with different features and configurations are required and need to be managed in a seamless way.

In this paper, we present a three-stage approach called CLabel based on crowdsourcing techniques and consensus mechanisms for enforcing collaborative web-resource labeling. By collaborative web-resource labeling, we refer to a crowdsourcing-based process by which the final label assigned to a web resource depends on the consensus (i.e., degree of agreement) reached by human workers engaged to this purpose. In addition to conventional decide-question tasks for label selection, CLabel is characterized by the use of create-question tasks for eliciting candidates directly from the crowd thus enforcing active participation and contribution of the workers by bringing their subjective points of view, creativity, and human understanding into the labeling process. Furthermore, CLabel combine crowdsourcing tasks and automated tools into a coherent process flow to enable label normalization and a fine, multi-perspective labeling web resources. A multi-dimensional design framework is exploited to support the specification of both decide-question and create-question tasks for a given labeling problem and for consensus management and evaluation. Consensus evaluation on executed tasks is employed to measure the formation of a shared user agreement about the final, definitive labels chosen for describing a given web resource. The music-emotion case-study is finally presented for collective web-resource labeling in the music domain to discuss the application of CLabel and related task/consensus design functionalities in a real scenario.

The paper is organized as follows. The related work as well as the original contribution of the proposed CLabel approach are discussed in Section 2. In Section 3, we present the CLabel approach for collaborative web-resource labeling. Sections 4 and 5 illustrate the crowdsourcing framework and the labeling techniques featuring the CLabel approach, respectively. Setup and configuration of CLabel in the Argo crowdsourcing system is discussed in Section 6. Experimental results are provided in Section 7. Finally, concluding remarks are given in Section 8.

2. Related work

Crowd labeling issues are mostly associated with the execution of decide-question tasks in which workers are expected to choose a tag for an item among a list of predefined options [1,5,6]. The use of create-question tasks is marginally envisaged as a possible contribution on crowdsourcing approaches and only a few research work have been proposed in this respect. In [9], an approach is presented for creating a taxonomy by relying on crowd workers that are asked to propose categories from scratch that finally contribute to generate the resulting taxonomy. In [10], an approach for crowd-based text document summarization is introduced, where workers are asked to shorten a sentence extracted from a text document. Other relevant work are about composite crowdsourcing approaches where the effort of a worker group is exploited to generate new tasks to be executed by other worker groups [11,12].

A crucial aspect in crowdsourcing applications is the trade-off between the quality of the obtained results and the cost of the crowdsourcing execution. In the last few years, a growing number of approaches are appearing where statistical techniques are implied to estimate the optimal number of repetitions of the same task in order to obtain high-quality results keeping low the costs of the crowdsourcing execution. In [13], an accurate study on advantages of task-repetitions is presented as well as novel techniques for estimating the "uncertainty" of a result that are used to decide whether task-repetitions are valuable or not. In [1], the authors focus on the problem of minimizing the total number of task-repetitions by relying on the estimation of the crowd-quality, in order to obtain a correct label with a predefined probability. In [14], a statistical model is presented where "self-reported worker confidence" are taken into account to reduce the need of task-repetitions. Recent work are being focused on how to aggregate multiple worker answers about a given task with the aim to derive a final comprehensive result. For example, in [15], a *Bi-Layer Clustering* (BLC) algorithm is presented to extract high-quality labels from a large set of (potentially) low-quality labels provided by crowd workers. The idea is that instances are inserted in a physical layer according to labels collected from the crowd. Then, labels are connected with instances of a conceptual layer through the BLC algorithm according to predefined conceptual features. As a further example, in [16], a framework named *Label-Aware Autoencoders* (LAA) is proposed to aggregate crowd wisdoms. LAA aims at recognizing "true labels" among those provided by crowd workers through inference and unsupervised mechanisms. A similar solution is discussed in [17] where topic modeling techniques based on the Latent Dirichlet Allocation are exploited for inference-based detection of "true labels".

Further strategies that are commonly employed to improve the quality of crowdsourcing results is to evaluate the quality of the workers involved in task execution. In addiction to techniques based on statistical models (e.g., [18,19]), techniques based on *gold-tasks* are also emerging, where the idea is to evaluate the worker trustworthiness by observing the answers provided to a placement test composed of gold tasks whose correct answers are known in advance [20–22]. Moreover, *peer-review* techniques are also adopted based on manual reviews the tasks executed by workers. The reviews are provided by an administrator or by other trusted workers themselves [23,24]. The idea is to infer the worker trustworthiness by exploiting the quality of executed tasks, the drawback of peer-review techniques consists in a system overhead that is introduced due to the effort required to reviewers to validate the worker activities. Recently, the idea to analyze and learn the worker skill/expertise from the answers given to previously-executed tasks is also being emerging to enforce a profile-based task routing approach. Some preliminary result of this kind is presented in [25,26].

Original contribution. With respect to the above solutions, CLabel provides two main contributions:

Use of create-question tasks in a composite, disciplined crowdsourcing process. As a difference with most of the state-of-the-art approaches, a peculiar aspect of CLabel is the use of crowdsourcing and create-question tasks to enforce label elicitation, so that different resource interpretations recognized by crowd workers can emerge based on different human expertise, creativity, and subjective point of view. This aspect is particularly relevant in real application scenarios where many different labels can be considered as appropriate and a set of predefined candidates (to

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.

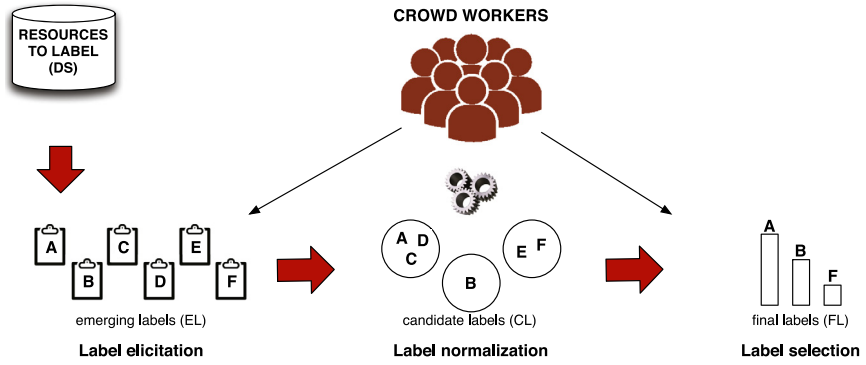

Fig. 1. The CLabel approach.

be listed as choices in decide-question tasks) is not available nor easy to determine with automatic tools/procedures (e.g., argument discovery, multimedia resource annotation). A further distinguishing feature of CLabel is about the capability to deal with the *variety* of the resulting set of resource labels. In CLabel, different labeling modalities (i.e., single, multiple) are supported, allowing to choose the number of labels that is required for any considered resource. In the state-of-the-art solutions, multiple resource labeling is supported as well, by returning those answers that obtained equal preferences by crowd workers. However, quality checking on the overall composition of the label set is not enforced, then it is possible that the result contains label repetitions/synonyms due to similar worker answers. In this respect, in CLabel, the use of a composite, disciplined labeling process characterized by the combination of crowdsourcing with automatic tools/techniques for normalization/classification of crowd preferences allows to effectively deal with the possible "noise" of label redundancies, mistypings, and misunderstandings that typically occur in automated labeling systems. Consequently, when multiple labels are returned for a given resource, they have different meaning and/or they represent a different aspect of the resource.

Use of consensus mechanisms in a configurable task-design framework. Quality assessment of crowdsourcing results is enforced in CLabel by measuring the crowd consensus on available candidate answers according to the notion of worker trustworthiness. As a difference with the state-of-the-art approaches, no a-priori knowledge and no training activity are required. This means that CLabel is capable (i) to select the most appropriate task force involved in the task execution based on system-managed trustworthiness constraints, and (ii) to progressively evolve the worker trustworthiness according to the quality of executed tasks without requiring any reviewing activity as occurs in peer-review and gold-task solutions. The use of a configurable task-design framework is an additional featuring aspect of CLabel based on the capability to setup both the task features (i.e., task-type, consensus-mode) and the execution features (i.e., task-force, worker-trustworthiness, quorum-majority). This way, it is possible to dynamically specify (and supervise) the tradeoff between the desired degree of accuracy of obtained resource labels and the corresponding crowdsourcing costs to be covered.

3. Collaborative web-resource labeling

Given a dataset *DS* of web resources to label, like for example a repository of images, multimedia documents, or object descriptions, the CLabel approach (Fig. 1) enforces a collaborative web-resource labeling process articulated in a sequence of three main steps called label elicitation, label normalization, and label selection.

Label elicitation. This is a crowd-supported step and it is responsible of collecting the *emerging labels* EL_{wr} proposed for describing a considered web resource $wr \in DS$ ($wr \mapsto EL_{wr}$). A worker receives an *elicitation task* containing the web resource wr with a possible associated description and she/he is asked to free-formulate her/his favorite label from scratch for wr. Multiple workers are involved in the execution of the same elicitation task with the aim to foster the emergence of many labels reflecting different interpretations/understandings of the resource wr. Label elicitation represents a core feature of CLabel, since it stimulates the acquisition of a variety of label options from the crowd itself without manual or tool-supported generation of predefined choices.

Label normalization. This is a tool-supported step and it is responsible of transforming the emerging labels EL_{wr} into a set of candidate labels CL_{wr} ($EL_{wr} \mapsto CL_{wr}$). This step aims at reducing the number of possible candidates by removing duplicate or synonym/similar labels among those collected from independent workers. In detail, it is possible that two workers independently provide the same/synonym label, meaning that they have a similar understanding of the considered web resource wr. On the opposite, it is also possible that two workers provide completely different labels, meaning that they have a different perception of wr. The goal of label normalization is twofold. On the one side, the resulting set CL_{wr} has to be *representative* as much as possible, meaning that all the different wr understandings belonging to EL_{wr} have to be included in CL_{wr}. This way, all the original worker contributions have the chance to become the final label FL_{wr}. On the other side, the set CL_{wr} has to be *synthetic* as much as possible, meaning that a candidate label $cl \in CL$ is the result of a similarity-based aggregation procedure where duplicate/synonym labels collected from workers are normalized and grouped.

Label selection. This is a crowd-supported step and it is responsible of selecting the final label FL_{wr} among the set of candidate labels CL_{wr} ($CL_{wr} \mapsto FL_{wr}$). The role of crowd workers is twofold. First, crowd workers are involved in the execution of *candidate-choice tasks*, where the goal is to select the favorite label *cl* among the candidates in CL_{wr}. Second, crowd workers are involved in the execution of *representative-choice tasks*, where the goal is to select the favorite label *el* among the elements composing the candidate $cl \in CL$ selected in the candidate-choice task. In both tasks, multiple workers are involved with the aim to capture the majority feeling of the crowd. However, in a CLabel execution, the workers involved in label selection of a resource wr are different from those involved in label elicitation to avoid possible bias in the choice of the favorite option within CL_{wr}. Consensus evaluation techniques are employed to determine the option with the majority of preferences within the set of voting workers.

4. The crowdsourcing framework for CLabel task design

Different types of crowdsourcing task are supported in CLabel. In particular, a creation task is required in the label elicitation step

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.

to enable workers to express their own creativity when proposing resource labels. On the opposite, a decision task is required in the label selection step to choose the most appropriate label among the proposed candidates. For specification of all the relevant parameters related to task and crowd-consensus evaluation, CLabel relies on task-design activities borrowed from the multi-dimensional modeling framework presented in [27].

4.1. Designing CLabel tasks

Task-design is articulated in two different steps called *task-feature specification* and *task-execution specification* as described in the following.

Task-feature specification. Task-design first requires to specify the main task features for what concerns (i) the nature of the task request to submit to the worker, and (ii) the mechanism to employ for crowd-consensus evaluation out of all the answers supplied by the involved workers. Given a task T to execute, feature design requires the specification of $T_f = \langle tt, cm \rangle$, where tt is the task-type and cm is the consensus-mode.

Task-type. This feature enables to define the nature of the task T to execute, based on the kind of worker contribution that is required for accomplishing the task. Possible task types are proposition and choice. Proposition denotes a task request where the worker has to formulate (i.e., propose) an answer from scratch as result of task execution. Choice denotes a task request where the worker has to select her/his answer among a set of predefined alternatives.

Consensus-mode. This feature allows to specify the mechanisms used for determining the task result based on the different task answers returned by the different involved workers. Possible consensus-evaluation modalities are equivalence and collection. Equivalence denotes a mechanism based on simply counting the number of identical worker answers to determine the task result. Given the set of answers A, the task result $\bar{A}$ is the worker answer with the highest frequency in A. This modality is appropriate when the choice task-type is selected, since answer options are predefined and answer variability is not possible. Collection denotes a modality where the task result is a set containing all the answers provided by the workers involved in the task execution. This modality is appropriate when the proposition task-type is selected, so that all the answers provided by workers are collected and taken into account.

Task-execution specification. After feature specification, task-design requires task-execution specification to configure parameters concerned with (i) the number and the reputation of the workers to involve in a task execution, and (ii) the degree of agreement required for considering a task as successfully completed. Given a task T, task-execution design requires the specification of $T_e = \langle tf, wt, qm \rangle$, where tf is the task-force, wt is the worker-trustworthiness, and qm is the quorum-majority.

Task-force (tf). This parameter is used to specify the number of workers $|W| = k$ to be involved in the execution of a certain task. In a task-force, each worker autonomously executes a received task and independently produces the answer according to her/his personal problem-understanding and expertise. A worker $w \in W$ is not aware of other workers being involved in the execution of a certain task, and the composition of the set W changes from one task to another to avoid mutual and history-based influence among workers.

Worker-trustworthiness (wt). This parameter allows to specify the minimum level of reputation (i.e., *trustworthiness*) $t_{min} \in [0, 1]$ required to a worker w for participating to the task-force involved in the execution of a task T. At the beginning of the crowdsourcing activities, a worker w has an initial trustworthiness value $t_w = t^0 \in [0, 1]$ which is periodically updated (e.g., after a certain bulk of task executions) to capture the capability of the worker w to successfully complete tasks, and thus to support the formation of an agreement on the result of executed tasks by providing answers that are shared with (the majority of) other workers. Only workers with a trustworthiness value $t_w \geq t_{min}$ can be involved in the execution of a task, meaning that the worker reputation t_{min} constitutes a lower bound on t_w under which the worker w is banned from the crowdsourcing activities and it is excluded from task assignments. The trustworthiness value is a system-level parameter managed by the crowdsourcing platform in a transparent way. Further details about the updating mechanisms of worker trustworthiness during the execution of crowdsourcing tasks are provided in [28].

Quorum-majority (qm). This parameter allows to specify the *quorum* $q \in [0, 1]$ representing the minimum percentage of workers in the task-force W that must agree on the task result for reaching the crowd-consensus, and thus the successful task completion.

4.2. The Argo crowdsourcing system

The CLabel approach has been enforced within the Argo crowdsourcing system [29] articulated in four system components called *task manager*, *consensus manager*, *reward manager*, and *uncommitment manager* (see Fig. 2). The **task manager** is in charge of dealing with all the aspects related to task operation by relying on the specified task-type and task-force parameters. In particular, the task manager has the role to prepare/compose the set of tasks to be executed and to assign them to workers for execution. For assignment, the task manager interacts with the reward manager for selecting the workers that satisfy the possible constraints on the worker-reputation parameter. In addition, the task manager is responsible of collecting and storing the task answers created by the crowd workers. The **consensus manager** has the responsibility to evaluate the level of agreement over the task answers received from the crowd workers. For consensus evaluation, the consensus manager relies on consensus-mode and quorum-majority parameters specified in task-design. The **reward manager** is in charge of periodically updating the worker reputation/trustworthiness based on the provided task answers by relying on the specified worker-reputation parameter. The **uncommitment manager** has the role to deal with tasks that are *uncommitted*, namely tasks that do not satisfy the consensus constraints specified in the consensus-mode.

5. Enforcing collaborative web-resource labeling

To enforce collaborative web-resource labeling, the CLabel techniques rely on a predefined configuration used for setup of parameters required in each stage of the labeling process. The predefined configuration contains the parameters of crowdsourcing tasks to be executed in label elicitation and label selection, as well as the similarity threshold of the clustering algorithm employed in label normalization. If required, the predefined configuration can be customized to better capture the specific needs of the case-study at hand. Once the task execution starts, configuration parameters, either predefined or customized, are exploited as constant values. Each CLabel stage is described in the following.

5.1. Label elicitation

This stage is characterized by the execution of crowdsourcing activities based on the specification of a *label-elicitation task*. Given a web resource $wr \in DS$, a label-elicitation task T^E is created with the following predefined configuration (note that the values

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.

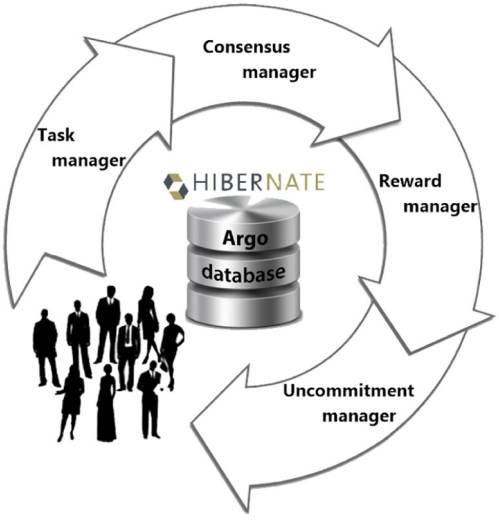

Fig. 2. The Argo system architecture.

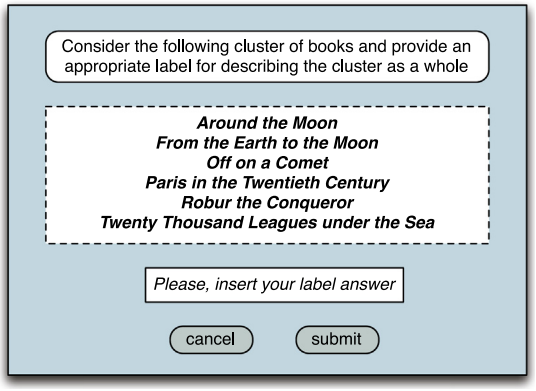

Fig. 3. An example of elicitation task.

Worker	Worker answer	Worker	Worker answer
w_1	adventure novels	w_6	novels
w_2	j. verne	w_7	novels by Jules Verne
w_3	science fiction	w_8	Jules Verne
w_4	Verne	w_9	verne
w_5	Science fiction	w_{10}	Jules Verne

Fig. 4. An example of emerging labels.

of crowdsourcing parameters in the predefined configuration have been determined according to the experimental results obtained on previous case studies [27]):

$T_f^E = \langle tt = \text{proposition}, cm = \text{collection} \rangle$

$T_e^E = \langle tf = 10, wt = 0.0, qm = \text{-} \rangle$.

Label-elicitation is an instance of create-question task for which it is crucial to foster the collection of a large number of different and (possibly) original labels for the considered web resource wr, thus encouraging the crowd creativity to emerge. For this reason, the proposition task-type and the collection consensus-mode are set in the predefined configuration. An original contribution can be generated by any active worker, and possible poor-quality contributions provided by inaccurate workers are eventually discarded in subsequent steps of CLabel. For this reason, the predefined configuration is characterized by a broad task-force ($tf = 10$) where any worker can be involved independently from her/his reputation ($wt = 0.0$). Borrowing the approach proposed in [9], the size of the task-force for the elicitation tasks has been defined through experimentation on real case-studies. The value $tf = 10$ represents a tradeoff between the need to retrieve enough labels for allowing the emergence of all the different resource interpretations, and the aim at avoiding the execution of unnecessary crowdsourcing tasks. In elicitation tasks, the quorum-majority parameter is not specified since consensus evaluation is not performed when the collection consensus-mode is enforced. This means that all the labels provided by the workers in the involved task force are collected and inserted in the set of emerging labels EL. The result of label elicitation is a set $EL_{wr} = \{el_1, \ldots, el_k\}$ (with k corresponding to the size of the task-force tf).

Example. Consider a web resource to label wr_A consisting in a group of book titles about novels by Jules Verne. The layout of the elicitation task T^E for wr_A is shown in Fig. 3. Receiving the elicitation task T^e to execute, each worker in the task-force provides her/his own label answer. An example of emerging labels EL_A is shown in Fig. 4.

5.2. Label normalization

This stage is characterized by the execution of classification techniques with the aim to transform the emerging labels EL into a set of candidate labels $CL = \{cl_1, \ldots, cl_h\}$ (with $h \leq k$). The goal is that all the similar emerging labels representing a common resource interpretation are put together into a single candidate label. As a result, on the one side, it is possible that a candidate label $cl \in CL$ is a singleton cluster $cl = \{el\}$ containing just one label that coincides with an emerging label $el \in EL$. In this case, el provides an original web-resource interpretation dissimilar to all the other emerging labels in $EL \setminus \{el\}$. On the other side, it is also possible that a candidate label $cl \in CL$ is a label cluster $cl = \{el_1, \ldots, el_l\}$ where multiple emerging labels are included in cl based on their mutual similarity. In this case, the labels $el_1, \ldots, el_l$ all provide a common web-resource interpretation and they are aggregated to originate a single candidate. The set of candidate labels CL is a hard clustering of the initial set of emerging labels EL, namely $\bigcup_{i=1}^{i=h} cl_i = EL$, and $\nexists\, cl_i, cl_j \in CL \mid cl_i \cap cl_j \neq \emptyset$.

The set of candidate labels CL is generated by exploiting a hierarchical clustering algorithm of agglomerative type based on the initial set of emerging labels EL. First, standard text-normalization techniques (i.e., tokenization, whitespace trimming, capital-letter reduction, and stemming) are applied to extract a canonical form of the emerging labels EL. Given $k = tf$ the size of the task force involved in label elicitation, a similarity matrix S of size k is built upon the text-normalized labels of EL by exploiting techniques for approximate string matching, like for example I-Sub, Q-Gram, Edit-distance, or Jaro–Winkler [30]. Hierarchical clustering is invoked to transform the labels EL into candidate labels CL according to the similarity results in S. Agglomerative refers to the fact that candidates are obtained through a series of successive merging operations over the emerging labels. Hierarchical refers to the fact that groups of similar labels are organized into a tree according to decreasing matching values of labels starting from the tree leafs up to the root.

The **hierarchical clustering algorithm** HC is shown in Fig. 5. Call $S[j, i] = S[i, j] = sim(el_i, el_j)$ the similarity value between the pair of labels $el_i, el_j \in EL$. Initially, a candidate label cl_i is created for each emerging label $el \in EL$. The two candidates cl_i and cl_j having the highest similarity value in S are merged by performing

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.

Procedure HC
Input: a set of emerging labels EL
Output: a set of candidate labels CL

Let k be the number of emerging labels in EL
1. Compute $sim(el_i, el_j)$ for any possible pair of emerging labels (el_i, el_j)
for $i := 1$ to k **do**
  $S[i, i] := 1$
  for $j := 1$ to k **do**
    $S[j, i] := S[i, j] := sim(el_i, el_j)$
  end for
end for
2. Create a candidate label cl_i for each el_i
3. /* Merge clusters until the matrix is empty */
repeat
  Select cl_i, $cl_j \mid S[i, j] = max_{s,t}(S[s, t])$
  $cl_i = cl_i \cup cl_j$
  for $l := 1$ to k **do**
    if $l \neq j$ **then**
      $S[i, l] := S[l, i] := min(S[l, i], S[l, j])$
    end if
    Update S by deleting row and column corresponding to cl_j
  end for
  $k := k - 1$
until $k = 1$

Fig. 5. The hierarchical clustering algorithm *HC*.

the union operation (i.e., $cl_i = cl_i \cup cl_j$). The row and the column of the newly defined candidate cl_i is updated in S by determining the similarity values between cl_i and each remaining candidate cl_l in S. After that, the row and the column of the candidate cl_j is deleted from the matrix S. The clustering procedure terminates when the dimension of S is 1. To calculate the similarity value between two candidates cl_i and cl_l with $cl_l \neq cl_j$, we rely on a *complete-link* strategy [31]. With complete-link, the similarity value between cl_i and cl_l, that is $S[i, l]$ in the matrix S, is defined as the minimum similarity value that cl_i and cl_j have with cl_l (i.e., $S[i, l] = S[l, i] = min\{S[l, i], S[l, j]\}$). We choose to adopt the complete-link strategy since it tends to produce small, homogeneous candidates where a minimum level of similarity is ensured to any pair of labels in a generated candidate cl_i. This is a desirable property of CLabel, since it enables to create well-separated candidate labels within *CL*. The

$\mathsf{CL}_A = \{$
$cl_1 = \{$science fiction, Science fiction$\}$
$cl_2 = \{$adventure novels, novels$\}$
$cl_3 = \{$Jules Verne, Jules Verne, novels by Jules Verne, j. verne, Verne, verne$\}$
$\}$

Fig. 7. An example of candidate labels.

result of the clustering algorithm is a similarity tree where (i) a leaf corresponds to an emerging label *el*, and (ii) an intermediate node *n* represents a "virtual candidate" containing the emerging labels belonging to the sub-tree rooted in *n*. A virtual candidate cl^{sim} denotes the sub-tree rooted in the intermediate node *n* tagged with the similarity value *sim*. A similarity threshold $th_{sim} \in (0, 1)$ is used to specify a minimum degree of similarity that a virtual candidate has to provide for being selected as candidate label $cl \in CL$ to be returned in the clustering result. The similarity threshold th_{sim} enables to determine the required homogeneity degree of the generated candidates. In the predefined configuration, the Q-Gram similarity function is set for approximate string matching and the similarity threshold is set to $th_{sim} = 0.8$, which guarantees highly-homogeneous label clusters (note that the value of the similarity threshold in the predefined configuration has been determined through experimentation and tuning based on our experience in web-data matching and classification [32]).

Example. Consider the emerging labels EL_A crowd-generated during label elicitation for the web resource wr_A. Fig. 6 shows the similarity tree resulting from the execution of the hierarchical clustering algorithm on the labels of EL_A. The set of candidate labels CL_A shown in Fig. 7 is generated as a result.

5.3. Label selection

This stage is characterized by the execution of crowdsourcing activities articulated in two sequential steps called *candidate choice* and *representative choice*.

Candidate choice consists in the crowd execution of a *candidate-choice task* for selecting the favorite candidate label $cl \in CL$. This step is required when more than one candidate label *cl* results from the label normalization stage (i.e., $|CL| > 1$). A candidate-choice task T^{CC} is created with the following predefined configuration (as for label elicitation, note that the setup of crowdsourcing parameters in the predefined configuration have

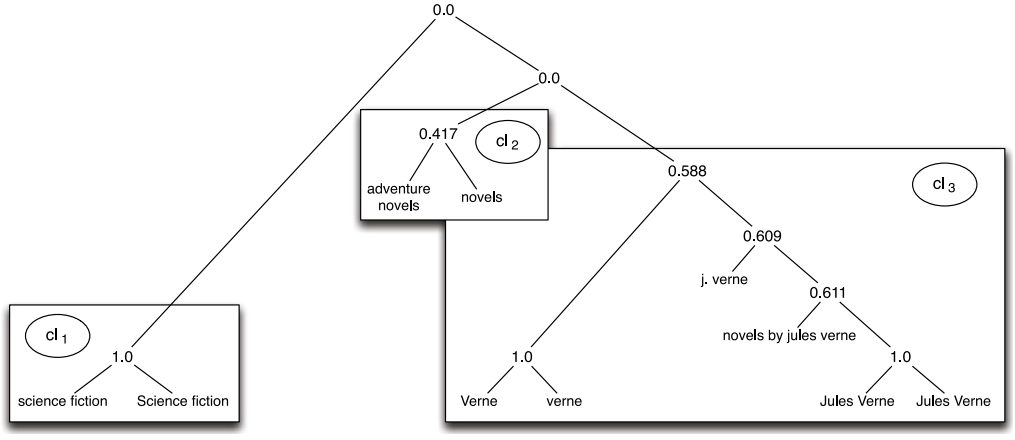

Fig. 6. Example of label normalization and resulting candidate labels.

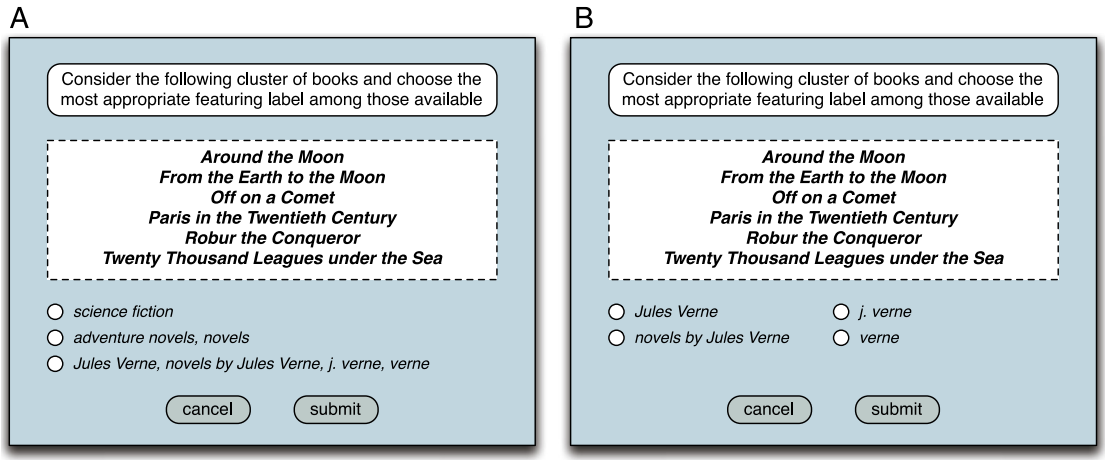

Fig. 8. An example of (A) candidate-choice and (B) representative-choice task.

been determined according to the experimental results obtained on previous case studies [27]):

$T_f^{CC} = \langle tt = \text{choice}, cm = \text{equivalence} \rangle$

$T_e^{CC} = \langle tf = 6, wt = 0.4, qm = 0.65 \rangle$.

Candidate-choice is an instance of decide-question task, then the choice task-type and the equivalence consensus-mode are set in the predefined configuration. A narrow task-force (i.e., predefined value $tf = 6$) composed with workers with (at least) middle degree of trustworthiness (i.e., $wt = 0.4$) is required in the predefined configuration, where the need of a qualified majority is also specified to commit an agreement on the final task result (i.e., $qm = 0.65$). This is a strict configuration conceived to consider a task as successfully completed only when a strong agreement is reached, thus limiting the likelihood that a fortuitous agreement among few, non-expert workers can lead to inaccurate task results. In a choice task-type, the possible answer options $AO = \{ao_1, \ldots, ao_h\}$ are predefined and the crowd worker has to select the preferred answer among those provided. In particular, a different answer option is set in the candidate-choice task for each candidate label $cl \in CL$ (i.e., $ao_i = cl_i \forall i \in [1, h]$). The task result is the answer option (i.e., the candidate cl_s) around which the crowd agreement is obtained after consensus evaluation.

Representative choice consists in the crowd execution of a *representative-choice task* for determining the favorite label among those aggregated within the candidate label $cl_s \in CL$ previously selected in candidate choice. This step is required when more than one label el belongs to the selected cl_s. A representative-choice task T^{RC} is a further instance of decide-question task and the predefined configuration is the same of candidate-choice task, namely:

$T_f^{RC} = \langle tt = \text{choice}, cm = \text{equivalence} \rangle$

$T_e^{RC} = \langle tf = 6, wt = 0.4, qm = 0.65 \rangle$.

In a representative-choice task, the set of answer options AO is defined on the basis of the labels contained in cl_s. Given $cl_s = \{el_1, \ldots, el_l\}$, a different answer option is provided for each label of cl_s (i.e., $ao_j = el_j \forall j \in [1, l]$). Consensus evaluation is used to determine the task result, and thus the final label FL_{wr} for the considered web resource wr.

Consensus evaluation techniques. For both candidate- and representative-choice tasks, the equivalence consensus-mode is selected to specify how to verify whether an agreement among the involved crowd workers has been reached, and thus whether the task can be considered as successfully completed. In CLabel, consensus evaluation is performed through the use of a weighted-voting mechanism called *supermajority* where the answer of each worker w in a task-force is weighted according to her/his trustworthiness values t_w [28]. For a task T, either candidate-choice or representative-choice task, the supermajority mechanism is implemented via the q-constraint and bop-constraint verification as follows.

We call *1st-crowd-answer* ca^1 the top-voted option within the set AO. We call W_{ca^1} the *supporters* of ca^1, namely the subset of workers in the task force that chose ca^1 as answer to the assigned task T. The answer ca^1 becomes the task T result $\bar{A}$ iff the following two constraints are satisfied.

Q-constraint. It is the *quorum constraint* to verify that the 1st-crowd-candidate ca^1 has enough weight (i.e., trustworthiness) for considering the task T as successfully completed, namely:

$$\sum_{w \in W_{ca^1}} t_w \geq q \cdot \sum_{w \in W} t_w$$

where $\sum_{w \in W_{ca^1}} t_w$ is the trustworthiness of the ca^1 supporters, and $\sum_{w \in W} t_w$ is the trustworthiness of all the workers in the task force involved in the task T execution.

Bop-constraint. It is the *balance-of-power* constraint to verify that a single worker cannot determine the final task result just by her/his own answer, thus limiting the influence of workers with high trustworthiness on the overall consensus evaluation process. The bop-constraint is checked by verifying that it is not possible to satisfy the q-constraint by moving a worker in W_{ca^1} to support another available answer option of AO, namely:

$$\sum_{w \in W_{ca^2}} t_w + t_w^{max} < q \cdot \sum_{w \in W} t_w$$

where ca^2 is the *2nd-crowd-answer*, that is the 2nd-voted option within the set AO, and t_w^{max} is the maximum trustworthiness value in the task-force.

When the supermajority constraints are not satisfied, the result of the task T is *uncommitted*. Management strategies for uncommitted tasks will be discussed in Section 6.

Example. Consider the set of candidate labels CL_A obtained from label normalization for the web resource wr_A. The layout of the candidate-choice task T^{CC} is shown in Fig. 8(A). For candidate-choice, the set of answer options is defined as $\mathsf{AO}_A = \{\{\text{science fiction}\}, \{\text{adventure novels, novels}\}, \{\text{Jules Verne, novels by Jules Verne, j. verne, verne}\}\}$. The set of

Answers to the task T^{CC}	
$a_1(t_{w_1} = 0.7)$:	Jules Verne, novels by Jules Verne, j. verne, verne
$a_2(t_{w_2} = 0.5)$:	adventure novels, novels
$a_3(t_{w_3} = 0.8)$:	Jules Verne, novels by Jules Verne, j. verne, verne
$a_4(t_{w_4} = 0.8)$:	Jules Verne, novels by Jules Verne, j. verne, verne
$a_5(t_{w_5} = 0.6)$:	Jules Verne, novels by Jules Verne, j. verne, verne
$a_6(t_{w_6} = 0.4)$:	Jules Verne, novels by Jules Verne, j. verne, verne

Fig. 9. An example of label selection.

answers collected from crowd workers are shown in Fig. 9 (each answer is associated with the trustworthiness of the corresponding worker). The 1st-crowd-answer is ca^1 = Jules Verne, novels by Jules Verne, j. verne, verne. Considering a quorum $q = 0.65$, the q-constraint is satisfied by ca^1, in that $(t_{w_1} + t_{w_3} + t_{w_4} + t_{w_5} + t_{w_6} = 3.3) > (0.65 \cdot \sum_{w \in W} t_w = 2.47)$. Moreover, we have $t_w^{max} = t_{w_4} = 0.8$. The bop-constraint is satisfied by 1st-crowd-answer ca^1, in that $(t_{w_2} + t_{w_4} = 1.3) < (0.65 \cdot \sum_{w \in W} t_w = 2.47)$. According to the supermajority mechanism, the crowd-consensus is reached and the agreement-based task result is Jules Verne, novels by Jules Verne, j. verne, verne.

Based on the result of the task T^{CC}, the layout of the representative-choice task T^{RC} for the web resource wr_A is shown in Fig. 8(B). After execution of task T^{RC} and corresponding consensus evaluation upon the collected worker answers, the final label selected for wr_A is FL_A = novels by Jules Verne.

6. Setting up Argo for CLabel enforcement

The Argo system enforces the CLabel execution through the selection of a *labeling modality* and a *configuration pattern*, respectively.

Choice of the labeling modality. The labeling modality allows to specify the number of distinct labels that can be assigned to a considered web resource $wr \in DS$. Two different modalities called simple-label and multiple-label are supported in Argo. The simple-label modality is the default option in Argo and it generates "one-to-one" associations, meaning that CLabel produces exactly one label for a considered resource wr. The multiple-label modality generates "one-to-many" associations, meaning that CLabel can produce more than one label for a considered resource wr. The multiple-label modality extends the CLabel approach on the label selection stage where the crowdsourcing tasks for candidate-choice T^{CC} are re-designed in the task-execution parameters. In particular, given a web resource wr to label, the task-force of T^{CC} is enlarged to collect more crowd preferences about the favorite candidate labels. The consensus manager of Argo is then invoked to rank candidates according to the received crowd preferences, which are used to generate the resulting label-set for the resource wr.

Choice of the configuration pattern. The use of crowdsourcing for enforcing resource labeling introduces expenses that need to be covered for rewarding human workers. Two different configuration patterns, called quality-oriented and efficiency-oriented, are defined in Argo for enabling to dynamically setup the impact of crowdsourcing activities on the overall CLabel process.

Quality-oriented configuration. The main goal is to rely on crowdsourcing as much as possible for enforcing accurate labeling results. This pattern is conceived for those situations in which crowdsourcing provides a crucial contribution for qualitative labeling and the use of automated techniques/tools is not possible or not completely effective. The quality-oriented configuration is characterized by the following features:

- *Candidate label refinement*. In label normalization, the hierarchical clustering algorithm is employed to determine the set of candidate labels *CL* out of the set of emerging labels *EL*. The use of an automated procedure can generate a set *CL* where some emerging label is misplaced, meaning that labels belonging to a candidate $cl \in CL$ could be not coherent. This situation can have a negative impact on the effectiveness of the crowdsourcing activities in the subsequent label selection stage, due to possible misunderstanding of crowd workers in evaluating candidates. For this reason, in the quality-oriented configuration, crowdsourcing is introduced also in the label normalization stage for supporting crowd refinement of the candidate labels *CL*. Decide-question tasks are enforced after the execution of hierarchical clustering to evaluate the composition of each candidate label. Given $cl \in CL$, a task is submitted to crowd workers where the set of answer options *AO* contains the emerging labels belonging to *cl*. Workers are asked to identify the (possible) misplaced label of *cl*. According to the result of consensus evaluation, labels can be shifted from one candidate to another to generate a refined set *CL* to be considered in the label selection stage.
- *Management of uncommitted tasks*. In the quality-oriented configuration, the uncommitment manager of Argo is invoked to schedule a new execution of the uncommitted task with a different task-execution specification. The re-execution of an uncommitted task is characterized by a different task-force, so that a worker cannot be twice-involved in the execution of the same task. The worker trustworthiness parameter wt is increased with the aim to increase the reliability of the task-force and thus the likelihood to obtain an agreement.

Efficiency-oriented configuration. The main goal is to rely on automated tools as much as possible by limiting the costs of crowdsourcing activities at the same time. This pattern is conceived for those situations in which automated techniques/tools are capable to provide effective results and crowdsourcing contributions are mainly related to the elicitation stage. The efficiency-oriented configuration is characterized by the following features:

- *Automatic choice of candidate representative*. Crowdsourcing is employed for the execution of representative-choice tasks with the aim at crowd-selecting the most appropriate representative of the candidate label $cl_s \in CL$. In the efficiency-oriented configuration, representative-choice tasks are not executed and candidate representatives are automatically selected according to a prominence-based criterion. Different notions of prominence are supported in Argo. For example, the degree of similarity among labels can be chosen as a criterion for prominence calculation. Given a candidate label $cl \in CL$, the candidate representative is the label $el \in cl$ that has the highest similarity value on average with the other labels belonging to *cl*. As another example, the worker trustworthiness can be used for prominence calculation. In this case, the candidate representative of $cl \in CL$ is the label $el \in cl$ that has been provided in the elicitation stage by the worker with the highest trustworthiness value.
- *Automatic resolution of uncommitted tasks*. In the efficiency-oriented configuration, the uncommitment manager of Argo is invoked to decide how to automatically determine the result of tasks that are not successfully committed (i.e., supermajority constraint not satisfied). The goal of the uncommitment manager is to reconsider the answers provided by crowd workers in the executed task and to downgrade consensus evaluation, and thus the associated supermajority constraints, until an agreement is found. Two

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.

possible consensus downgrades are supported in Argo. A kind of downgrade regards the involved task-force tf. In this case, the answers provided by workers with lower trustworthiness are excluded from consensus evaluation. Another kind of downgrade regards the quorum-majority qm. In this case, the task result is committed on the worker answer that obtains the higher consensus, independently from the q-constraint verification.

7. Experimental results

System-level evaluation of Argo and related techniques for consensus management has been performed against the SQUARE benchmark [33] and obtained results are reported in [28].

A preliminary CLabel evaluation has been performed on a case-study of web-resource labeling called **fictional-book** that has been also exploited for the running examples throughout the paper. In this case-study, each resource to label is a group of book titles with one or more common property (e.g., author, genre, publisher). A crowd worker visualizes a set of titles and can exploit external datasources (e.g., Wikipedia) for exploring the book details (if necessary). The task request is to provide a label for describing the book titles contained in the task as a whole. In fictional-book, the labeling criterion is implicitly suggested to the crowd workers, since the possible common properties are known in advance. The experiment on the fictional-book case-study has been performed to analyze the crowd behavior throughout the overall CLabel process. By observing the experiment results, we note that the final label FL_{wr} returned by CLabel always corresponds to one of the common properties shared by the books belonging to the considered group/task wr. For most of the groups/tasks (i.e., 60%), the final selected label is the author name, while the literary genre is the CLabel result in the majority of the remaining cases. Further details about the obtained results on the fictional-book case-study are discussed in [34].

In the following, we focus on a new case-study of web-resource labeling called **music-emotion**, which represents a more challenging labeling test, since the considered web resources are not associated with descriptive metadata and the crowd workers can rely only on their own personal music knowledge and/or listening experience for the label choice.

7.1. The music-emotion case-study

In music-emotion, each resource is a song, namely an audio track, and the request for the crowd workers is to provide a label for describing the emotional mood inspired by the song after a listening experience. A summary-view of the music-emotion case-study is shown in Table 1. Music-emotion is based on a dataset of 288 resources (i.e., songs) to label. The experimentation has a duration of 30 days and it is conducted with a crowd of 223 workers selected from a class of master-degree students playing the role of workers (average worker age is 21 years old). As occurs in a conventional crowdsourcing paradigm, workers involved in the case-study are selected on the basis of a voluntary participation, rather than according to expertise-based constraints. As a result, workers are not constrained to perform a minimum number of tasks. Moreover, we stress that a task can be refused by a worker when she/he feels to have insufficient expertise to provide a reliable answer. We observe that each worker executed around 11 tasks on average in the case-study, and a total of 264 labeled songs are returned as a result of CLabel.

In the following, we first present the case-study configuration and we provide details about the obtained results. Then, we focus on discussing the results on two specific tests devoted to the *label variety analysis* (T1) and to the *comparison against Stereomood* (T2), respectively.

Table 1
Summary-view of music-emotion case-study.

Experiment duration:	30 days
# of involved workers:	223
# of music tracks in the dataset:	288
# of labeled music tracks:	264
Average # of tasks per worker:	10.95
About label selection:	
# of executed tasks:	407
# of labels committed at 1st execution:	127
# of labels committed at 2nd execution:	137
# of terminated tasks:	3

Case-study configuration and obtained results. The music-emotion case-study relies on the predefined configuration presented in Section 5 and on the quality-oriented configuration pattern introduced in Section 6. Moreover, multiple-label and single-label modalities are employed for tests T1 and T2, respectively. In music-emotion, the execution of the label elicitation stage produced 2780 emerging labels that became 874 candidate labels after the execution of label normalization. This means that the execution of hierarchical clustering returns 874 clusters with an average of 3.18 labels inside. The cluster composition spans from singleton clusters (i.e., candidates with a unique representative label) to clusters with many labels inside, sometimes up to six labels. Frequently, label aggregation into clusters is due to basic syntax differences, such as for example plural forms and capital letters, that are captured through a preliminary step of text-normalization. A high number of singleton clusters is returned (i.e., around 300 clusters, ∼34% of resulting 874 clusters), thus expressing the attitude of crowd workers to provide original labels and confirming the positive contribution of crowdsourcing when applied to resource labeling issues. However, it is important to note that singleton clusters are selected as final resource label only for 20% of songs in the music-emotion dataset, while most of the resources are finally labeled with candidates corresponding to clusters with more than one label inside. This result is not surprising and it is motivated by the fact that popular labels represent the preferred choice of crowd workers in the label selection stage. This is a positive result of CLabel since it indicates that labels provided as final answer of the labeling process represent a mix of popular, frequently-recognized resource interpretations and original, rarely-recognized resource interpretations. For label selection, an overall number of 407 tasks have been executed including both candidate and representative choice tasks. Since the same predefined configuration is employed in both candidate and representative choice tasks, it is not surprising that overall consensus performances coincide on average for the two kinds of label selection tasks. Namely, given that a task-force $tf = 6$ and a quorum-majority $q = 0.65$ are specified in the predefined configuration, we note that committed tasks (i.e., tasks where consensus have been reached) are characterized by 4.19 workers on average participating to the consensus (i.e., workers in the support group of 1st candidate answer), which corresponds to 70% of worker consensus on average. Since the quality-oriented configuration pattern is enforced, uncommitted tasks are submitted to re-execution with a task force characterized by an increased trustworthiness value. In label selection, we obtained that 127 and 137 resources have been labeled as result of the first and second crowdsourcing executions, respectively. In the configuration, we expected to support up to three executions of a crowdsourcing task. In this respect, we observe that no resources have been labeled in the third crowdsourcing execution and three crowdsourcing tasks have been terminated, meaning that the three executions have been completed without reaching a committed

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.

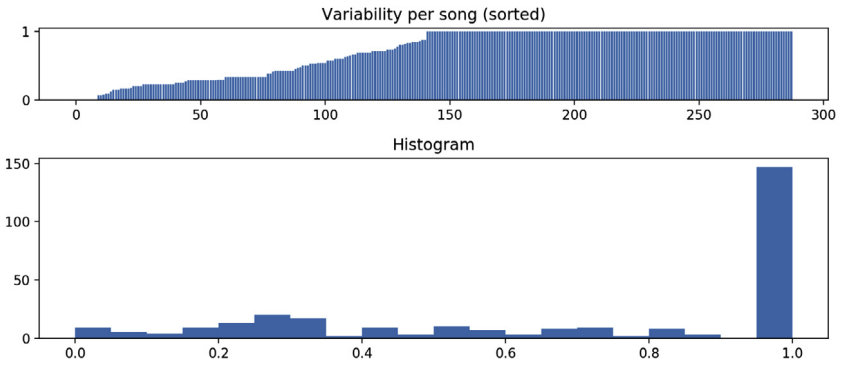

Fig. 10. Label variety of CLabel in the music-emotion case-study.

result (i.e., no consensus reached). For 21 resource in the music-emotion dataset, the tasks of the label selection stage have not been completed due to missing answers from crowd workers before the deadline of experimentation. Thus, these resources are considered as unlabeled in the final result (8% of considered resources in the music-emotion dataset).

Label variety analysis (T1). The use of crowdsourcing to collect many emerging labels reflecting different interpretations/understandings of a considered web resource (i.e., label elicitation stage) is a peculiar aspect of CLabel. However, the availability of many labels is not an advantage by itself, due to the redundancies and/or mistypings that frequently occur when multiple crowdsourcing answers from different workers are collected for a single task. Literature approaches are capable to manage multiple labels either provided by automated procedures or collected from crowdsourcing, but post-processing of obtained results to discard duplicates or poorly-meaningful labels is not enforced or it is not completely effective. For this reason, the use of a label normalization stage represents a further distinguishing aspect of CLabel, since it allows to reconcile duplicate/synonym labels in a single candidate. In this test, we aim at evaluating the variety of labels provided by CLabel. This means that our goal is (i) to analyze the multiple labels provided by CLabel as result for a given resource, and (ii) to evaluate whether each resulting label represent a different resource interpretation. To this end, we recall that we rely on the Argo system with a setup based on the multiple-label modality and we consider the top-three labels returned by CLabel.

Method. For label variety evaluation, we calculate the degree of similarity among the final labels FL_{wr}={FL_1, FL_2, FL_3} associated with each web resource (i.e., song) wr. For similarity calculation, we rely on the WordNet lexical system [35], so that we can capture the meaning similarity among labels, instead of the conventional syntax similarity. Given two labels $FL_i, FL_j \in FL_{wr}$, the similarity degree $\sigma(FL_i, FL_j)$ is calculated as:

$$\sigma(FL_i, FL_j) = \max(\omega(syn_k, syn_m)),$$

where (i) syn_k, syn_m are WordNet synsets associated with FL_i, FL_j, respectively, and (ii) $\omega(syn_k, syn_m)$ is the Wu–Palmer similarity function, which returns a score denoting how similar two word senses are, based on the depth of the two synsets in the WordNet taxonomy according to their Least Common Subsumer (LCS).

The lower is the result of $\sigma(FL_i, FL_j)$, the higher is the corresponding label variety, meaning that FL_i, FL_j have two different meanings and provide two different resource interpretations. In particular, for each resource/song wr, we calculate the variety of the associated CLabel labels FL_{wr} as the maximum distance among the label similarity values, namely:

$$\max(1 - \sigma(FL_i, FL_j))$$

where $FL_i, FL_j \in FL_{wr}$ and $i \neq j$.

Discussion. In Fig. 10, we show the sorted distribution of label variety for the resource/songs of music-emotion (mean value: 0.71; standard deviation: 0.34; median value: 1.00). Given a resource/song wr, results confirm a high level of variety of the CLabel labels FL_{wr}. In particular, for most of the resources, the label variety is 1.00, meaning that the three labels do not share a Least Common Subsumer in WordNet and represent a different meaning (i.e., resource interpretation) that the crowd workers associated with the song. This result is particularly interesting since it is applied to a multimedia dataset where resources are not associated with descriptive metadata and the subjective point of view of crowd workers represents the crucial contribution for resource/song annotation. Moreover, we note that the number of labeled resources (i.e., 264) represents the 0.92% on the overall number of songs in the music-emotion dataset (i.e., 288). By labeled song, we mean a resource that obtained a final label as result of the CLabel approach, namely a resource on which workers reached the consensus commitment in both candidate-choice tasks and representative-choice tasks. We argue that this is a further positive result of CLabel, since it indicates that the consensus evaluation mechanism of CLabel succeeds to enforce an effective collaborative approach to resource labeling where the agreement formation among groups of independent crowd workers is actually enforced.

Comparison against Stereomood (T2). In this test, we consider the song classification provided by the Stereomood system [36] where audio tracks are organized in predefined categories according to a mix of social-based and expert-based recommendations. The goal of the test is to analyze the Stereomood labels in comparison with the labels provided by crowd workers in CLabel, with the aim to observe and discuss both similar and dissimilar behaviors. To this end, we recall that we rely on the Argo system with a setup based on the simple-label modality.

Method. We choose to enforce a dimension-based analysis where "sentiment" and "music" are the considered dimensions. The goal is to observe how the Stereomood and the CLabel labels are placed with respect to these dimensions. In other words, for each song, we aim at analyzing whether and how the Stereomood label and the CLabel label are compliant in expressing a sentiment or a music genre. In particular, we consider the terms ST = sentiment and MT = music. Given a resource/song wr, we call SL_{wr} and FL_{wr} the associated Stereomood and CLabel labels, respectively. For each wr, we calculate the similarity degree of both SL_{wr} and FL_{wr} with the sentiment term ST and the music term MT, namely:

$$\begin{array}{ll} \sigma(ST, SL_{wr}) & \sigma(ST, FL_{wr}) \\ \sigma(MT, SL_{wr}) & \sigma(MT, FL_{wr}) \end{array}$$

where σ returns the similarity degree based on the WordNet lexical system and the Wu–Palmer function according to the Least

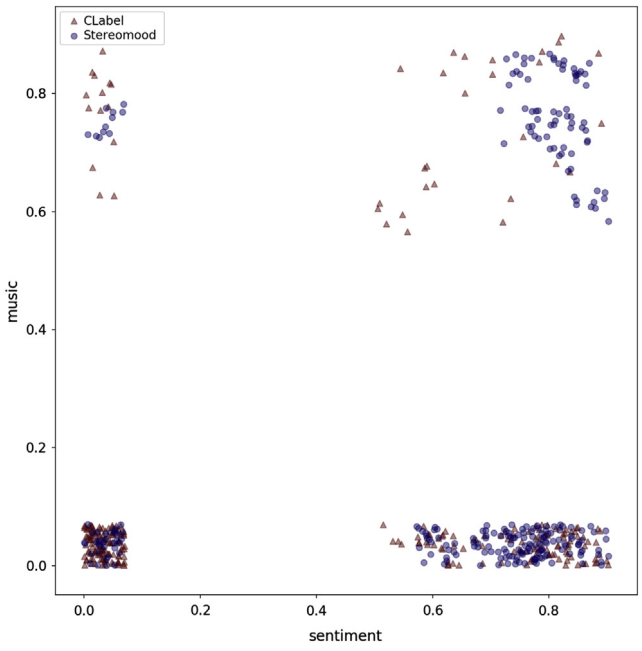

Fig. 11. Comparison of Stereomood and CLabel in the music-emotion case-study.

Common Subsumer of synsets associated with the considered terms/labels.

Discussion. The choice to enforce a dimension-based analysis is due to the fact that crowd workers provided very creative results, then the precise correspondence of CLabel labels with the Stereomood labels can be considered an unfrequent event. In Fig. 11, we show the scatter plot of both Stereomood and CLabel labels with respect to sentiment and music dimensions. As a general remark, we note that there is a high correlation between Stereomood and CLabel labels. Moreover, most of the resources/songs are distributed in the bottom part of the diagram, meaning that the music genre is less relevant than the sentiment as labeling criterion. Items placed in the bottom-left corner of the diagram represent labels that are poorly concerned with both sentiment and music genre or that are not recognized in WordNet. In this case, Stereomood and CLabel agree in labeling resources by leveraging on a term that is other than a sentiment or a music genre. Examples of labels for songs in this area are epic/mistery and lost in thoughts/carefree (Stereomood/CLabel). Items placed in the bottom-right corner of the diagram represent labels that express a sentiment without being concerned with a music genre. Examples in this area are happy/glee and calm/melancholy (Stereomood/CLabel). Items placed in the top-left corner of the diagram represent labels that express a music genre without being concerned with a sentiment. Examples in this area are swing/swinging (Stereomood/CLabel). Finally, items placed in the top-right corner of the diagram represent labels that both express a sentiment and a music genre. Examples in this area are spiritual/melancholy and sweet/love (Stereomood/CLabel).

Final considerations. According to the experimental results, we argue that CLabel succeeds in providing resource labels representing a mixed result based on (i) human creativity emerging from original, crowd-generated labels, and (ii) normalization techniques for extracting popular labels corresponding to widely-recognized resource interpretations. The use of CLabel and related crowdsourcing tasks and consensus-based mechanisms is particularly appropriate for the music-emotion case-study as well as for those scenarios where resource metadata are not available and feature extraction tools are not completely effective. As a general remark, CLabel results to be particularly appropriate for application to labeling case-studies where human feelings and preferences are decisive to select the answers (i.e., labels) supported by the majority of the crowd. Moreover, we note that CLabel succeeds in providing label variety when multiple labels are required for annotating resources, thus avoiding duplicate, repetitive labels as occurs in many literature approaches. Application of CLabel to labeling scenarios where a set of resource labels are tool-generated through automatic procedures is still possible and useful for determining the final result. In this case, the label elicitation stage can be skipped and stages of label normalization and selection can be directly executed on the available labels.

8. Concluding remarks

In this paper, we presented the CLabel approach for collaborative web-resource labeling based on a combination of crowdsourcing and aggregation/normalization techniques to be invoked in a seamless and disciplined way. A peculiar feature of CLabel is the use of create-question tasks and consensus mechanisms in a configurable task-design framework, so that crowd workers are actually involved in both elicitation/proposal of possible label candidates, and subsequent choice of the final labels selected for assignment to the given resources.

Ongoing research work is focused on extending the proposed CLabel approach and related techniques to other kind of articulated crowdsourcing-based problem where the use of both create-question and decide-question tasks can be effectively employed, like for example resource classification and collaborative storytelling.

References

[1] D.R. Karger, S. Oh, D. Shah, Efficient crowdsourcing for multi-class labeling, in: Proc. of the Int. Conference on Measurement and Modeling of Computer Systems, (SIGMETRICS 2013), ACM, Pittsburgh, PA, USA, 2013, pp. 81–92.
[2] Q.V.H. Nguyen, C.T. Duong, T.T. Nguyen, M. Weidlich, K. Aberer, H. Yin, X. Zhou, Argument discovery via crowdsourcing, VLDB J. 26 (4) (2017) 511–535.
[3] G. Demartini, D.E. Difallah, P. Cudré-Mauroux, ZenCrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking, in: Proc. of the 21st WWW Conference, ACM, Lyon, France, 2012, pp. 469–478.
[4] E.E. Arolas, F. G.-L. de Guevara, Towards an integrated crowdsourcing definition, J. Inf. Sci. 38 (2) (2012) 189–200.
[5] F.K. Khattak, A. Salleb-Aouissi, Robust crowd labeling using little expertise, in: Proc. of the 16th Int. Conference on Discovery Science, (DS 2013), Springer, Singapore, 2013, pp. 94–109.
[6] W. Tang, M. Lease, Semi-supervised consensus labeling for crowdsourcing, in: Proc. of the 2nd SIGIR Int. Workshop on Crowdsourcing for Information Retrieval, CIR 2011, 2011.
[7] D.W. Barowy, C. Curtsinger, E.D. Berger, A. McGregor, AutoMan: A platform for integrating human-based and digital computation, in: Proc. of the 27th Annual ACM SIGPLAN OOPSLA Conference, Tucson, AZ, USA, 2012.
[8] A. Bozzon, M. Brambilla, S. Ceri, A. Mauri, Reactive crowdsourcing, in: Proc. of the 22nd Int. WWW Conference, Rio de Janeiro, Brazil, 2013.
[9] L.B. Chilton, G. Little, D. Edge, D.S. Weld, J.A. Landay, Cascade: Crowdsourcing taxonomy creation, in: Proc. of the Int. Conference on Human Factors in Computing Systems, (CHI 2013), ACM, Paris, France, 2013, pp. 1999–2008.
[10] M.S. Bernstein, G. Little, R.C. Miller, B. Hartmann, M.S. Ackerman, D.R. Karger, D. Crowell, K. Panovich, Soylent: a word processor with a crowd inside, in: Proc. of the 23rd Annual ACM Symposium on User Interface Software and Technology, UIST 2010, New York, NY, USA, 2010, pp. 313–322.
[11] A. Kulkarni, M. Can, B. Hartmann, Collaboratively crowdsourcing workflows with turkomatic, in: Proc. of the ACM Conference on Computer Supported Cooperative Work, Seattle, Washington, USA, 2012, pp. 1003–1012.
[12] D. Jurgens, Embracing ambiguity: A comparison of annotation methodologies for crowdsourcing word sense labels, in: Proc. of the Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Atlanta, Georgia, USA, 2013, pp. 556–562.
[13] V.S. Sheng, F.J. Provost, P.G. Ipeirotis, Get another label? Improving data quality and data mining using multiple, noisy labelers, in: Proc. of the 14th Int. Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, 2008, pp. 614–622.

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.

[14] S. Oyama, Y. Baba, Y. Sakurai, H. Kashima, Accurate integration of crowdsourced labels using workers' self-reported confidence scores, in: Proc. of the 23rd Int. Joint Conference on Artificial Intelligence, IJCAI 2013, Beijing, China, 2013.
[15] J. Zhang, V.S. Sheng, T. Li, Label aggregation for crowdsourcing with bi-layer clustering, in: Proc. of the 40th Int. ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Shinjuku, Tokyo, Japan, 2017, pp. 921–924.
[16] L. Yin, J. Han, W. Zhang, Y. Yu, Aggregating crowd wisdoms with label-aware autoencoders, in: Proc. of the 26th Int. Joint Conference on Artificial Intelligence, IJCAI-17, Melbourne, Australia, 2017, pp. 1325–1331.
[17] L. Pion-Tonachini, S. Makeig, K. Kreutz-Delgado, Crowd labeling latent dirichlet allocation, Knowl. Inf. Syst. (2017).
[18] P.G. Ipeirotis, F. Provost, J. Wang, Quality management on amazon mechanical turk, in: Proc. of the 16th ACM SIGKDD Workshop on Human Computation, Washington, DC, USA, 2010, pp. 64–67.
[19] M. Joglekar, H. Garcia-Molina, A.G. Parameswaran, Evaluating the crowd with confidence, in: Proc. of the 19th ACM International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, 2013, pp. 686–694.
[20] J.S. Downs, M.B. Holbrook, S. Sheng, L.F. Cranor, Are your participants gaming the system? Screening mechanical turk workers, in: Proc. of the 28th Int. Conference on Human Factors in Computing Systems, (CHI 2010), ACM, Atlanta, Georgia, USA, 2010, pp. 2399–2402.
[21] J. Le, A. Edmonds, V. Hester, L. Biewald, Ensuring quality in crowdsourced search relevance evaluation: The effects of training question distribution, in: Proc. of the 1st SIGIR Workshop on Crowdsourcing for Search Evaluation, Geneva, Switzerland, 2010, pp. 21–26.
[22] D. Oleson, A. Sorokin, G.P. Laughlin, V. Hester, J. Le, L. Biewald, Programmatic gold: Targeted and scalable quality assurance in crowdsourcing, in: Proc. of the 1st AAAI Workshop on Human Computation, San Francisco, California, USA, 2011.
[23] S. Dow, A.P. Kulkarni, B. Bunge, T. Nguyen, S.R. Klemmer, B. Hartmann, Shepherding the crowd: Managing and providing feedback to crowd workers, in: Proc. of the Int. Conference on Human Factors in Computing Systems, (CHI 2011), ACM, Vancouver, BC, Canada, 2011, pp. 1669–1674.
[24] D.L. Hansen, P.J. Schone, D. Corey, M. Reid, J. Gehring, Quality control mechanisms for crowdsourcing: peer review, arbitration, & expertise at familysearch indexing, in: Proc. of the 16th ACM Conference on Computer Supported Cooperative Work, CSCW 2013, San Antonio, TX, USA, 2013, pp. 649–660.
[25] S. Amer-Yahia, S.B. Roy, Toward worker-centric crowdsourcing, IEEE Data Eng. Bull. 39 (4) (2016) 3–13.
[26] G. Kazai, J. Kamps, N. Milic-Frayling, Worker types and personality traits in crowdsourcing relevance labels, in: Proc. of the 20th ACM Conference on Information and Knowledge Management, CIKM, Glasgow, Scotland, UK, 2011, pp. 1941–1944.
[27] S. Castano, A. Ferrara, S. Montanelli, A multi-dimensional approach to crowd-consensus modeling and evaluation, in: Proc. of the 34th Int. Conference on Conceptual Modeling, ER 2015, Stockholm, Sweden, 2015.
[28] S. Castano, A. Ferrara, L. Genta, S. Montanelli, Combining crowd consensus and user trustworthiness for managing collective tasks, Future Gener. Comput. Syst. 54 (2016).
[29] The Argo Crowdsourcing Project, http://island.ricerca.di.unimi.it/projects/argo/, Italian language.
[30] G. Navarro, A guided tour to approximate string matching, ACM Comput. Surv. 33 (1) (2001).
[31] C.D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.
[32] S. Castano, A. Ferrara, S. Montanelli, Structured data clouding across multiple webs, Inf. Syst. 37 (4) (2012).
[33] A. Sheshadri, M. Lease, SQUARE: A benchmark for research on computing crowd consensus, in: Proc. of the 1st AAAI Conference on Human Computation and Crowdsourcing, HCOMP 2013, Palm Springs, CA, USA, 2013.
[34] S. Castano, A. Ferrara, S. Montanelli, Designing crowdsourcing tasks with consensus constraints, in: Proc. of the Int. IEEE Conference on Collaboration Technologies and Systems, CTS 2016, Orlando, FL, USA, 2016, pp. 97–103.
[35] The WordNet Lexical System, https://wordnet.princeton.edu/.
[36] Stereomood, https://soundcloud.com/stereomood.

Silvana Castano is full professor of Computer Science at the University of Milano, where she chairs the Information systems & knowledge management (ISLab) group. She received the Ph.D. degree in Computer and Automation Engineering from Politecnico di Milano. Her main research interests are in the area of databases and information systems with focus on methods and tools for per heterogeneous data analysis, matching, classification, integration, summarization, consensus-based crowdsourcing, schema, ontology, instance matching, data security. She has been serving as PC member for several important database, information systems, and Semantic Web conferences. In 2009, she is PC co-chair of the ER 2009 conference. From 2008 to 2014, she were President-Elect of GRIN, the Italian association of University Professors in Informatics.

Alfio Ferrara is associate professor of Computer Science at the University of Milano, where he received his Ph.D. in Computer Science in 2005. His research interests include database and semi-structured data integration, Web-based information systems, data analysis, and knowledge representation and evolution.

Stefano Montanelli is Assistant Professor in Informatics at the University of Milano (UNIMI), where he received his Ph.D. in Informatics in 2007. His main research interests include Semantic Web, ontology matching and evolution, and web data classification. Recent research activities are focused on visual exploration of web data, smart-city data management, and crowd phenomena. A three-stage approach called CLabel is proposed for enforcing collaborative web-resource labeling. CLabel is characterized by the disciplined combination of automatic tools/techniques and crowdsourcing. CLabel is capable of dealing with complex crowdsourcing processes where multiple task typologies, including create-questions, are required. CLabel leverages on crowd preferences and consensus for capturing the different interpretations that can be associated with a considered web resources.

Please cite this article in press as: S. Castano, et al., Leveraging crowd skills and consensus for collaborative web-resource labeling, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.024.