Gamifying Knowledge Maintenance

Mladjan Jovanovic
Department for Information Engineering and Computer Science
University of Trento, Italy
mladjan.jovanovic@unitn.it

***Abstract* — We look at gamification as a mean to handle Linked Data quality problems that are difficult to solve in an automated way. In particular, we look at the use of word games as a knowledge maintenance strategy that is cost-efficient and accurate in terms of the level of granularity of the errors to be spotted. We have classified the most common quality problems encountered in our knowledge base - Entitypedia. Based on this classification, we have implemented a quality improvement methodology for knowledge maintenance that leverages on gamification. We empirically evaluated how this methodology could efficiently improve data quality in Entitypedia. The results show that gamification-enabled knowledge maintenance is a promising and affordable way to improve the quality of Linked Data.**

Keywords—knowledge maintenance; games with purpose; Entitypedia

I. INTRODUCTION

One of the critical aspects of Linked Data success is related to the varying quality of its sources that results from the combination of data content and data import procedures. This often poses serious problems to developers aiming to seamlessly consume the data in their applications. Linked data sources that are transformed into linked form are highly heterogeneous in terms of structure, format and vocabulary. Some of the quality issues, e.g., missing values, can be easily repaired automatically, but others require manual intervention, e.g., incorrect values or incorrect links.

Entitypedia is a knowledge base which comprises external datasets with typical knowledge management tasks such as ontology alignment, semantic matching, and natural language processing [3]. It describes ground knowledge in terms of entities. Entities are representations of real-world objects that can be found in different contexts in our everyday life. Each entity is described with a set of attributes and relations with other entities. In this respect, Entitypedia includes a schema for describing entities (known as TBox) and entities themselves (known as ABox). Currently it contains around 10M entities described with 80M facts (attributes and relations). It has been incrementally populated with domain knowledge from external data sources that include GeoNames, [1] Wikipedia, [2] YAGO [1], WordNet, [3] and MultiWordNet. [4] Although Entitypedia contains clean data, there is a large amount of mistakes, and corrections are needed to keep this knowledge up to date. We have evaluated an average correctness of 98% [2]. It means that the number of mistakes in triples is around 1.6 M. A triple is defined as (entity_id, attribute_id, attrobute_value).

In this paper, we look into gamification [8] as a fast and cost-efficient way to correct mistakes. In particular, we use word games because there is a lot of them and they are popular. They can take different forms, but have many common elements, such as clue-answer pairs. Aside from textual data, word games can easily handle other types of media, such as images. More concretely, we identified two types of mistakes in Entitypedia: (i) typos (mistakes in syntax) and (ii) disagreements (mistakes in meaning).

We have designed a game framework that implements common elements for different word games, such as clue-answer pairs. On top of the framework, we implemented a set of well-known word games, such as Hangman and Crosswords. The content for the games is imported from Entitypedia (as triples) and adapted to a form suitable for gameplay (as clue-answer pairs). The games can work with both text and images as clues.

We empirically evaluated how word games could be efficiently used to improve Linked Data quality in Entitypedia. The results show that using word games is a promising and affordable way to enhance the quality of Linked Data. In the long run, using word games may address many of the problems that fundamentally constrain the usability of Linked Data on the Web in real-world applications.

The rest of the paper is organized as follows. In Section 2 we show a motivating problem in the context of our entity-centric data management framework. Section 3 describes logical activities behind the data cleaning pipeline. We show our game framework in Section 4. Section 5 illustrates crosswords word game. Then we describe how we measure players' reputation in Section 6. In Section 7 we present the evaluation we have conducted. An overview of the related work is given in Section 8. Our conclusions are given in Section 9.

[1] http://www.geonames.org/ last access date: 15.05.2015.
[2] http://www.wikipedia.org/ last access date: 15.05.2015.
[3] http://wordnet.princeton.edu last access date: 15.05.2015.
[4] http://multiwordnet.fbk.eu/ last access date: 15.05.2015.

II. Motivating Problem

In this section, we explain the practical problem that we address. Entitypedia is a multilingual knowledge base [3]. Its data model is defined following the faceted approach to organize and represent knowledge, and is focused on the notion of domain. A domain is defined as a 4-tuple <C, E, R, A> where:

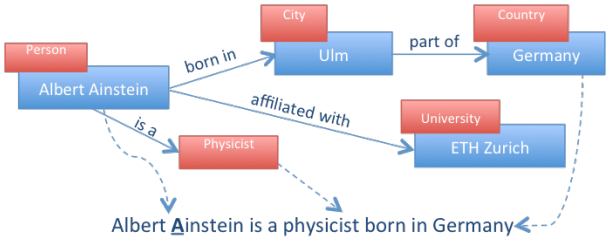
- C is a set of classes,
- E is a set of entities,
- R is a set of binary relations,
- A is a set of attributes.

Knowledge is organized around entities. Entities are representations of real-world objects that can be found in different contexts in our everyday life, as illustrated in Fig. 1. Each entity is described with a set of attributes and relations with other entities. An entity has a reference class that actually determines its type. An entity type is defined in terms of attributes, relations (such as born-in, part-of), services (such as computeAge or computeInverseRelation) and categories of metaatributes (such as mandatory, identifying, permanent, timespan, provenance). There are relatively few common sense entity types (such as person, event) and many application- and context-dependent entity types.

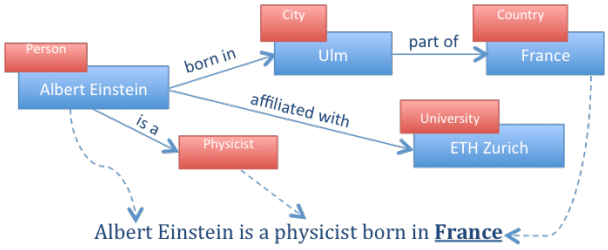
While the combination of machine-driven extraction and human effort is a reasonable approach to produce a baseline version of the resource, there are data quality issues mainly due to the (sometimes) low quality of source data. Data can be incorrect, partially overlapping and inconsistent, or incomplete. Low quality may also result from the data import procedure, such as incorrect or incomplete extraction of objects, incorrect extraction of data types or incorrect links to external resources. All these factors largely influence the quality of the services that can use Entitypedia, such as search, navigation and exploration applications.

In general, two basic types of mistakes can be found in Entitypedia's content, as highlighted by the underlining in Fig. 1:

- **Typos** (mistakes in syntax),
- **Disagreements** (mistakes in meaning).



(a)



(b)

Fig. 1. Simplified examples of mistakes that can be found in entity-centric knowledge graph, (a) syntax-level mistake (typo) and (b) semantic-level mistake (disagreement).

III. Data Cleaning Workflow

We apply data certification pipeline to maintain knowledge, as described in Fig. 2. This pipeline is a five-stage process, defined as follows.

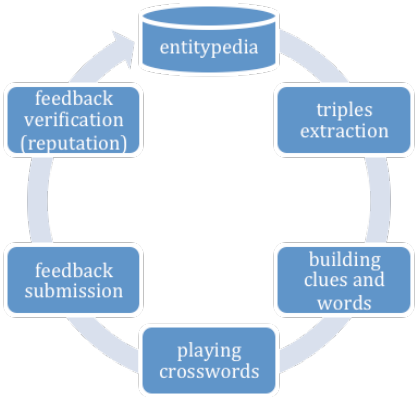


Fig. 2. Data certification pipeline. Entity attribute-value pairs are expressed as word-clue pairs to be used in word games.

The general idea is to embed data directly into crosswords and have such games solved by many players. The Entitypedia repository works as the source of data for certification. The activities are organized as follows:

- **Step 1:** we take the Entitypedia's content and extract it into the different triple forms, such as *entity-relation-attribute* or *attribute-relation-value*. Triples provide the necessary information to generate game content. Fig. 3 gives an example of such a triple. Triples extraction allows for generality and flexibility in selecting specific data configurations to certify.
- **Step 2:** triples are transformed into clue-word pairs that are common for many word games (as illustrated in Fig. 3). From there, games select the content that is most appropriate to them and present it to players.
- **Step 3:** this step is concerned with gameplay (further discussed in Section V). There are two basic types of gameplay:
  - creating crossword puzzles, and
  - solving crossword puzzles.
- **Step 4:** during gameplay players submit feedback. They spot mistakes and provide corrections in either a word or a clue. Fig. 6 shows the user interface.

- **Step 5:** quality of the feedback is measured against players' reputation. The feedback verification stage corresponds to a final quality control of submitted feedback (further described in Section VI).

Fig.3 illustrates the way data is used for crossword puzzles. It shows an excerpt of the knowledge graph from Entitypedia, which contains three entities: an entity representing a person, Leonardo Da Vinci, a city of Florence, where he was born, and a country, Italy, which Florence is part of. The entity graph displays also entity types. All this information can be used to create clue-answer template. Such template describes a particular configuration of entities in the graph and contains a textual phrase with blanks. Usually there are many configurations where such template applies. An example would be as per Fig. 3, the phrase "an X born in Y", where X is constrained to be an entity type Person, with X representing the profession, and Y being the location where the Person was born. In our example, the X is Leonardo Da Vinci, the artist, and the Y is Italy, the country.

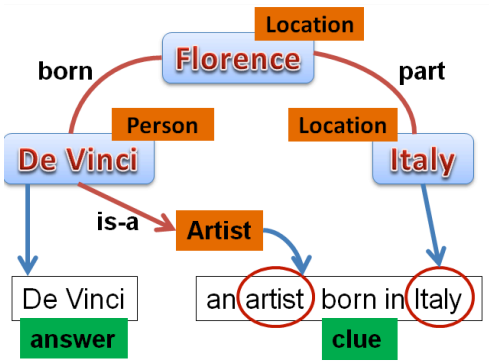


Fig. 3. Template for mapping triples from entity-centric knowledge graph into word games primitives (word and clues).

IV. Game Framework

To implement the pipeline, we have designed and developed a set of tools organized as word games framework, described in Fig. 4. The framework contains common word game components (content management, feedback management, expertise, reputation management) and a set of word game implementations.

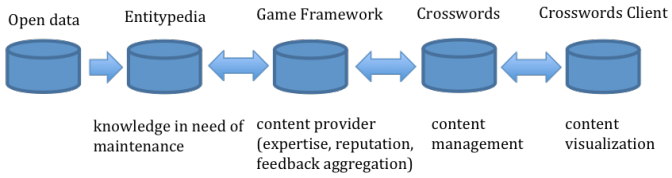


Fig. 4. Word games framework.

The Entitypedia provides content for the games, namely, the data for verification. As explained in the Introduction, it is created from existing Open Data sources.

The Game Framework provides content for word games and several services, such as player profiles, reputation computation, and feedback management. Entitypedia content is represented by *words* and *clues*, which are created from entities using *templates* (exemplified in Fig. 3). The template is a starting point for content generation. It identifies certain entity configurations in the knowledge base and extracts enough information about such configuration to generate word game content.

The Crosswords Client visualizes the generated content and implements the twofold interaction with the player (further described in Section V). Namely, the direction from Entitypedia to games represents the flow of data to be verified and the reverse direction represents the flow of feedback on such data.

V. Crosswords

The Crosswords game [5] allows players to create or solve word puzzles, identify mistakes in clue-answer pairs, and submit feedback according to a pre-defined data certification pipeline. Our implementation of Crosswords exploits common features of crossword puzzles. However, we have implemented additional features needed for knowledge maintenance, such as fixing mistakes and reputation-based quality control. In this respect, there are two principal ways of engaging the players in data certification: crosswords creation and crosswords solving.

A. Crosswords creation

The game allows creating crossword puzzles in two ways: manual and assisted mode. The manual mode relies only on the player's knowledge. That is, the player creates or picks the layout, creates the grid and writes the clues. The assisted mode of creating a puzzle is computer-aided. An illustration of the user interface to this end is given in Fig. 5. In this mode, the involvement of the computer is that of helping the player, rather than substituting the author. Most popular computer aids for filling the grid are computerized dictionaries, word lists and various masked search tools. The clue writing process is still completely manual, although there are databases with crossword clues.

The players are likely to report errors in the content they use to build the puzzle. The manual mode gives players freedom and engages the audience. However, it may require certain level of domain knowledge. This is why it can be combined with assisted crossword editing. Assisted crossword editing features a tool to choose words which fit the grid, and to choose clues for words. This tool operates on content from Entitypedia. The entities from the repository are transformed and displayed to the authors in the clue-answer format, making the puzzle creation easier. In Fig. 5, coloured bars next to each word denote difficulty. Word frequencies are a convenient measure of familiarity of the word. We use word frequencies from the Google Ngram, [6] corpus to measure the difficulty of the words that we use in our games. Clue difficulty is more difficult to measure. Here we use following criteria:

[5] Crosswords Word Game: http://games.entitypedia.org/crosswords/
[6] https://books.google.com/ngrams last access date: 15.05.2015.

- the familiarity of words used in the clue itself,
- the amount of information included in the clue, such as the number of entity attributes used to generate the clue.

We use the same 5-level rating we used for word difficulty to rate the hardness of clues. When authors use clues and answers based on this data, they actively engage with the content and therefore there is a high probability that if the data contains a mistake, it will be noticed and reported. The provided tool includes a mechanism for reporting mistakes, namely feedback submission.

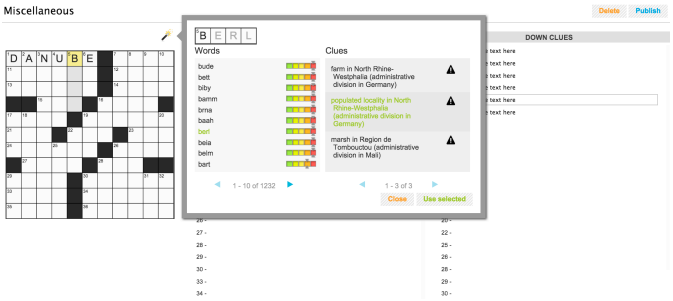


Fig. 5. Tool for assisted clue editing.

While creating a puzzle, the player usually pays close attention to the content provided and therefore is likely to notice mistakes. Such mistakes can be reported using a feedback submission form (invoked by clicking on the exclamation triangle icon next to the clue). Fig. 6 shows the feedback submission form for a word-clue pair. The form displays information following the original template used for the creation of the word-clue pair. Corrections can be provided either for the word or for the clue.

Fig. 6. Feedback submission form.

B. *Crossword solving*

The second way to clean the data is to actually solve crossword puzzles. The user interface to this end is illustrated in Fig. 7. When solving a crossword puzzle, players pay close attention and engage with the content. This raises the probability that errors in the content will be spotted and reported. Namely, while a player is solving the puzzle, active processing of information during her answer search loads it into her short-term memory and makes it easy to notice discrepancies in the data. Moreover, on this step, checked cells in layouts allow for checking the answer. This is important for two reasons: first, it gives players hints in the form of crossing clues, thus making the task easier; second, it makes mistakes more obvious and easier to notice.

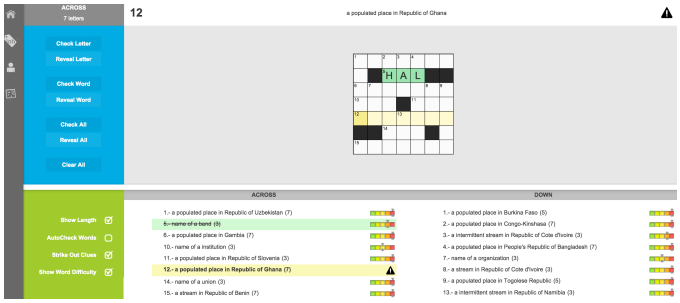


Fig. 7. Basic interface for solving crosswords.

In the case of crossword solving, players provide two types of feedback, explicit and implicit.

Explicit feedback refers to spotting and fixing mistakes in words and clues. It is used to provide corrections and, based on the correction, it is collected per word-clue pair. Any element in the pair can be incorrect and fixed as such.

Implicit feedback is a result of playing the game and giving correct answers (implicit positive feedback) and incorrect answers of the correct length (implicit negative feedback). It results from a player's actions in the game. For example, correct guess from the first attempt without any assistance indicates the player's confidence in the Entitypedia triple used to generate that clue-answer pair.

We use both kinds of feedback to calculate a player's reputation.

VI. Computing Reputation

After having described the Crosswords game, we provide a close look into the quality control mechanism, namely player reputation. We measure a player's reputation by calculating two characteristics: confidence and correctness. We use implicit and explicit feedback to calculate these characteristics.

A. *Computing Confidence*

Confidence reflects how well the player knows the fact used as a basis for the clue. Let us consider a typical case in word games, a single word as highlighted in Fig. 8.

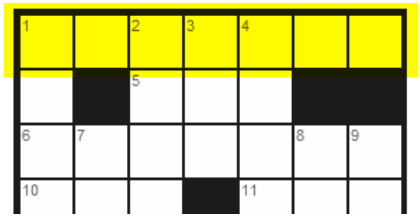


Fig. 8. Fragment of puzzle layout.

Crosswords contain letters that belong to two clues (checked letters), and letters that belong to one clue only (unchecked). For example, a top left cell contains a checked letter: across and down clues check it. The letters next to it, both across and down are unchecked: they only belong to one clue each. The ratio of checked to unchecked letters varies by crosswords style. Unchecked letters should be known, guessed

or revealed, while checked letters can be guessed by a crossing clue. Players can also reveal both kinds of letters separately as a single letter or as a part of a revealed clue.

After analyzing players' implicit feedback, we make the following assumptions about players in word games:

- they fill known words first;
- they complete known words in streaks (uninterrupted sequences of keystrokes);
- they need less assistance (checked letters) for known words.

In addition to this, we know both clue and answer difficulty. We can calculate a player's confidence in the fact expressed by the triple (entity_id, attribute_id, attribute_value) by the length of "winning streak". That is, we look at how many letters a player typed sequentially minus the letters present in the grid before the streak's start. We then discount that by clue difficulty. Harder clues are usually more "distant" from the right answer. In this case players might need more help recalling the answer even if they know it confidently. We account for difficult by discounting 30% of known letters for each level of difficulty, word and clue combined. Therefore our discount goes from 0 to 30% of known letters, rounded up to the nearest integer. In other words, in the hardest clue and answer possible, a player can reveal up to 30% of word letters before finishing the word and still be considered knowing the word confidently. This leads us to the following formula to compute the confidence:

$$confidence = \begin{cases} 1 - \dfrac{kL * (1 - 0.03 * d)}{wL}, & if\ kL < wL \\ 0, & if\ kL = wL, \end{cases}$$

where *kL* (known letters) means the amount of letters visible before the winning streak (completion of the word), *d* is a sum of word and clue difficulty and varies from minimum 2 to maximum 10 and *wL* is the word length. *Confidence* varies from 1, meaning fully confident (the case when the answer was typed in without any assistance), to 0 (the case when the answer was revealed). Confidence is assigned per-user per statement, that is, per user per triple (entity_id, attribute_id, attribute_value).

Table 1. Confidence values for 5-letter words of various difficulties.

difficulty	known letters					
	0	1	2	3	4	5
2	1.0000	0.8120	0.6240	0.4360	0.2480	0.0000
3	1.0000	0.8180	0.6360	0.4540	0.2720	0.0000
4	1.0000	0.8240	0.6480	0.4720	0.2960	0.0000
5	1.0000	0.8300	0.6600	0.4900	0.3200	0.0000
6	1.0000	0.8360	0.6720	0.5080	0.3440	0.0000
7	1.0000	0.8420	0.6840	0.5260	0.3680	0.0000
8	1.0000	0.8480	0.6960	0.5440	0.3920	0.0000
9	1.0000	0.8540	0.7080	0.5620	0.4160	0.0000
10	1.0000	0.8600	0.7200	0.5800	0.4400	0.0000

Table 1 illustrates the confidence formula in action. The horizontal axis gives a number of letters present in the grid before completion of the word, whereas the vertical axis indicates difficulty as a sum of the word and the clue. If we look at a single row, we can notice that as the number of known letters increases, our confidence measure decreases as it becomes easier to guess a word. We can also see that for the fixed number of known letters confidence increases with more difficult words and clues.

B. Computing Correctness

Correctness shows how correctly a player performs. To calculate correctness we introduce ground truth (GT) and extrapolate player correctness on the ground truth to the rest of player's contributions. To introduce ground truth, we mark certain statements, that is, triples (entity_id, attribute_id, attribute_value) as definitely correct (positive) or definitely wrong (negative). These marked statements constitute our body of ground truth against which we measure a players' correctness. We should note that marking a statement as being correct means that we have a correct answer. However, marking the statement as being wrong means that the answer that is provided is wrong, but the correct answer might actually be unknown or not provided. We expect that the amount of negative ground truth is negligible, based on the estimation of knowledge base correctness at 98%, nevertheless, we consider such cases.

Having established the ground truth, we take into account both kinds of feedback - implicit (coming from gameplay) and explicit (coming from report error form) - to calculate player correctness. Let us consider possible cases:

- *Positive ground truth (GT+)*. Suppose we have a positive ground truth statement "Rome is a capital of Italy", and the clue-answer pair "capital of Italy"-"Rome".
  - *explicit feedback (EF)*
    - *correct (EF+):* ignored; player feedback: "Rome is a capital of Italy". This is a repetition, probably not intentional, and it is discarded.
    - *incorrect (EF-):* decreases correctness; player feedback may vary from simpler "Rome is NOT a capital of Italy" to more specific "Paris is a capital of Italy". Since this is the ground truth statement, the player is wrong here and therefore we decrease the player's correctness value.
  - *implicit feedback (IF)*
    - *correct (IF+):* increases correctness; player typed "Rome" as an answer. This is correct and player correctness value increases, taking confidence into account.
    - *incorrect (IF-):* ignored; player typed "Baku", then "Oslo". This input is wrong and it is ignored, interpreted as guesses.

- *Negative ground truth (GT-)*. Suppose we have a negative ground truth statement "Rome is a capital of Greenland", and the clue-answer pair "capital of Greenland"-"Rome".
    - *explicit feedback (EF)* increases correctness. A player reports the error, marking the clue-answer pair as wrong and (optionally) provides correct value. We increase the correctness value because the player has spotted known mistake.
    - *implicit feedback (IF -> EF)* invloves asking for explicit user feedback. A player types "Rome", either by belief or to fill the grid and check the clues. It is not possible to tell this apart, therefore at this point the game can tell the player that the answer "fits the grid" and from the game's point of view (points, bonuses, achievements, etc) it is considered correct. But actually is incorrect and the game asks for the correct value.

For a positive ground truth, explicit correct feedback is ignored, because it is just repetition and is not relevant. Explicit incorrect feedback decreases correctness. Implicit correct feedback increases correctness, taking confidence into account. Implicit incorrect feedback is ignored, as it might represent guesses.

For a negative ground truth, explicit feedback increases correctness and provides confirmation of the error. Implicit feedback might be used to generate a hypothesis to test. In this case, implicit feedback might be interpreted as a player trying to provide the correct value (which might not fit the grid).

The above categories count as positive or negative. The value of correctness varies from 0 to 1. Users with an empty intersection between their feedback and ground truth are assigned a neutral correctness value of 0.5.

With respect to the cases above, we have a formula to calculate overall player correctness using player feedback on ground truth:

$$Correctness = \frac{\sum_{i \in GT^+} Conf_i * IF_i^+ - \sum_{i \in GT^+} EF_i^-}{Count(GT^+)} + \frac{\sum_{i \in GT^-} EF_i}{Count(GT^-)},$$

where GT^+*(GT-)* is player feedback on positive (negative) ground truth, $Conf_i$ is a confidence value of implicit correct feedback item IF_i^+, $Count(GT^+)$ is the amount of player feedback items on positive ground truth, $Count(GT^-)$ is the amount of player feedback items on negative ground truth. Each feedback item is considered as having a numerical value of 1.

VII. Evaluation

In this section, we describe the results of a user study we have carried out. In our evaluation we investigated the following research question: **(RQ)** Whether players can spot and correct the mistakes contained in a puzzle game content, and thereby improve correctness of the data contained in the knowledge base. For this purpose, we measured the amount of corrections (feedback items) that participants submitted while either solving or creating crossword puzzle games. In the following, we describe the experiment and discuss the results from the study.

A. Experiment Design

There were 70 participants in the experiment. The participants were non-native speakers of English with a degree in computer science. We split them into two groups: crossword solvers and crossword builders. We asked crossword solvers (58) to solve 3 crosswords. We asked crossword builders (18) to create 2 crosswords.

We had 16 crossword puzzles in the system available for solving. A puzzle contained 37 clues on average.

Crossword builders were allowed to create new puzzles. There were 1,430,596 answers (words) and 2,730,719 clues available for creating crossword puzzles. The majority (around 1,3 million) of answer-clue pairs consisted of names of places (countries, villages, cities, rivers, etc). A smaller amount (approximately 120.000) of answer-clue pairs included common English words, their definitions, and relations among them.

We introduced a small percentage of mistakes into the puzzle contents. There were two categories of mistakes: typos in the clues or answers, and disagreements between the clue and the answer. The amount of mistakes varied from 0 to 5 mistakes per puzzle. For the experiment, there were 626 answer-clue pairs in the puzzles, of them 25 contained mistakes.

B. Results

The experiment was scheduled to run for 7 days. During this time, participants created 8 crosswords and solved 13 crosswords with 130 game instances in total.

Out of 25 intentional mistakes, players detected a total of 9 triples as erroneous and reported them through 17 feedback items. After obtaining the results, we classified them using the given taxonomy. A summary of these observations is shown in Table 2.

Table 2. Intentional mistakes and feedback distribution for puzzle games.

Number of mistakes per puzzle	Number of puzzles	Number of reported mistakes			Feedback count		
		typo answer	typo clue	dissag.	typo answer	typo clue	disagg.
1	6	1			1		
		-	-	1	-	-	1
2	4	4			9		
		4	-	-	9	-	-
3	2	4			7		
		2	1	1	4	2	1
5	1	-			-		
Total	**13**	**9**			**17**		
		6	**1**	**2**	**13**	**2**	**2**

Most of the reported feedback (88%) refers to typos, of them 13 for the answers and 2 for the clues. This can be expected since spotting disagreement may require a certain level of knowledge in the domain, whereas the presence of contextual information (either in clue or answer) for the typos may ease their identification. All intentional mistakes were reported while solving puzzles. From Table 2 we can notice that the number of feedback items follows the increase in introduced mistakes per puzzle. Combination or intersection of different clues in a puzzle makes it easier to perceive incorrect answers or clues by players.

An interesting fact is that, aside from intentional mistakes in triples, participants reported mistakes that already existed in the puzzle content, as reported in Table 3. In particular, they noticed 12 incorrect clue-answer pairs, out of which 11 refer to typo mistakes, 3 in answers and 8 in clues.

We also measured the correctness of the feedback submitted by participants. Feedback correctness was measured against ground truth facts for answer-clue pairs. In addition, we classified incorrect feedback into false feedback and wrong format feedback. For example, feedback `yard` as a correction for the typo in clue-answer pair `afre – Unit of area, often compared to a football field (noun),` is considered to be wrong. In this case, `acre` is correct answer. However, if for the same pair a player submits feedback as `should be acre`, it is considered as being in wrong format.

Table 3 reports on correctness of feedback related to intentional and unintentional mistakes. Unintentional mistakes were already present in puzzle content. They may originate from source, open data, or they may result from the data import process. The participants noticed 12 original mistakes versus 9 introduced. If we look at feedback correctness, we can notice a higher percentage of correct feedback for unintentional mistakes. In particular, we found 77% correctness for unintentional mistakes and 65% correctness for intentional mistakes. In total, we had 30 feedback items where 21, i.e., 70 %, was correct. Out of 9 incorrect feedback items, the majority (67%) is in wrong format and a smaller amount (33%) is false. However, this may be the user interface issue related to user experience.

Table 3. Feedback contributed for different types of mistakes.

Mistake type	Number of reported mistakes	Total feedback count	Correct feedback	Incorrect feedback	
				false	wrong format
Intentional	9	17	11	6	
				1 (a)	5
Unintentional	12	13	10	3	
				2	1
Total	**21**	**30**	**21**	**9**	
				3	**6**

C. Discussion

Referring back to the research question formulated at the beginning of this section, our experiment let us understand the effectiveness of using gamification to improve knowledge quality. Basically, we had two types of mistakes in the knowledge base – the ones that were already present (unintentional) and the ones that we introduced for the purpose of the experiment (intentional). Participants reported a higher number of unintentional mistakes, 12 against 9 intentional. This may explained by the fact that introduced mistakes can be biased by human knowledge, preferences, opinions and attitudes. For example, preferential use of known facts that may not be familiar to a respondent. These factors might make them more difficult to spot.

We measured and compared the feedback produced for each of the two categories of mistakes against a manually defined golden standard. Overall, feedback correctness was 70%. We showed that for both kinds of mistakes, gamification is a feasible solution to enhance the quality of the data contained in a large entity-centric knowledge base, such as Entitypedia.

VIII. Related Work

Our work is situated in a larger research area concerned with human computation, namely crowdsourcing [15] and gamification [17] for linked data management and quality improvement. We have decided to use gamification as a human computation technique. We made this choice for two major reasons. First, it is a cost-efficient approach as the incentive is non-monetary. This is important if we look at the volumes of existing open data knowledge bases and acknowledge the fact that their content continually changes and evolves. Second, we tried to utilize existing games that people are familiar with and play in general in order to outreach the user community.

At a more technical level, specific linked data management tasks have been subject to human computation. The examples include games for ontology alignment [6], for building domain ontologies from linked data [7], or interlinking of open datasets [13]. On the other hand, crowdsourcing has been used for guiding entity resolution algorithms to produce accurate results where humans are asked questions to resolve data records [11]. Here we can also find approaches that rely on the the crowd to create linked data [12] or to taxonomize large datasets [16]. Existing frameworks for management of the Web Linked Open Data are often limited in their ability to produce interpretable results, for they require user expertise or they are bound to a given data set.

Regarding linked data quality improvement, researchers have mainly analyzed the quality of Web open data. The research described in [4] proposes a methodology to discover quality issues in DBPedia. They combine MTurk and TripleCheckMate [14] in a contest form to find and verify mistakes in triples. However, crowd engagement to fix the mistakes is left to be implemented.

A general solution for gathering human annotations for different types of media is introduced with CrowdTruth, a crowdsourcing annotation platform [5]. Instead of the traditional inter-annotator agreement, it implements disagreement-based metrics to evaluate the data quality issues, such as ambiguity and vagueness.

To our knowledge, we are the first that designed word games for maintaining knowledge bases. Our word games implement the complete Linked Data maintenance process, including correction of mistakes and quality control of the corrections. They are general and flexible in a sense that they can work with Linked Data coming from different domains and represented in different formats (both text and images).

IX. Conclusion

In this paper, we presented a data certification pipeline that exploits gamification. The pipeline is implemented as a word games platform that takes content from the Entitypedia knowledge base, transforms the content into a form suitable for gameplay and brings corrections back from the crowd. We selected a subset of Entitypedia content with known (intentional) mistakes (referring to typos and disagreements) and asked players to provide corrections while solving or creating crossword puzzles.

The evaluation showed that the approach is successful; in particular, the experiment revealed that players with no expertise in knowledge management can be a useful resource to identify given quality issues in an accurate and affordable manner, by playing crossword puzzles. Moreover, the participants identified unintentional mistakes that existed in the content.

As a main point for improvement, we see work on user engagement. In this respect, our future work will follow two directions. The first direction includes promoting the games and content creation. The second is concerned with doing user studies to understand and implement human incentive mechanisms in our games.

Acknowledgements

The research described in this paper is joint work with Fausto Giunchiglia and Aliaksandr Autayeu to whom I express gratitude for collaboration and contribution. The work is supported by European Union's 7th Framework Programme projects ESSENCE Marie Curie Initial Training Network (GA no. 607062).

References

[1] Suchanek, F. M., Kasneci, G., Weikum, G.: Yago: A large ontology from wikipedia and wordnet. Web Semantics: Science, Services and Agents on the World Wide Web vol. 6 no. 3, 2008, pp. 203-217.

[2] Maltese, V.: Enforcing a semantic schema to assess and improve the quality of knowledge resources. International Journal of Metadata, Semantics and Ontologiesto, 2015, to appear.

[3] Giunchiglia, F., Maltese, V., Biswanath, D.: Domains and context: first steps towards managing diversity in knowledge. Web Semantics: Science, Services and Agents on the World Wide Web vol. 12, 2012, pp. 53-63.

[4] Acosta, M., Zaveri, A., Simperl, E., Kontokostas, D., Auer, S., Lehmann, J.: Crowdsourcing linked data quality assessment. The Semantic Web – ISWC. Springer Berlin Heidelberg, 2013, pp. 260-276.

[5] Oana, I., Khamkham, K., Cristea, T., Dumitrache, A., Rutjes, A., van der Ploeg, J., LuRomaszko, L., Aroyo, L., Jan Sips, R.: CrowdTruth: Machine-Human Computation Framework for Harnessing Disagreement in Gathering Annotated Data. The Semantic Web – ISWC. Springer International Publishing, 2014, pp. 486-504.

[6] Thaler, S., Siorpaes, K., Simperl. E.: Spotthelink: A game for ontology alignment. In Proceedings of the 6th Conference for Professional Knowledge Management, 2011, pp. 246-253.

[7] Markotschi, T., Volker, J.: Guesswhat?! - human intelligence for mining linked data. In Proceedings of the Workshop on Knowledge Injection into and Extraction from Linked Data at EKAW, 2010, paper no. 9.

[8] Deterding, S.: Gamification: designing for motivation. ACM Interactions vol. 9 no. 4, 2012, pp. 14-17.

[9] Quinn, J., Bederson, B.: Human computation: a survey and taxonomy of a growing field. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2011, pp. 1403-1412.

[10] Von Ahn, L., Dabbish, L.: Designing games with a purpose. Communications of the ACM vol 51 no 8, 2008, pp. 58-67.

[11] Wang, S., Lofgren, P., Garcia-Molina, H.: Question selection for crowd entity resolution. In Proceedings of the VLDB Endowment vol. 6, 2013, pp. 349-360.

[12] Abdulbaki, U.: Linked crowdsourced data-Enabling location analytics in the linking open data cloud. In Semantic Computing (ICSC), 2015 IEEE International Conference on. 2015, pp. 40-48.

[13] Celino, I., Contessa, S., Corubolo, M., Dell'Aglio, D., Della Valle, E., Fumeo, S., Krüger, T.: Linking smart cities datasets with human computation–the case of urbanmatch. The Semantic Web – ISWC. Springer Berlin Heidelberg, 2012, pp. 34-49.

[14] Kontokostas, D., Zaveri, A., Auer, S., Lehmann, J.: Triplecheckmate: A tool for crowdsourcing the quality assessment of linked data. Knowledge Engineering and the Semantic Web. Springer Berlin Heidelberg, 2013, pp. 265-272.

[15] Kittur, A., Nickerson, J. V., Bernstein, M., Gerber, E., Shaw, A., Zimmerman, J., Horton, J.: The future of crowd work. In Proceedings of the conference on Computer supported cooperative work. ACM, 2013, pp. 1301-1318.

[16] Bragg, J., Weld. D.: Crowdsourcing Multi-Label Classification for Taxonomy Creation. In Proceedings of the First AAAI Conference on Human Computation and Crowdsourcing, 2013, pp. 25-33.

[17] Siorpaes, K., Hepp, M.: Games with a Purpose for the Semantic Web. IEEE Intelligent Systems vol. 23 no. 3, 2008, pp. 50-60.