DATA PAPER

Refining Linked Data with Games with a Purpose*

Irene Celino[†]**, Gloria Re Calegari & Andrea Fiano**

Cefriel, Milano 20126, Italy

Keywords: Human computation; Games with a purpose; Linked data; Knowledge graph; Data refinement; Data linking; Truth inference

Citation: I. Celino, G. Re Calegari & A. Fiano. Refining linked data with games with a purpose. Data Intelligence 2(2020), 417-442. doi: 10.1162/dint_a_00056

Received: April 15, 2019; Revised: August 10, 2019; Accepted: October 10, 2019

ABSTRACT

With the rise of linked data and knowledge graphs, the need becomes compelling to find suitable solutions to increase the coverage and correctness of data sets, to add missing knowledge and to identify and remove errors. Several approaches – mostly relying on machine learning and natural language processing techniques – have been proposed to address this *refinement* goal; they usually need a *partial gold standard*, i.e., some "ground truth" to train automatic models. Gold standards are manually constructed, either by involving domain experts or by adopting crowdsourcing and human computation solutions. In this paper, we present an open source software framework to build *Games with a Purpose for linked data refinement*, i.e., Web applications to crowdsource partial ground truth, by motivating user participation through fun incentive. We detail the impact of this new resource by explaining the specific *data linking* "purposes" supported by the framework (creation, ranking and validation of links) and by defining the respective *crowdsourcing tasks* to achieve those goals. We also introduce our approach for *incremental truth inference* over the contributions provided by players of Games with a Purpose (also abbreviated as GWAP): we motivate the need for such a method with the specificity of GWAP *vs*. traditional crowdsourcing; we explain and formalize the proposed process, explain its positive consequences and illustrate the results of an experimental comparison with state-of-the-art approaches. To show this resource's versatility, we describe a set of *diverse applications* that we built on top of it; to demonstrate its reusability and extensibility potential, we provide references to detailed documentation, including an entire *tutorial* which in a few hours guides new adopters to customize and adapt the framework to a new use case.

[†] Corresponding author: Irene Celino (Email: irene.celino@cefriel.com; ORCID: 0000-0001-9962-7193).

[*] Extended version of the paper "A Framework to Build Games with a Purpose for Linked Data Refinement", accepted at the 17th International Semantic Web Conference.

1. INTRODUCTION

In the era of data-driven technologies, evaluating and increasing the quality of data is an important topic. In the Semantic Web community, the emergence of the so-called knowledge graphs has been welcomed as a success of the linked data movement but, in the meantime, has raised a number of research challenges about their management, from creation to verification and correction. The term knowledge graph refinement [1] has been used to indicate the process to increase the quality of knowledge graphs in terms of finding and removing errors or adding missing knowledge. The same refinement operation is a challenge also for any linked data set of considerable size.

Addressing at scale the linked data refinement problem requires a trade-off between purely-manual and purely-automatic methods, with approaches that, on the one hand, adopt human computation [2] and crowdsourcing [3] to collect manually labeled training data and, on the other hand, employ statistical or machine learning to build models and apply the refinement operation on a larger scale. Indeed, active learning [4] and other recent research approaches in the artificial intelligence area have put back the "human-in-the-loop" by creating mixed human-machine approaches.

In this paper, we present an open source and reusable resource to build human computing applications in the form of games with a purpose [5] aimed to solve linked data refinement tasks. The framework can be adopted both for refining a pre-existing knowledge graph (when the human intervention is enough to complete the refinement task) and for building a gold standard (to be used as training set when machine learning approaches are needed to process very large linked data sets). By *refinement* in this paper, we mainly mean *data linking*, as more precisely defined in Section 3. The presented resource consists of both a software framework and a crowdsourcing approach, that can be customized and extended to address different data linking issues.

The paper is organized as follows: Section 2 presents related work; Section 3 defines the refinement purpose addressed by our framework and Section 4 explains the crowdsourcing task; details about our incremental truth inference algorithm are given in Section 5; the software resource is presented in Section 6 and some applications built on it are illustrated in Section 7; an evaluation of the effectiveness of the truth inference approach is illustrated in Section 8; since it is a recently released resource, to explain its potential customization and to simplify its adoption, we set up an entire tutorial, briefly introduced in Section 9; the code of the framework and the tutorial are available on GitHub[①] and documented on Apiary[②] (links throughout the paper); Section 10 concludes the paper.

2. RELATED WORK

Data linking is rooted in the record linkage problem studied in the databases community since the 1960s [6]; for this reason, in the Semantic Web community, the term is often used to name the problem of

[①] https://github.com/STARS4ALL/gwap-enabler/wiki.

[②] https://gwapenablerapi.docs.apiary.io.

finding equivalent resources on the Web of linked data [7]; in this meaning, data linking is the process to create links that connect subject- and object-resources from two different data sets through a property that indicate a correspondence or an equivalence (e.g., owl: sameAs).

We prefer to generalize the concept of *data linking* extending it to the task of creating links in the form of RDF triples, without limitation to specific types of resources or predicates, nor necessarily referring to linking across two different data sets or knowledge graphs (data linking can happen also within a single data set or knowledge graph). In this sense, data linking can be interpreted as a solution to a *linked data refinement* issue, i.e., the process to create, update or correct links in a data set, in which a link is any relation made of a subject-predicate-object triple. As defined in [1] with respect to knowledge graphs, with data linking we do not consider the case of constructing a data set or graph from scratch, but rather we assume an existing input data set which needs to be improved by adding missing knowledge or identifying and correcting mistakes.

The Semantic Web community has long investigated the methods to address the data linking problem, by identifying linked data set quality assessment methodologies [8] and by proposing manual, semi-automatic or automatic tools to implement refinement operations [9, 10]. The large majority of refinement approaches, especially on knowledge graphs in which scalable solutions are needed, are based on different statistical and machine learning techniques [11, 12, 13, 14].

Machine learning methods, however, need a partial *gold standard* to train automated models; those training sets are usually created manually by experts: while this usually leads to higher quality trained models, it is also an expensive process, so those "ground truth" data sets are usually small. Involving humans at scale in an effective way is, on the other hand, the goal of crowdsourcing [3] and human computation [2]. Indeed, microtask workers have been employed as a means to perform manual quality assessment of linked data [15, 16].

Among the different human computation approaches, Games with a Purpose (GWAP) [5] have experienced a wide success, because of their ability to engage users through the incentive of fun. A GWAP is a gaming application that exploits players' actions in the game to solve some (hidden) tasks; users play the game for fun, but the "collateral effect" of their playing is that the comparison and aggregation of players' contributions are used to solve some problems, usually labelling, classification, ranking, clustering, among others. Also in the Semantic Web community, GWAPs have been adopted to solve a number of linked data management issues [17], from multimedia labelling to ontology alignment, from error detection to link ranking: data cleansing in DBpedia [18], collecting linguistic annotations [19], ontology alignment [20], rating triples based on their popularity [21], linking multimedia data sets about smart cities [22], crowdsourcing location-based knowledge [23], annotating art-related media to improve search engines [24], evaluating people ability in recognizing and building triples [25], ranking artworks by their recognizability while creating awareness [26], building a training set for image classification [27].

Aggregating users' contribution is a key issue in Human Computation systems like GWAPs. Originally, aggregation was based on simple agreement: in the ESP game [28], the very first GWAP ever released, players typed in textual labels to tag images and two agreeing users were enough to consider the label "true". Afterwards, "ground truth" tasks, i.e., problems with known solution, were introduced to check the quality of contributions to cope with random answers or malicious players [2, 29].

In crowdsourcing settings, truth inference is always computed ex-post, i.e., contributions aggregation is performed only after the collection of all data from all users [30]; in other words, the number of repetitions (i.e., number of different users required to solve the same task) is set at design time. Related investigation exists on the evaluation of repeated labelling strategies [31] to understand when it is more convenient to stop collecting users contributions. Promising results to optimize the number and quality of collected contributions are coming from active learning research [32, 33], which integrates machine learning with crowdsourcing approaches, also in the case of large-scale data sets. In this paper, we introduce our approach for incremental truth inference that addresses the minimization of repeated labelling, by dynamically detecting when there are "enough" repetitions.

While the general guidelines and rules to build a GWAP have been described and formalized [34] (game mechanics like double player, answer aggregation like output agreement, task design, among others), building a GWAP for linked data management still requires time and effort. To the best of our knowledge, source code was made available only for the labelling game Artigo [24].

3. DATA LINKING PURPOSE

As explained in Section 2, with *data linking* we refer to the general problem of creating links in the form of triples. In this section, we provide the basic definitions and illustrate the cases that our framework supports as purpose of the games that can be built on it.

3.1 Basic Definitions

The following formal definitions will be used throughout the paper.

Resources *$\mathcal{R}$ is the set of all resources (and literals), whenever possible also described by the respective types. More specifically: $\mathcal{R} = \mathcal{R}_s \cup \mathcal{R}_o$, where $\mathcal{R}_s$ is the set of resources that can take the role of subject in a triple and $\mathcal{R}_o$ is the set of resources that can take the role of object in a triple; the two sets are not necessarily disjoint, i.e., it can happen that $\mathcal{R}_s \cap \mathcal{R}_o \neq \varnothing$.*

Predicates *$\mathcal{P}$ is the set of all predicates, whenever possible also described by the respective domain and range.*

Links *$\mathcal{L}$ is the set of all links; since links are triples created between resources and predicates it is: $\mathcal{L} \subset \mathcal{R}_s \times \mathcal{P} \times \mathcal{R}_o$; each link is defined as $l = (r_s, p, r_o) \in \mathcal{L}$ with $r_s \in \mathcal{R}_s$, $p \in \mathcal{P}$, $r_o \in \mathcal{R}_o$. $\mathcal{L}$ is usually smaller than the full Cartesian product of $\mathcal{R}_s$, $\mathcal{P}$, $\mathcal{R}_o$, because in each link (r_s, p, r_o) it must be true that $r_s \in domain(p)$ and $r_o \in range(p)$.*

Link scores *σ is the score of a link, i.e., a value indicating the confidence on the truth value of the link; usually $\sigma \in [0,1]$; each link $l \in \mathcal{L}$ can have an associated score.*

One final note on subject resources: in the following sections, as well as in the framework implementation, we always assume that any subject entity can be shown to players in the game through some *visual representation*; if the entity is a multimedia element (image, video, audio resource) this requirement is automatically satisfied; in other cases, some additional information about the subject may be required: e.g., a place could be represented on a map, a person through his/her photo, a document with its textual content.

3.2 Data Linking Cases

Given the previous definitions, we can split the general data linking problem in a set of more specific cases as follows.

Link creation *A link l is created: given $\mathcal{R} = \mathcal{R}_s \cup \mathcal{R}_o$ and $\mathcal{P}$, the link $l = (r_s, p, r_o)$, $r_s \in \mathcal{R}_s$, $p \in \mathcal{P}$, $r_o \in \mathcal{R}_o$ is created and added to $\mathcal{L}$.*

All three components of the link to be created exist, i.e., they are already included in the sets $\mathcal{R}$ and $\mathcal{P}$. It is important to note that ***classification*** can be seen as a special case of link creation in which, given a resource $r_s \in \mathcal{R}_s$ to be classified and the predicate $p \in \mathcal{P}$ indicating the relation between the resource and a set of possible categories $\{cat_1, cat_2, \ldots, cat_n\} \subset \mathcal{R}_o$, the resource $r_o \in \{cat_1, cat_2, \ldots, cat_n\}$ is selected to create the link $l = (r_s, p, r_o)$.

For example, this is the case of music classification: given a list of resources of type "music tracks" in $\mathcal{R}_s$, the predicate mo:genre $\in \mathcal{P}$ and a set of musical styles in $\mathcal{R}_o$, the task is to assign the music style to each track by creating the link (*track*, *genre*, *style*).

The framework presented in this paper supports any link creation case (including classification) when resources and predicates are already part of $\mathcal{R}$ and $\mathcal{P}$. The case of link creation in which new resources and/or predicates are added to $\mathcal{R}$ and/or $\mathcal{P}$ (e.g., free-text labelling of images) is currently not supported by our framework, but it could be one of its possible extensions.

Link ranking *Given the set of links $\mathcal{L}$, a score $\sigma \in [0,1]$ is assigned to each link l. The score represents the probability of the link to be recognized as true. Links can be ordered on the basis of their score σ, thus obtaining a ranking.*

In other words, we consider a Bernoulli trial in which the experiment consists in evaluating the "recognizability" of a link and the outcome of the experiment is "success" when the link is recognized and "failure" when the link is not recognized. Under the hypothesis that the probability of success is the same every time the experiment is conducted, the score σ of a link l is the estimation for the binomial proportion in the Bernoulli trial.

In the case of human computation, crowdsourcing or citizen science, each trial consists of a human user that evaluates the link and states that, in his/her opinion, the link is true (success) or false (failure); the human evaluators, if suitably selected, can be considered a random sample of the population of all humans; therefore, aggregating the results of the evaluations in the sample, we can estimate the truth value of a link for the entire population, by computing the probability of each link to be recognized as true. Then, ordering links on the basis of their score means having a metrics to compare different links on their respective "recognizability".

For example, this could be the case of ranking photos depicting a specific person (e.g., an actor, a singer, a politician): given a set of images of the person, human-based evaluation could be employed to identify the pictures in which the person is more recognizable or more clearly depicted.

Link validation *Given the set of links $\mathcal{L}$, a score $\sigma \in [0,1]$ is assigned to each link l. The score represents the actual truth value of the link. A threshold $\bar{s} \in [0,1]$ is set so that all links with score $\sigma \geq \bar{s}$ are considered true.*

The difference between link validation and the previous case of link ranking is twofold: first, in link validation each link is considered separately, while in link ranking the objective is to compare links; secondly, while in link ranking human judgment is used to estimate the subjective opinion of the human population, in the case of link validation the hypothesis is that, if a link is recognized as true by the population of humans (or by a sample of that population), this is a good estimation of the actual truth value of the link. The latter is also the reason for the introduction of the threshold $\bar{s}$: while the truth value is binary (0=false, 1=true), human validation is more fuzzy, with "blurry" boundaries; the optimal value for the threshold is very domain- and application-dependent and it is usually empirically estimated.

An example of link validation would be assessing the correct music style in audio tracks: it is well-known that sometimes music genres overlap and identifying a music style could also be subjective (e.g., there is no strict definition of what "rock" is); employing humans in this validation would mean attributing the most shared evaluation of a music track's genre.

As mentioned before, in the last two cases, the human evaluation of a link can be considered a Bernoulli trial: each link l is assessed n times by n different users u; the link is recognized as true X times (with of course $X \leq n$); each user u_i can be more or less reliable and, in some cases, it is possible to estimate his/her reliability ρ_i. Therefore, the score of a link is $\sigma = f(n, X, \rho)$, i.e., it is a function of the number of trials n, the number of successes X and the reliability values of the involved users $\rho = \{\rho_1, \rho_2, \ldots, \rho_n\}$.

4. CROWDSOURCING TASKS FOR DATA LINKING

Our framework allows to design and develop GWAP applications to solve data linking issue. In other words, the games built on top of our framework ask players to solve atomic data linking issues as basic tasks within the gameplay.

It is worth noting that building a GWAP does not automatically guarantee to collect enough players/played games to solve the data linking problem at hand; however, in our experience, if the task to be solved is properly embedded in a simple game mechanics and if the game is targeted to a specific community of interest, a GWAP is a valuable means to collect a "ground truth" data set to train machine learning algorithms [27].

4.1 Game Basics

Each GWAP built with our framework is a simple casual game organized in rounds; each round is formed by several levels and each level requires the player to perform a single action, which corresponds to the creation, ranking or validation of a link. According to the definition of von Ahn [17], each GWAP is an *output-agreement double-player game*: users play in random pairs and the game score is based on the agreement between the answers provided by the players (i.e., if they agree, they get points). Our framework does not require users to play simultaneously, because it implements a common strategy in this kind of games, in which a user plays with a "recorded player", so the game scoring is obtained by matching answers provided at different times.

Our framework allows for both time-limited game rounds, in which players can answer to a variable number of tasks per round depending on their speed, and for level-limited rounds, in which players have a maximum number of tasks to address in each round; the choice of either option depends on considerations related to the specific task difficulty and to the game incentive mechanism.

The game adopts a *repeated linking* approach by asking different players to address the same data linking task; conversely, the same task is never given twice to the same player. The "true" solution of a data linking task, therefore, comes from the aggregation of the answers provided by all the users who "played" the task in any game round. The number of players required to solve a data linking task depends on the aggregation algorithm as explained in Section 4.3.

It is worth noting that the game scoring (i.e., points gained by players) is not directly related to the data linking scoring (i.e., the attribution of a score σ to a link l): the former is an engagement mechanism to retain players, the latter is the very purpose of the game.

4.2 Crowdsourcing Definitions

Hereafter we formulate the crowdsourcing problem in a GWAP, by giving some definitions. For simplicity of explanation, we specifically consider the case of multinomial classification tasks (with a pre-defined set of labels), but the approach can be easily extended to open labelling or other data linking cases with no loss of generality.

We consider a Game with a Purpose aimed to solve a set of tasks $T = \{t_n \mid n = 1, 2, \ldots, N\}$, say, classifying a set of painting images. Each task is a labelling task, in which a label is assigned from a set of admissible values $V = \{v_l \mid l = 1, 2, \ldots, L\}$, say, the painting technique used by the artist (e.g., watercolour, gouache, oil paint).

The GWAP is played by a set of users $U = \{u_k \mid k = 1, 2, \ldots, K\}$. In each game round, a player is assigned a subset $T' \subset T$ of tasks to be solved, in our example, a set of painting pictures to be classified. Given a set of "ground truth" tasks $G = \{g_m \mid m = 1, 2, \ldots, M\}$ for which the solution is known (i.e., a set of paintings for which the actual painting technique is known), in each game round the player is also given a set $G' \subset G$ of control tasks. Control tasks are exactly in the same form of the tasks to be solved, so that the user cannot distinguish them. The answers to control tasks are used to estimate the reliability q_k of the player, which is useful to "weight" contributions on unsolved tasks during truth inference. If, in a game round, player 9 correctly answers 3 out of 4 control tasks, his reliability for that game round will be $q_9 = 0.75$.

Player contributions are collected in a matrix $C = \{c_{n,k} \mid n = 1, 2, \ldots, N \wedge k = 1, 2, \ldots, K\}$, initialized with null or zero values and filled with labels from V whenever a player completes a task (i.e., if player 3 says that painting 5 is a watercolour piece, $c_{3,5}$ will be "watercolour"). The goal of the GWAP is not to completely fill up C, on the contrary C should remain a sparse matrix, with the minimum possible number of players contributions (i.e., non-zero values) required to infer the "true" labels for the tasks.

Finally, *truth inference* is a function applied on players' answers C and reliability values q_k to infer the result set $\hat{Y} = \{\hat{y}_n \mid n = 1,2,\ldots,N\}$, $\hat{y}_n \in V$ for each of the tasks in T. $\hat{Y}$ is computed by aggregation of users' contributions and is an estimate of the "true" unknown labelling Y of the tasks (in our example, for each image, we compute the "true" painting technique). Truth inference is *incremental* if, at each new contribution from a GWAP player, a new estimation of $\hat{Y}$ is computed.

To understand if a task t_i can be considered completed, truth inference computes a set of scores representing the confidence values on the association between t_i (a painting) and each possible labelling value v_i (a painting technique). In other words, the aggregation algorithm builds and updates a matrix of *estimation scores* $\Sigma = \{\sigma_{n,l} \mid n = 1, 2, \ldots, N \wedge l = 1, 2, \ldots, L\}$, $\sigma_{n,l} \in [0,1]$. As in record linkage literature [11], those scores start from 0, and are incrementally increased according to user contributions. Each task t_i is solved when the maximum of its scores $\sigma_{i,*}$ (i.e., the ith row of matrix Σ) overcomes some threshold $\bar{s}$; the "completion" condition can be therefore formulated as follows:

$$\forall t_i \in T \, \exists j \in \{1,2,\ldots,L\} | \sigma_{i,j} = max(\sigma_{i,*}) \wedge \sigma_{i,j} > \overline{s} \tag{1}$$

For example, if the threshold is $\bar{s} = 0.80$ and the scores for painting 7 are $\sigma_{7,watercolour} = 0.25$, $\sigma_{7,gouache} = 0.31$ and $\sigma_{7,oil} = 0.82$, then the algorithm assigns the "oil" painting technique to painting 7 and the task is completed. Truth inference algorithms differ for their specific approach to update the matrix Σ of scores when aggregating user contributions.

4.3 Atomic Tasks and Truth Inference

As mentioned in Section 4.1, the atomic task in any GWAP built with our framework is an individual data linking task. For example, in the case of music classification, a player could be given an audio track (the resource r_s), the relation mo:genre (the predicate p) and some options for music genres (e.g., classical, pop, rock, electronic, representing the potential objects r_o of the link l); the action for the player would be to choose the genre (the resource r_o, say ‘rock’) that better describes the audio track. By performing this action (the atomic task), the player is saying that he believes that the link $l = (r_s, p, r_o)$ is “true”; the game therefore alters the truth score σ of the link l by incrementing it. The score is modified at every player action.

In literature, truth inference algorithms [30] have been heavily explored in crowdsourcing to aggregate and make sense of workers’ contributions. Most state-of-the-art algorithms (e.g., majority voting, expectation maximization [35], message passing [36]) are computed ex-post, i.e., all contributions are first collected and then aggregated, usually by means of iterative algorithms to infer the truth and estimate worker quality until convergence; this requires setting a-priori the number of repetitions of user labelling on each task, possibly collecting redundant information.

In most crowdsourcing platforms (like Amazon Mechanical Turk[3] or Figure Eight[4]), tasks are assigned in batches or Human Intelligence Tasks (HITs) and workers are required to submit their answers within a specific time-frame in order to be eligible for payment [37]. In contrast, in GWAPs contributions are collected as soon as a user decides to play the game: the flow of incoming answers is therefore subject to the “appreciation” of the game by players and a long-tail effect is very often recorded, with a few players playing a lot of rounds and the majority of participants being active for a few minutes only. Therefore, it is of utmost importance to exploit every single player’s contribution and to infer truth in an incremental way, assigning the same task to the minimum sufficient number of different players.

To address the above requirement, in our GWAP Enabler framework the truth inference approach is as illustrated in Figure 1. Each time a player starts a game round, we assign a set of tasks to be solved, some of which are control tasks. We collect the answers from the player and we compute his/her reliability. Then, for each unsolved task, we perform a step of truth inference, and we incrementally compute a new estimation of the task solution. If the new estimation is “good enough” (*cf*. exit condition of Equation (1)), the task is considered solved and removed from the game and its result returned. Otherwise, the task is kept in the game and assigned to the next user/player.

[3] https://www.mturk.com.

[4] https://www.figure-eight.com.

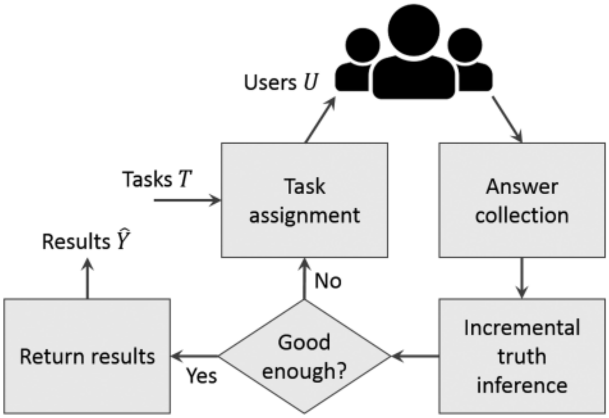


Figure 1. Incremental Truth Inference approach.

In the next section, we explain the incremental truth inference algorithm and its details. The benefits of such an approach with an experimental evaluation are reported in Section 8.

5. INCREMENTAL TRUTH INFERENCE ALGORITHM

This section is devoted to explain the details of our incremental truth inference approach. We first detail the main requirements, then we illustrate the algorithm in pseudo-code and finally we discuss the satisfaction of the prerequisites.

5.1 Requirements

As mentioned in the previous section, in the case of Games with a Purpose, some specific requirements emerge that motivate the need for a new truth inference approach:

[R1] *Dynamic estimate of labelling quality*, by computing player reliability on control tasks: quality estimate is a usual issue in crowdsourcing, but micro-task workers may solve all the assigned tasks at once; we would like to take into account that GWAP players can play the game in different moments with different levels of attention, hence their quality/reliability can change over time and cannot be computed once and for all.

[R2] *Coping with varying difficulty of labelling task*, including possibly multiple classification or even uncertain classification tasks, which means that we cannot make any a-priori hypothesis on the number of redundant labelling actions required to solve each task.

[R3] *Incremental computation of truth inference*: as introduced in Section 1, in GWAPs we would like to aggregate contributions as soon as they are available, because there is no pre-defined time-frame for players' input.

[R4] *Dynamic minimization of the number of required repeated labelling*, to avoid useless redundancy: if a task is "easy" (i.e., it is not controversial or does not require specific competences/expertise to be worked out) we would like to automatically ask fewer players to solve it, while if a task is "hard" we would like the task to automatically remain longer in the game to be "played".

5.2 Algorithm

The approach outlined in Figure 1 is explained in details in the following Algorithm 1. Each time a player starts a game round (line 2), he/she is assigned a set of link refinement tasks to be solved.

Algorithm 1. Incremental Truth Inference Algorithm

```
1  while $T \neq \emptyset$ do
2      $u_k \leftarrow$ GetActiveUser($U$)
       /* measure user reliability on control tasks */
3      $G' \leftarrow$ AssignControlTasks($G, u_k$)
4      $errors \leftarrow 0$
5      foreach $g_i$ in $G'$ do
6          $c_{i,k} \leftarrow$ CrowdsourceAnswer($g_i, u_k$)
7          if $c_{i,k} \neq$ TrueAnswer($g_i$) then
8              $errors \leftarrow errors + 1$
9          end
10     end
11     $\rho_k \leftarrow$ ComputeUserReliability($errors$, size($G'$))
       /* aggregate user answers in truth inference */
12     $T' \leftarrow$ AssignTasks($T, u_k$)
13     foreach $t_i$ in $T'$ do
14         $c_{i,k} \leftarrow$ CrowdsourceAnswer($t_i, u_k$)
15         UpdateSolutionEstimate($t_i, c_{i,k}, \rho_k$)
16         if isTaskSolved($t_i$) then
17             $\hat{y}_i \leftarrow c_{i,k}$
18             $T \leftarrow T - \{t_i\}$
19         end
20     end
21 end
22 return $\hat{Y}$
```

The player provides answers to each task without being able to distinguish between unsolved tasks and control tasks (*cf*. lines 6 and 14). The answers on control tasks are used to compute player's reliability, which is a function of the number of mistakes (lines 5-11); reliability is computed per each game round. There are of course different ways to realize the ComputeUserReliability function of line 11: the simplest way is to use the percentage of correct labels in control tasks, i.e., $\rho_k \leftarrow 1 - errors/\text{size}(G')$. In other cases, it may be safer to strongly penalize players which submit random answers; in the games built with our framework (*cf*. Section 7), to have a conservative estimation, we adopt the following Equation:

$$\rho_k \leftarrow e^{-\alpha \cdot errors} \tag{2}$$

where α is set (for example) so that ρ_k almost halves with 1 mistake and then quickly decreases with further errors.

On the other hand, the answers on unsolved tasks are weighted with the reliability value and used to update the estimation scores (lines 14–15); for each task t_i and for each possible solution v_j, the UpdateSolutionEstimate function is implemented as follows:

$$\sigma_{i,j} \leftarrow \begin{cases} \sigma_{i,j} + \delta \cdot \rho_k & \text{if } c_{i,k} = v_j, \\ \sigma_{i,j} & \text{otherwise.} \end{cases} \tag{3}$$

where $c_{i,k}$ is the answer contributed by the user u_k with reliability ρ_k and δ is an increment that depends on the minimum redundancy required for the task. In other words, the score of the link indicated by the user is incremented and the increment is weighted with the user reliability, while the scores of the other links (not indicated by the user) remain unchanged.

At each truth inference step, the task completion condition is checked (line 16) with Equation (1) and, if it holds, the task solution is returned and the task removed from the game (lines 17-18). The algorithm iterates until all tasks are solved (line 1) and truth is inferred on all tasks (line 22).

5.3 Requirement Satisfaction

Qualitatively, we now assess how the approach presented in this section addresses the requirements listed in Section 5.1 and we discuss some of its positive consequences.

Labelling quality is controlled via the updates of the estimation scores $\sigma_{n,l}$, incremented with players' contributions which are weighted with the reliability values ρ_k. This means that the proposed approach takes into consideration the quality of contributions and "measures" it at each game play, thus relying on a "local" trustworthiness value; the dynamic re-computation of ρ_k fulfills requirement [R1], by addressing the fact that the same player can show a different behavior in different moments of his/her playing, e.g., being careful *vs*. distracted.

The estimation scores $\sigma_{n,l}$, their update function (*cf*. Equation (3)) and the task completion condition (*cf*. Equation (1)) have also other interesting properties. The scores are attributed to each task-label combination and updated at each user contribution.

If a task t_i is "easy", different players will attribute the same label v_j and the respective score $\sigma_{i,l}$ will quickly increase and overcome the threshold $\bar{s}$ of the exit condition. On the contrary, if a labelling task is difficult or controversial, different GWAP players may give different solutions from the set V to the same task t_i, so potentially all scores in $\sigma_{i,*}$ get updated but none of them easily overcomes $\bar{s}$.

In other words, the proposed approach fulfills requirements [R2] on task difficulty, because easy and difficult tasks are automatically detected and treated accordingly, and [R4] on repeated labelling, as the number of players asked to solve the same task is dynamically adjusted.

It is worth noting that in record linkage literature [6], scores are assigned to each possible couple of records, and usually the "matching" score is increased while the "non-matching" scores are decreased respectively. In the cases of possibly multiple labelling and uncertain solutions (*cf*. requirement [R2]), we propose to increase the score of the user-provided solution, without decreasing the score of the alternative solutions. Of course, variations of the update function in Equation (3) can be introduced, depending on the scenario characteristics. For example, if $c_{i,k} \neq v_j$, then $\sigma_{i,j}$ could be decreased of a quantity $\delta' \cdot \rho_k$, where δ' is the decrement amount.

By design, Algorithm 1 fulfills requirement [R3], since each player contribution (line 14) triggers a step of the truth inference estimate (line 15) and leads to the exit condition check (line 16). This incremental approach ensures that the task is assigned to players only until an inferred "true" solution is reached, thus avoiding useless redundancy of labelling (again satisfying requirement [R4]).

The dynamically adjusted repeated labelling has also the consequence of indirectly *estimating task complexity*: indeed we can say that the more contributions are needed to satisfy the exit condition of Equation (1), the more difficult the task. Therefore, whenever an assessment of the task difficulty is required, the number of collected contributions can be adopted as a proxy measure. In our previous work [27] we indeed demonstrated that this empirical measure of difficulty is highly correlated with the (lack of) confidence value resulting from machine learning classifiers applied to the same data.

A final note on task assignment: it is a common best practice to give each task to a crowd worker at most only once and to perform answer aggregation on responses from different workers; this is also true for GWAPs, in that the same player could get bored if requested to solve the same problem over and over. This means that task assignment to player u_k (lines 3 and 12) takes tasks from G and T, respectively among those that u_k never solved before. A pragmatic strategy to avoid using up the entire set G of control tasks, that we usually adopt when implementing GWAPs, is to dynamically increment G by adding the solved tasks from the set T (those removed when the "true" solution is inferred), so line 18 could become: $T \leftarrow T - \{t_i\}$; $G \leftarrow G + \{t_i\}$.

6. THE GWAP ENABLER FRAMEWORK

To give a better idea of our software framework, we give some details on its technical internals. The GWAP Enabler is released as open source with an Apache 2.0 license and is made available at https://github.com/STARS4ALL/gwap-enabler.

6.1 Structure of the Framework

Our GWAP Enabler is a template Web application formed by some basic software building blocks, which provide the basic functionalities to build a linked data refinement game; by customizing the "template", any specific GWAP application can be built. The three main components are the User Interface (UI), the Application Programming Interface (API) and the Data Base (DB).

The main table of the database is named resource_has_topic and contains all the links $l = (r_s, p, r_o) \in \mathcal{L}$. Each link has a score $\sigma \in [0,1]$ which is updated every time it is played by a user. The subject r_s and object r_o resources of links are stored respectively in the tables named resource and topic. For example, referring to the music classification case illustrated in Sections 3 and 4, the resources table should contain the audio tracks and the topics table should list all the possible music styles. These three tables together contain all the data linking problem information and they are initially filled according to the purpose of the specific GWAP.

In addition to those tables, the database contains further information to customize the game according to the desired behavior: the configuration table to change the truth inference parameters and the badge table to change the badges given as reward to players during the gameplay. The remaining tables manage internal game data such as users, rounds, leaderboard or logs and are automatically filled during the gameplay; as such, they do not need to be modified or filled: they can of course be freely adapted at developers' will, being aware that altering those tables will require also modifying the code.

From a technical point of view, the GWAP Enabler is made up of an HTML5, AngularJS and Bootstrap frontend, a PHP backend to manage the logic of the application and a MySQL database, as shown in Figure 2. The communication between frontend and backend happens through an internal Web-based API, whose main operations consist in retrieving the set of links to be shown to players in each game round, updating the link score according to the agreement and disagreement of players and updating the points of the players to build leaderboards and assign badges.

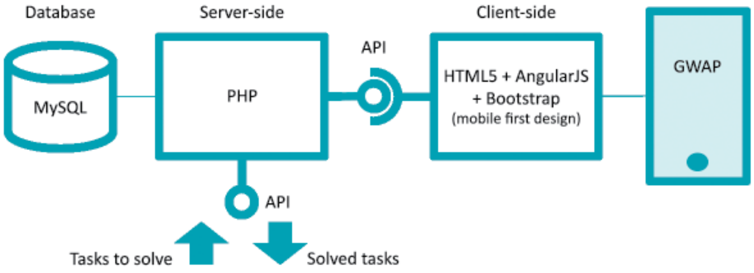


Figure 2. Architecture of the GWAP enabler.

Another external Web-based API is also set up to feed the game with new data and to access the GWAP results aggregated by the truth inference algorithm; in particular the API methods allow submitting new tasks to be solved, retrieving solved tasks (= refined links) in both JSON and JSON-LD formats, getting the number of tasks to be completed and listing the main KPIs of the GWAP (e.g., throughput, average life-play or ALP and expected contribution [17]) for the game evaluation. This API is exhaustively documented at https://gwapenablerapi.docs.apiary.io/.

Further details about the framework architecture can be found in the GitHub repository; the code of the enabler is in the "App" folder whereas the scripts to create the database are in the "Db" folder. Further details about the installation steps, the technical requirements, the database structure and some customization option are given in the wiki pages of the repository at https://github.com/STARS4ALL/gwap-enabler/wiki.

6.2 How to Build a GWAP by Using the Framework

To create a game instance out of the provided template, a developer should perform a series of operations and changes that affect the three building blocks constituting the framework.

First of all, the basic data linking case (*cf*. Section 3) and atomic crowdsourcing task (*cf*. Section 4) should be designed to address the specific use case. Then, the database has to be filled up with data by adding the core resources and links (resource, topic and resource_has_topic tables), the GWAP aggregation parameters (configuration table) and badges information (badge table). Please, note that a pre-processing phase is required to prepare the data and a careful analysis of the specific refinement purpose is an essential step to find and tune the proper parameters, thus this initial step could be long and complex depending on the context.

Once data are in the DB, the code can be run as is or it can be tailored to address specific requirements; for example, game mechanics could be altered, further data to describe resources or links can be added (e.g., maps/videos/sounds) or a different badge/point assignment logic can be defined. Finally, the UI should be customized with the desired look and feel and the specific game elements (like points, badges, leaderboards) could be modified to give the game a specific flavor or mood.

7. EXISTING APPLICATIONS BUILT ON TOP OF THE FRAMEWORK

We used the enabler to build three GWAPs that address the three different classes of data linking. Indomilando game aims to rank a set of cultural heritage assets of Milano, based on their recognizability; Land Cover Validation is an example of link validation game in which users are involved in checking land use information produced by two different (and disagreeing) sources; Night Knights is a game for both link creation and validation in which a set of images has to be classified into a predefined set of categories. Moreover, after its public release as open source, our framework was reused by a different research group to build their own game to collect analogies, and therefore we report also about this further exploitation of our GWAP Enabler.

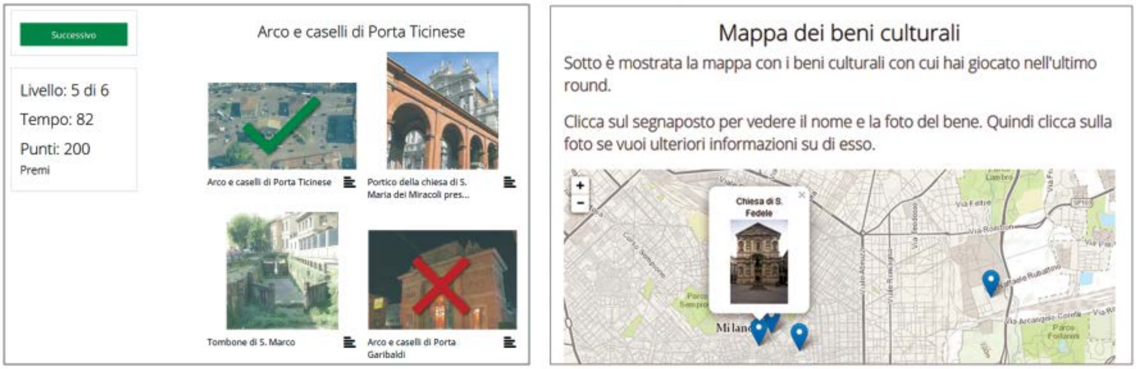
7.1 Link Ranking: Indomilando

Indomilando[5] [26] is a Web-based Game With a Purpose aimed to rank a quite heterogeneous set of images, depicting the cultural heritage assets of Milan. In each round the game shows the name of an asset and four photos, in which one represents the asset and the other three are put as distractors. The user has to choose the right picture related to the asset and, as an incentive, he gains points for each correct answer. A photo is removed from the game when it is correctly chosen three times. All the given answers on the photo (selection or non-selection of the picture) are recorded and analyzed ex-post to measure "how much" the picture represents the asset: the intuition is that, the more a photo is correctly identified by players, the more recognizable it is.

In Indomilando, we have a set of links *l* that connects each photo with the asset it refers to; the assets and the photos are the subjects r_s and objects r_o of the links to be ranked. By counting and suitably weighting the number of times the pictures has been recognized (or not), we calculate the scores σ of these links. Since they represent the probability of the links to be recognized as true, by ordering them we can rank the links, and thus the pictures of the cultural heritage assets of Milan, on the basis of their recognisability.

The output of this game can be employed for various goals: selecting the best pictures representing an asset, understanding if an asset would benefit from further photo collection or evaluating if an asset may require additional promotional campaign because it is less recognized.

From a gamification point of view, users gain points for each correct answer and can challenge other players in a global leaderboard. Another incentive we give to players is the possibility to view on a map the assets they played with and to display their historic and cultural description, as shown in Figure 3: this is an additional learning reward that Indomilando players showed to appreciate. These incentives were very effective and the game had a great success: all the 2,100 pictures we put in the game were played and ranked by 151 users, with a throughput of 125 photo ranked/hour.


Successivo
Arco e caselli di Porta Ticinese
Arco e caselli di Porta Ticinese
Portico della chiesa di S. Maria dei Miracoli pres...
Tombone di S. Marco
Arco e caselli di Porta Garibaldi
Chiesa di S. Fedele
Milano
+
-

Livello: 5 di 6

Tempo: 82

Punti: 200

Premi

Mappa dei beni culturali

Sotto è mostrata la mappa con i beni culturali con cui hai giocato nell'ultimo round.

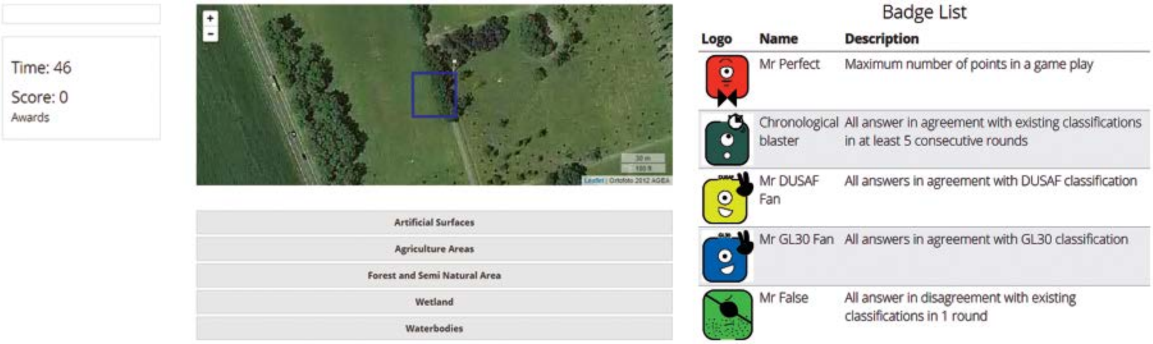
Clicca sul segnaposto per vedere il nome e la foto del bene. Quindi clicca sulla foto se vuoi ulteriori informazioni su di esso.

Figure 3. Indomilando: gameplay (left) and asset visualisation on a map (right).

[5] https://ns3056488.ip-213-32-26.eu/smartculture-games/indomilando.

7.2 Link Validation: Land Cover Validation Game

The Land Cover Validation game[6] [38] is designed to engage Citizen Scientists in the validation of land cover data in the Como municipality area (in Lombardy, Italy). The player is requested to classify the areas in which two different land cover maps disagree: the DUSAF classification[7] made by Lombardy Region and GlobeLand30[8] provided by a Chinese agency. The validation is presented to the user as a classification task: given an aerial photo of a 30x30 square area (pixel), the player has to select the right category from a predefined list of land use types (e.g., residential or agricultural areas), as shown in Figure 4. As regards the incentives and the entertaining environment, players gain points and badges if they agree with one of the existing classifications and they can challenge other players in the leaderboard.



Time: 46
Score: 0
Awards

Artificial Surfaces
Agriculture Areas
Forest and Semi Natural Area
Wetland
Waterbodies

Badge List

Logo	Name	Description
	Mr Perfect	Maximum number of points in a game play
	Chronological blaster	All answer in agreement with existing classifications in at least 5 consecutive rounds
	Mr DUSAF Fan	All answers in agreement with DUSAF classification
	Mr GL30 Fan	All answers in agreement with GL30 classification
	Mr False	All answer in disagreement with existing classifications in 1 round

Figure 4. Land cover validation game: pixel classification (left) and badges (right).

From the data linking perspective, each pixel is the subject r_s of two links, one connecting the pixel with its land cover defined by DUSAF ($r_{o_{DUSAF}}$) and the other with the GlobeLand30 classification ($r_{o_{GL30}}$). Each time one of the two land cover options is selected, the score σ of the corresponding link is increased. This score represents the link truth value and a threshold is set so that all links with a score higher than this value are considered true. When a link score exceeds the threshold, the corresponding pixel is removed from the game.

The game completely fulfilled its goal, since all the target 1,600 aerial pixels were validated, thanks to 68 gamers that played more than 20 hours during the FOSS4G Europe Conference in 2015.

7.3 Link Creation and Validation: Night Knights

Night Knights[9] [27] is a GWAP designed to classify images taken from the International Space Station (ISS) into a fixed set of six categories, developed within a project that aims to increase the awareness about the light pollution problem.

[6] http://landcover.como.polimi.it/landcover.

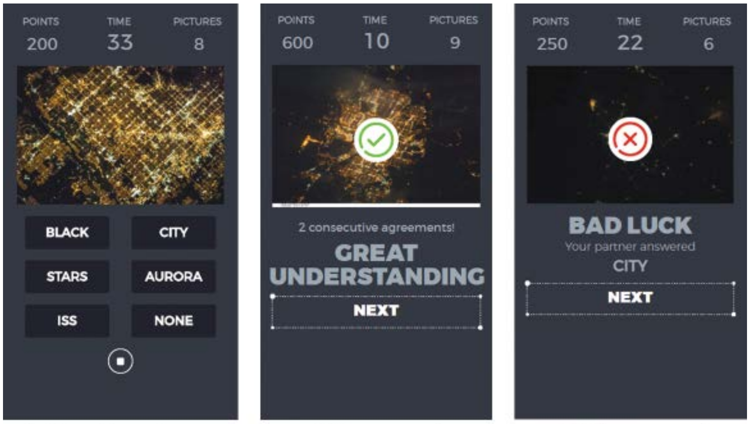
[7] http://www.geoportale.regione.lombardia.it/en/home.

[8] http://www.globallandcover.com/GLC30Download/index.aspx.

[9] http://www.nightknights.eu.

Each human participant plays the role of an astronaut that, coupled with another random player, has the mission of classifying as many images as possible playing one-minute rounds. As Figure 5 shows, if players agree on the same classification they get points, which are collected to challenge other users in a global leaderboard.

The hidden goal of the game is to create new links between each image r_s and its correct category r_o, by cross-validating them using the contributions of multiple users, suitably aggregated. The link creation algorithm works as follows: starting from a set of links connecting each image with all the available categories, the score σ of a link is increased if a player chooses the corresponding category. Each image is offered to multiple players, whose contributions are weighted according to their reliability (measured with assessment tasks) and aggregated in the link score; once the score overcomes a specific threshold, the image is classified and removed from the game. By design, a minimum of four agreeing users are required to reach the classification threshold. More than 35,000 images have been classified since the launch of the game in February 2017 by more than 1,000 players, with a throughput of 203 images classified/hour.



(a) Classify an image (b) Agreement (c) Disagreement

Figure 5. Night Knights: the game play.

7.4 Reusing and Extending the Framework: ARGO

A research group from the University of Milano Bicocca[10], working on analogical reasoning, contacted us: they were aware of our expertise on human computation and gamification and they wanted some suggestions about how to design a game with a purpose application to involve users and collect propositional analogies (in the form $a : b = c : d$). Their objective was, on the one hand, to investigate how people come up with analogies and, on the other hand, to build a gold standard to train a machine learning algorithm to extract propositional analogies from textual resources.

[10] http://www.disco.unimib.it.

After discussing and brainstorming some ideas for the game, we told them about our framework. With no further support from our side, by simply familiarizing with the software through the tutorial (see Section 9) and the related documentation, a Master student was able to customize the framework and, on top of it, develop ARGO, "Analogical Reasoning Game with a purpOse"[11], a simple game to collect analogies.

Figure 6 shows a sample screen, in which users are asked to identify which TV personality is for Italy what David Letterman is for USA, choosing one of the proposed options. In this case, the *data linking task* is an extension of what we described in Section 3: the user is asked to link a partial analogy like "Letterman : USA = ?x : Italy" to the most suitable value for "?x".

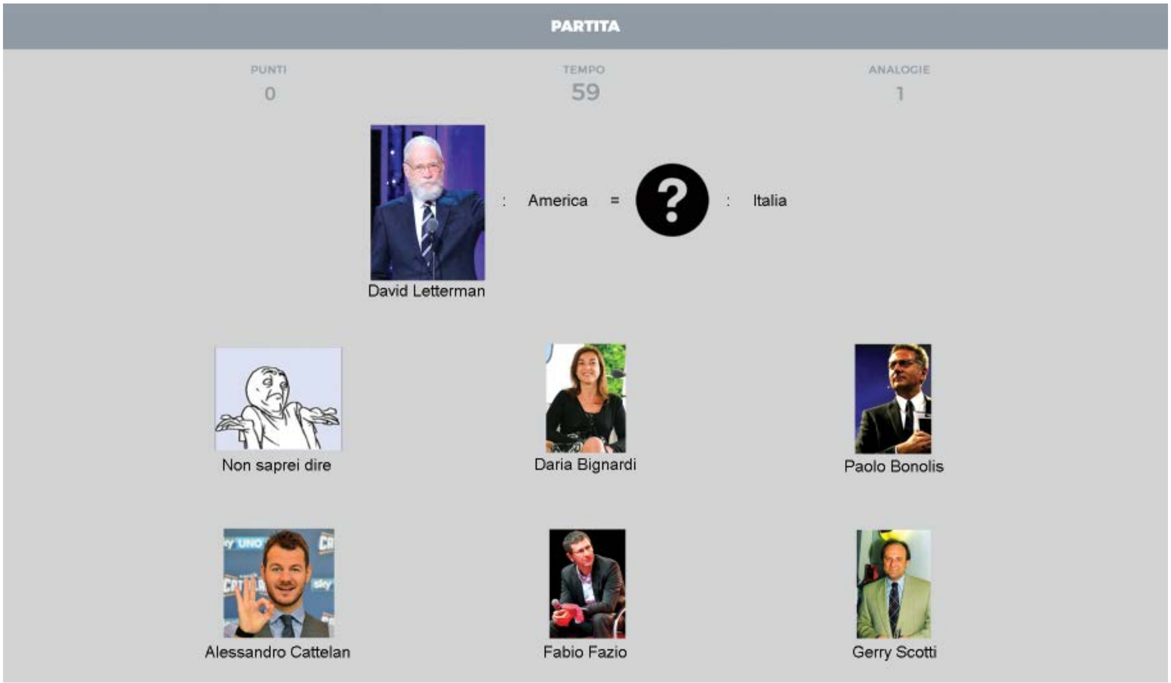


Figure 6. Sample screenshot from ARGO game to collect analogies.

In the course of a short experimentation, during a public event, the researchers from the University of Milano Bicocca were able to involve 90 users and to collect around 1,200 contributions, that were enough for their further analysis and investigation.

[11] http://argo.disco.unimib.it/.

8. EVALUATION OF INCREMENTAL TRUTH INFERENCE

To evaluate the proposed truth inference algorithm we performed a comparative assessment with alternative solutions, on the basis of part of the data collected through two of the GWAPs described in the previous section: the LCV Game and Night Knights.

A first evaluation is based on the total number of contributions to be collected. In most crowdsourcing settings, where aggregation is computed ex-post, a fixed number of contributions is collected per each task. Let's consider the multinomial classification of N tasks with L admissible labels, with a minimum of p agreeing labels per task. To implement an ex-post aggregation with simple majority voting, the total number of needed contributions is the *redundancy* r computed as

$$r \leftarrow N \cdot ((p - 1) \cdot L + 1) \tag{4}$$

Moreover, in traditional micro-work/crowdsourcing settings, there is experimental evidence of 40%-45% of spammers among crowd workers [39, 40], thus redundancy could be even higher than the one computed in Equation (4).

Table 1 shows the theoretical and empirical numbers for LCV Game and Night Knights: the incremental approach that we propose leads to a sensible "saving" (roughly between 40% and 60%) in terms of redundancy, since whenever the minimum number p of contribution is enough to consider the task solved, no more labels are sought. The reduction in terms of repetitions is due to the fact that "easy" tasks get solved with the minimum number of users (3 for LCV Game and 4 for Night Knights), because the users easily agree on the solution. It is also worth noting that the reduction is higher when the number of alternatives (6 labels for Night Knights *vs*. 5 labels for LCV Game) and/or the minimum number of users is higher. On the other hand, this saving could be smaller in presence of a high share of "difficult" or controversial tasks.

Table 1. Number of required contributions for truth inference over N tasks.

GWAP	N	L	p	**Theoretical** r	**Actual** r	**% diff.**
LCV Game	~1,000	5	3	~11,500	~6,400	-44%
Night Knights	~27,700	6	4	~525,000	~205,000	-61%

Note: Regarding L possible labels and a minimum of p agreeing answers: comparison between theoretical redundancy r (under the hypothesis of ex-post aggregation with simple majority voting) and actual numbers as experimentally measured in the two considered GWAPs applying our incremental truth inference approach.

Finally, to assess the ability of our incremental approach to infer the truth, we applied state-of-the-art algorithms for ex-post data aggregation and compared the resulting classification on the contribution collected by our GWAP. Namely, we run expectation maximization (EM, *cf*. [35]) and message passing (MP, *cf*. [36]), which are the most frequently used truth inference algorithms; then, we compared the aggregated labels with a confusion matrix. Table 2 summarizes the results of comparing our algorithm with EM and

MP, respectively. Since we do not have a ground truth, we can only compare how much the results obtained are similar, by computing metrics over the confusion matrix. The results reported in Table 2 show that indeed the overlap between the "truths" inferred with the compared algorithms is very high (accuracy always over 96%) and the agreement statistics confirm it (Kappa statistics and adjusted Rand index very high). This proves the validity and applicability of our approach.

Table 2. Truth inference results comparison between our incremental approach and state-of-the-art techniques.

GWAP	Algorithms	Accuracy	Kappa	Rand
LCV Game	incremental vs. EM	96.1%	93.4%	88.7%
	incremental vs. MP	96.9%	94.7%	90.6%
Night Knights	incremental vs. EM	99.7%	99.4%	99.4%
	incremental vs. MP	99.8%	99.6%	99.6%

Note: EM: expectation maximization, MP: message passing, and various metrics: accuracy of the confusion matrix, Kappa statistics, adjusted Rand index corrected-for-chance [41].

9. A STEP-BY-STEP TUTORIAL TO REUSE THE FRAMEWORK

In the previous sections, we showed how the GWAP Enabler was successfully employed to implement games to create and validate links through an image classification process and to rank links based on their recognizability. Since it is a new resource, in this section, we introduce a tutorial that guides new adopters to customize and adapt the framework to a new use case. All the required changes to the GWAP Enabler sources are explained step-by-step in the GitHub repository at https://github.com/STARS4ALL/gwap-enabler-tutorial.

The goal of this tutorial is twofold; on the one hand, we provide developers with a guided example of an "instantiation" of our framework; on the other hand, we demonstrate how this GWAP Enabler could be used and adapted to build a crowdsourcing Web application to enrich and refine OpenStreetMap data (and, consequently, its linked data version LinkedGeoData), in which users are motivated to participate through fun incentives.

More specifically, the GWAP application built in the tutorial is about classifying OpenStreetMap restaurants on the basis of their cuisine type. Data about restaurants are selected in a specific area (we give the example of the city of Milano, but developers can easily change it to their preferred location); those restaurants with an already specified type of cuisine are taken as "ground truth" (for the assessment tasks to compute player reliability), whereas all the remaining one are the subject resources, target of the classification process.

Players are randomly paired and are shown the restaurant name and position on a dynamic map; the game consists in finding an agreement on the cuisine type, selected in a set of predefined categories (the most widely used in OpenStreetMap). As a result, players contributions are aggregated in the truth inference process that implements the link collection and validation.

To create such an application, some changes to the framework's core functionalities are required; while in the GWAP Enabler by default each resource is displayed to players in the form of an image, in this tutorial scenario we want to show developers how to display both a textual information (the restaurant's name) and an interactive map (the restaurant's position). Therefore, this requires (1) to correctly store the relevant information in the database, (2) to modify the API code to retrieve the additional data and (3) to modify the UI code to display name and map.

In the tutorial, we provide some basic instruction and we explain how to embed the map by using Leaflet[⑫], an open-source JavaScript library for mobile-friendly interactive maps. We do not detail the graphical aspect, letting developers define their desired look-and-feel to give the game a more personal flavor or mood. By going through the wiki instructions, developers can get their up and running GWAP in about half a day and they can gain enough knowledge about the framework to be able to reuse it for their own purpose, since the tutorial touches upon all the relevant modifications.

This tutorial was exploited by the people at the University of Milano Bicocca to create the ARGO game described in Section 7.4. They reported that, in less than a week, they were able to realize their vision by designing the experiment, preparing the data, implementing the game on top of our framework and deploying it. This experience testifies that our GWAP Enabler is indeed an easy-to-use and easy-to-extend resource.

10. DISCUSSION AND CONCLUSION

In this paper, we presented an open source software framework to build Games with a Purpose embedding a crowdsourcing task for linked data refinement. The framework is aimed to help in the process of collecting manually-labelled gold standard data, which are needed as training set for automatic learning algorithms to implement refinement operations (link collection, ranking and validation) on large scale linked data sets and knowledge graphs. In other words, the presented framework helps to simplify the tedious and expensive human process of data collection, letting researchers focus on the subsequent steps of their scientific study and experimentation.

We introduced the data linking cases implemented by the framework to explain its level of generality and potential for reuse; we illustrated the crowdsourcing task and truth inference process to clarify its design and possible customizations. Then, we gave some technical details about the internals of the GWAP Enabler, designed and developed accordingly to the most common Web development best practices, and we demonstrated the framework versatility by describing the diverse applications we built on top of it and an experimental evaluation of our incremental truth inference approach.

Since our framework not only helps collecting data linking task solutions from game players, but also automatically computes data aggregation by means of a specific truth inference approach, we also introduced

[⑫] http://leafletjs.com.

our incremental algorithm, explaining its rationale, its implementation and offering a qualitative and quantitative evaluation of its benefits.

Finally, we presented a step-by-step tutorial as a more detailed documentation and as a means to ease the reuse of this new resource; following the tutorial, a developer is guided to build an entirely new GWAP in a few hours, saving significant coding effort.

We provided all the references to get access to the framework code (released under an Apache 2.0 license) and its online documentation which consists of data schemas, API specification and sample input/output data, technical requirements and installation instructions, guided instruction to customize the GWAP Enabler.

The framework could be further extended to cover other refinement cases like free text labelling (i.e., insertion of new literal objects) or data linking issues related to the choice of different predicates.

AUTHOR CONTRIBUTIONS

The ideas and concepts presented in the paper are the results of at least three years of cooperation between the authors. I. Celino (irene.celino@cefriel.com) focused on data linking, crowdsourcing tasks and incremental truth inference; G. Re Calegari (gloria.re@cefriel.com) and A. Fiano (andrea.fiano@cefriel.com) focused on the framework, its applications, evaluation and tutorial. All authors contributed to the manuscript writing and they edited and reviewed the final version of the article.

ACKNOWLEDGEMENTS

This work was partially supported by the STARS4ALL project (H2020-688135) co-funded by the European Commission.

REFERENCES

[1] H. Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic Web 8(3)(2017), 489–508. doi: 10.3233/SW-160218.

[2] A.J. Quinn & B.B. Bederson. Human computation: A survey and taxonomy of a growing field. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2011, pp. 1403–1412. doi: 10.1145/1978942.1979148.

[3] J. Howe. The rise of crowdsourcing. Wired magazine 14(6)(2006), 1–4.

[4] I. Celino, A. Fiano & R Fino. Analysis of a cultural heritage game with a purpose with an educational incentive. In: International Conference on Web Engineering, 2016, pp. 422–430. doi: 10.1007/978-3-319-38791-8_28.

[5] L. Von Ahn & L. Dabbish. Designing games with a purpose. Communications of the ACM 51(8)(2008), 58–67. doi: 10.1145/1378704.1378719.

[6] I.P. Fellegi & A.B. Sunter. A theory for record linkage. Journal of the American Statistical Association 64(328) (1969), 1183–1210. doi: 10.2307/2286061.

[7] A. Ferrara, A. Nikolov & F. Scharffe. Data linking for the semantic web. In: Semantic Web: Ontology and Knowledge Base Enabled Tools, Services, and Applications, 2013, 169–200. doi: 10.4018/978-1-4666-3610-1.ch008.

[8] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann & S. Auer. Quality assessment for linked data: A survey. Semantic Web 7(1)(2016), 63–93. doi: 10.3233/SW-150175.

[9] C. Fürber & M. Hepp. Using SPARQL and SPIN for data quality management on the semantic Web. In: W. Abramowicz & R. Tolksdorf (eds.) Business Information Systems (BIS 2010). Berlin: Springer, pp. 35–46. doi: 10.1007/978-3-642-12814-1_4.

[10] C. Guéret, P. Groth, C. Stadler, & J. Lehmann. Assessing linked data mappings using network measures. In: Extended Semantic Web Conference, 2012, pp. 87–102.

[11] H. Paulheim & C. Bizer. Improving the quality of linked data using statistical distributions. International Journal on Semantic Web and Information Systems (IJSWIS) 10(2)(2014), 63–86.

[12] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy … & W. Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 601–610. doi: 10.1145/2623330.2623623.

[13] J. Sleeman & T. Finin. Type prediction for efficient coreference resolution in heterogeneous semantic graphs. In: Semantic Computing (ICSC), 2013 IEEE Seventh International Conference, 2013, pp. 78–85. doi: 10.1109/ICSC.2013.22.

[14] J. Sleeman, T. Finin & A. Joshi. Topic modeling for RDF graphs. In: Workshop on Linked Data for Information Extraction (LD4IE) at ISWC, 2015, pp. 48-62.

[15] M. Acosta, A. Zaveri, E. Simperl, D. Kontokostas, S. Auer & J. Lehmann. Crowdsourcing linked data quality assessment. In: International Semantic Web Conference, 2013, pp. 260–276. doi: 10.1007/978-3-642-41338-4_17.

[16] E. Simperl, B. Norton & D. Vrandecic. Crowdsourcing tasks in linked data management. In: Proceedings of the Second International Conference on Consuming Linked Data, 2012, pp. 61–72.

[17] K. Siorpaes & M. Hepp. Games with a purpose for the semantic Web. IEEE Intelligent Systems 23(3)(2008). doi: 10.1109/mis.2008.45.

[18] J. Waitelonis, N. Ludwig, M. Knuth & H. Sack. WhoKnows? Evaluating linked data heuristics with a quiz that cleans up DBpedia. Interactive Technology and Smart Education 8(4)(2011), 236–248. doi: 10.1108/17415651111189478.

[19] J. Chamberlain, M. Poesio & U. Kruschwitz. Phrase detectives: A web-based collaborative annotation game. In: Proceedings of the International Conference on Semantic Systems (I-Semantics' 08), 2008, pp. 42–49.

[20] S. Thaler, E.P.B. Simperl & K. Siorpaes. SpotTheLink: A game for ontology alignment. Wissensmanagement 182(2011), 246–253.

[21] J. Hees, T. Roth-Berghofer, R. Biedert, B. Adrian, & A. Dengel. BetterRelations: Using a game to rate linked data triples. In: Annual Conference on Artificial Intelligence, 2011, pp. 134–138.

[22] I. Celino, S. Contessa, M. Corubolo, D. Dell'Aglio, E. Della Valle, S. Fumeo & T. Krüger. Linking smart cities datasets with human computation: The case of UrbanMatch. In: P. Cudré-Mauroux et al. (eds.) The Semantic Web – ISWC 2012. Berlin: Springer, pp. 34–49. doi: 10.1007/978-3-642-35173-0_3.

[23] I. Celino, D. Cerizza, S. Contessa, M. Corubolo, D. Dell'Aglio, E.D. Valle, & S. Fumeo. Urbanopoly: A social and location-based game with a purpose to crowdsource your urban data. In: Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Conference on Social Computing (SocialCom), 2012, pp. 910–913. doi: 10.1109/SocialCom-PASSAT.2012.138.

[24] C. Wieser, F. Bry, A. Bérard & R. Lagrange. ARTigo: Building an artwork search engine with games and higher-order latent semantic analysis. In: First AAAI Conference on Human Computation and Crowdsourcing, 2013.

[25] I. Celino, I., E. Della Valle & R Gualandris. On the effectiveness of a mobile puzzle game UI to crowdsource linked data management tasks. In: 1st International Workshop on User Interfaces for Crowdsourcing and Human Computation, 2014.

[26] B. Settles. Active learning. Synthesis Lectures on Artificial Intelligence and Machine Learning 6(1)(2012), 1–114.

[27] G. Re Calegari, G. Nasi & I. Celino. Human computation vs. machine learning: An experimental comparison for image classification. Human Computation Journal 5(1)(2018), 13–30.

[28] L. Von Ahn & L. Dabbish. Labelling images with a computer game. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2004, pp. 319–326. doi: 10.1145/985692.985733.

[29] U. Ul Hassan, S. O'Riain & E. Curry. Effects of expertise assessment on the quality of task routing in human computation. In: Proceedings of the 2nd International Workshop on Social Media for Crowdsourcing and Human Computation, 2013. doi: 10.14236/ewic/sohuman2013.1.

[30] Y. Zheng, G. Li, Y. Li, C. Shan & R. Cheng. Truth inference in crowdsourcing: Is the problem solved? Proceedings of the VLDB Endowment 10(5)(2017), 541–552. doi: 10.14778/3055540.3055547.

[31] V.S. Sheng, F. Provost & P.G. Ipeirotis. Get another label? Improving data quality and data mining using multiple, noisy labelers. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 614–622.

[32] B. Mozafari, P. Sarkar, M.J. Franklin, M.I. Jordan & S. Madden. Active learning for crowd-sourced databases. arXiv preprint. arXiv:1209.3686, 2012.

[33] B. Mozafari, P. Sarkar, M. Franklin, M. Jordan & S. Madden. Scaling up crowd-sourcing to very large datasets: A case for active learning. In: Proceedings of the VLDB Endowment 8(2)(2014), 125–136.

[34] E. Law & L.v. Ahn. Human computation. Synthesis Lectures on Artificial Intelligence and Machine Learning 5(3)(2011), 1–121. doi: 10.2200/S00371ED1V01Y201107AIM013.

[35] A.P. Dawid & A.M. Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. Applied statistics(1979), pp. 20–28. doi: 10.2307/2346806.

[36] D.R. Karger, S. Oh & D. Shah. Iterative learning for reliable crowdsourcing systems. In: Advances in neural information processing systems, 2011, pp. 1953–1961.

[37] D.C. Brabham. Crowdsourcing. Cambridge, MA: MIT Press, 2013.

[38] M.A. Brovelli, I. Celino, A. Fiano, M.E. Molinari & V. Venkatachalam. A crowdsourcing-based game for land cover validation. Applied Geomatics 10(1)(2018), 1–11. doi: 10.1007/s12518-017-0201-3.

[39] R. Shah. Spam hurts crowdsourcing but can't kill it (Forbes Contributor Opinions). Available at: https://www.forbes.com/sites/rawnshah/2010/12/17/spam-hurts-crowdsourcing-but-cant-kill-it.

[40] J. Vuurens, A.P. de Vries & C. Eickho. How much spam can you take? an analysis of crowdsourcing results to increase accuracy. In: Proc. ACM SIGIR Workshop on Crowdsourcing for Information Retrieval (CIR'11), 2011, pp. 21–26.

[41] W.M. Rand. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association 66(336)(1071), 846–850. doi: 10.2307/2284239.

AUTHOR BIOGRAPHY

Irene Celino is the Head of the Knowledge Technologies group at Cefriel, where she leads an R&D team and she is Portfolio and Project Manager. With expertise in Semantic Web and Human Computation technologies, her research activities cover the application of such innovative technologies to the design and development of Web applications, search engines, recommendations systems and mobile games, especially in Smart City and transportation-related scenarios. She has over 15 years of experience in over 30 R&D cooperative projects, both at National/Regional level and at European level within FP6, FP7, H2020 and EIT Digital. She is author of over 70 scientific publications in peer-reviewed journals, books and conferences.
ORCID: 0000-0001-9962-7193

Gloria Re Calegari is a researcher at Cefriel. She has a computer science background and her fields of expertise are Data Science and Human Computation technologies. Her research activities cover the design and development of gamified application and Games with a Purpose, next to the development of machine learning solutions that bring together humans and artificial intelligence. During her over 5 years of experience in R&D cooperative projects, both at National and Regional level, she published more than 20 scientific publications in peer-reviewed journals and conferences.
ORCID: 0000-0002-4558-229X

Andrea Fiano is a senior developer at Cefriel. Starting with the development of Web application in .Net, he continued with the development of backend solutions and REST APIs in Java and practiced with Single Page Applications and Progressive Web App in Angular and Node.js. He provides his expertise in the development of customer tailored solutions as well as in supporting the research branch. In particular, he has helped in the field of Human Computation with the development of some Games with a Purpose in the Smart City and Crowdsourcing scenarios.
ORCID: 0000-0002-0963-4689