Using microtasks to crowdsource DBpedia entity classification: A study in workflow design

Qiong Bu, Elena Simperl, Sergej Zerr and Yunjia Li
School of Electronics and Computer Science, University of Southampton
E-mail: {q.bu, e.simperl,s.zerr, yl2}@soton.ac.uk

Abstract. DBpedia is at the core of the Linked Open Data Cloud and widely used in research and applications. However, it is far from being perfect. Its content suffers from many flaws, as a result of factual errors inherited from Wikipedia or glitches of the DBpedia Information Extraction Framework. In this work we focus on one class of such problems, un-typed entities. We propose an approach to categorize DBpedia entities according to the DBpedia ontology using human computation and paid microtasks. We analyzed the main dimensions of the crowdsourcing exercise in depth in order to come up with suggestions for workflow design and study three different workflows with automatic and hybrid prediction mechanisms to select possible candidates for the most specific category from the DBPedia ontology. To test our approach we run experiments on CrowdFlower using a dataset of 120 prevously unclassified entities, and evaluate the answers of the crowd. Our study shows that the microtask based free text approach achieved the highest precision at moderate cost compared to other workflows. However, each workflow has its merit and none of the worflows seems to perform exceptionally well on entities that the DBpedia Extraction Framework fails to classify. We discuss these findings and their potential implications for the design of effective crowdsourced entity classification in DBpedia and beyond.

Keywords: task design, workflow design, entity classification, DBpedia, microtask crowdsourcing, CrowdFlower

1. Introduction

DBpedia is a community project, in which structured information from Wikipedia is published as Linked Open Data (LOD) [1]. The resulting dataset consists of an ontological schema and a large number of entities covering, by virtue of its origins, a wide range of topics and domains. With links to many other sources in the LOD Cloud, it acts as a central hub for the development of many algorithms and applications in academia and industry. Still, no matter how popular, DBpedia is far from being perfect. Its many flaws have been subject to extensive studies and inspired researchers to design tools and methodologies to systematically assess and improve its quality [2,3,4].

In this paper we focus on a particular class of errors, un-typed entities. According to the latest statistics [5], the English DBpedia (version 3.9) contains 4.58 million things, but only 4.22 million among them are classified according to the DBpedia ontology. This is due to several factors, including glitches in the extraction process, but also incomplete or incorrect mappings from Wikipedia infoboxes to the DBpedia ontology[1], and any attempt to fix the problem will most likely require human intervention [6].

DBpedia relies on a community of volunteers and a wiki to define and maintain a collection of mappings in different languages[2]. However, this is a process that is primarily designed for domain experts; it requires a relatively high degree of familiarity with knowledge engineering and the intricacies of both Wikipedia and DBpedia, and assumes participation is primarily intrin-

[1]http://mappings.dbpedia.org/server/statistics/*/
[2]http://mappings.dbpedia.org/index.php

sically motivated. While this self-organizing community approach works well, it is also acknowledged to be rather slow and challenging to manage; a substantial share of mappings is still missing [7]. Other forms of crowdsourcing could be used to improve on these aspects, most importantly paid microtasks.

Paid microtask crowdsourcing use services such as Amazon's Mechanical Turk[3] or CrowdFlower[4] to undertake work as a series of small tasks that together comprise a large unified project to which many people contribute. It is typically applied to repetitive activities that lend themselves to parallelization and do not require specific expertise beyond the most common human knowledge and cognitive skills: a paid microtasks project is broken down into many units that are self-contained and executed independently by different people for a fee in the range of several cents to a dollar. The approach has already been used in different areas related to Linked Data and semantic technologies [2,8,9,10,11,12]. In particular, [2,4,9] have shown that it can achieve reasonable accuracy in quality repair and entity typing tasks compared to expert crowds at a faster turnaround.

In this work we are considering the problem of systematically selecting the most specific category from a tree of hierarchically organized labels through employing microtasks. We analyze the main dimensions of the crowdsourcing exercise in depth in order to come up with suggestions for workflow design, which are grounded in existing literature in cognitive psychology, and to understand their implications in terms of precision and cost. To this end, we propose a workflow model consisting of a predictor, able to suggest a category candidate for a given entity as well as a crowd based error detection and correction algorithms. Using three alternative workflows, we compared the performance of an automatic predictor and two crowd based approaches: a naive predictor, were the whole ontology tree is traversed top down by the crowd and a microtask based free text predictor, able to make a decision based on human text input. To test our approach we run experiments on CrowdFlower using a dataset of 120 entities previously unclassified by the DBpedia community and compare the answers of the crowd with gold standard. Our experiments show that the microtask based free text approach achieved the highest precision at moderate cost compared to other workflows.

[3]`https://www.mturk.com/mturk/welcome`
[4]`http://www.crowdflower.com/`

However, each workflow has its merit and in this work we provide in-depth evaluation to support experts in selection decisions.

The rest of this paper is structured as follows: we start with an analysis of the state of the art in the areas of paid microtasks and machine learning in Section 2. In Section 3 we present our entity classification model based on machine and human input. Section 4 is dedicated to the design of our experiments where we measure the precision and costs of our model in three alternative workflows. Section 5 discusses the evaluation results and their implications for applying human computation and microtasks to similar scenarios. Finally, we conclude with a summary of our findings and plans for future work in Section 6.

2. Related work

In recent years, researchers have successfully applied human computation and crowdsourcing to a variety of scenarios which use Linked Data and semantic technologies. For the purpose of this paper we take a closer look at three classes of approaches, games with a purpose (GWAP), paid microtasks and automatic approaches involving machine learning. Both game with a purpose and paid microtasks are closely related to workflow and interface design, as well as to quality assurance, and have been applied to tasks that are similar to ours. Machine learning approaches are mostly applied for dynamic prediction during the crowdsourcing tasks.

2.1. Games with a purpose

The Semantic Web community has developed several games to lower the barrier of entry in conceptual modeling, ontology population, data interlinking, and semantic annotation [13,14,15,16]. These efforts offer great insights into the feasibility of a human computation approach to Semantic Web-related tasks. Siorpaes and Hepp were the first to bring GWAPs to ontology engineering through their OntoGame framework, including games such as OntoPronto [14], in which topics described in Wikipedia articles are classified according to PROTON, an upper-level ontology; SeaFish [16], which produces clusters of similar images; or SpotTheLink [17], which gamifies ontology alignment. Most games proposed in OntoGame are designed as multi-players games: users play against or with each other and achieve points only if their results coincide. Other researchers have taken up on the idea and developed their own games for other purposes. For instance, GuessWhat [13] takes seed concepts from Linked Data exploiting matching URIs in DBpedia, Freebase, and OpenCyc, and collects superclass and property information. It then generates class expressions and asks players to guess the described concepts, which in turn helps to build a new ontology. FreeAssociation and Categodzilla/Categorilla [18] apply the GWAP paradigm to semantic annotation scenarios, using different incentives to influence user behavior. There is clear empirical evidence that variations in task description impact data quality [18], motivating our search for alternative workflows for the DBpedia entity typing. In this work we base on the experiences described in the literature of the GWAP area for building a hybrid free text based predictor for the most specific category within an ontology.

2.2. *Paid microtasks*

The use of paid microtask platforms to add crowd and human computing capabilities to Semantic Web algorithms is equally accepted. In ontology engineering, the InPhO (Indiana Philosophy Ontology) project [10] used a hybrid approach to dynamically build a concept hierarchy by collecting user feedback on concept relationships from Mechanical Turk and automatically incorporating the feedback into the ontology. Mortensen et al. [11] did something similar to verify relationships in biomedical ontologies. Sarasua et al. [12] introduced and evaluated the Crowdmap model to convert ontology alignment task into microtasks. [19] reported on a microtask experiment that uses microtask workers as ontology evaluators. For semantic annotations, one of most cited works is [20], which elaborates on the quality of crowd-generated labels in natural language processing tasks, comparing it to expert-driven training and gold standard data. Closer to our task, ZenCrowd [9] explored the combination of probabilistic reasoning and crowdsourcing to improve the quality of entity linking. Finally, Dbpedia was subject to a series of microtask experiments in [2], which proposed a methodology to assess Linked Data quality and a workflow by which parts of this methodology could be executed as a combination of a contest targeting Linked Data experts [4] and CrowdFlower microtasks. In contrast, rather than to build and evaluate ontologies, in this work we are concentrating on developing and evaluating workflow models to annotate entities with classes from a given ontology.

2.3. *Machine Learning in Crowdsourcing*

As crowdsourcing is being widely used, it raises the challenge that how to handle noisy data and ensure high quality answers. Ipeirotis et al. [21] propose an algorithm to evaluate the quality of each worker, hence able to estimate the quality of the labels in Amazon MTurk platform. Difallah et al. [22] analysed the long-term historical data from MTurk and proposed features that can be used in a predictive model to help estimate the performance of tasks. Meanwhile, Dey [23] looked at how active learning, and additional contextual information could help improve the accuracy of data input from users. Whitehill et al. [24] present a probabilistic model to infer the correct label of each image by considering the level of expertise of the labeler as well as the difficulty of each image. Studies also show that it is possible to combine automatic prediction methods (Bayesian/Generative probabilistic models) with additional input from crowd to help improve the accuracy [25,26,27,28]. These aforementioned approach are mainly focusing on using machine learning to infer the quality of the answer. Other research specially using Machine Learning in large scale crowdsourcing [29,30] has been conducted in citizen science projects and classification tasks. In one case, machine learning algorithm is used to help the decision making of further task allocation. The later case on product classification which is similar to entity classification in our case with a given ontology, combines both, machine learning as well as manually created rules with the input from the crowd. All approaches described above, use machine learning for predicting purpose to navigate the task distribution, or to assess the quality of the results. In contrast, in this work we use machine learning as a support for the crowd to locate promising areas within the ontology with respect to occurrence of the most specific class label for a given entity.

This literature was used as foundational reference for the design of the microtask workflows introduced in this paper, including aspects such as instructions, user interfaces, task configuration, validation, and aggregation of results. The main novelty of our work lies in the systematic study of a set of alternative workflows. We believe such a analysis would be beneficial for a much wider array of Semantic Web and Linked Data scenarios in order to truly understand the complexity of the overall design space. This would also allow us to define more differentiated best practices for the use of crowdsourcing in real-world projects, which are likely to have other budgetary constraints than the one-off experiments usually found in the literature.

3. Approach

Entity classification problem considered in this paper, is a problem of selecting the most specific type from a given class hierarchy for a particular entity *e*. As a class hierarchy can contain thousands of classes, this task is hard to be solved manually for a few experts maintaining the data set, especially for large entity batches. Automatic and semi-automatic classification is a well-studied area with a variety of probabilistic and discriminative models, that can assist in this context. However, whilst there exist a number of possible machine based classification approaches, they all accept certain error margins of a few dozens of percents, such that manual correction of their output by an expert remains a heavy overhead. In this section we propose a model for a human computation based approach for reducing the amount of the corrections to be done by an expert.

Depending on a particular scenario, in some cases it is enough to detect the error, in other cases it is also required to correct it. In our model we propose to detect or correct the error by letting the crowd traverse the DBpedia ontology starting with the nodes proposed by the (semi-) automatic approach, until a correct node (the most specific category) is found. We break early in case we detect that the correct node does not exist in the ontology.

3.1. The Workflow Model

In this section we more formally describe (i) our human computation driven error detection and correction model for semi-automatic entity annotation using hierarchically structured set of labels. Additionally, we provide a (ii) cost model and (iii) describe the modeling of the microtasks at Crowdflower.

Definition 1 (DBpedia Ontology) *The DBpedia ontology $\boldsymbol{O}$ is a tree structure, where each node corresponds to an entity category and can contain a list of references to the children nodes each of those corresponds to a more specific sub-category of the parent entity type. The root node $\boldsymbol{ROOT}$ ("Thing") is the top most node in the ontology. .*

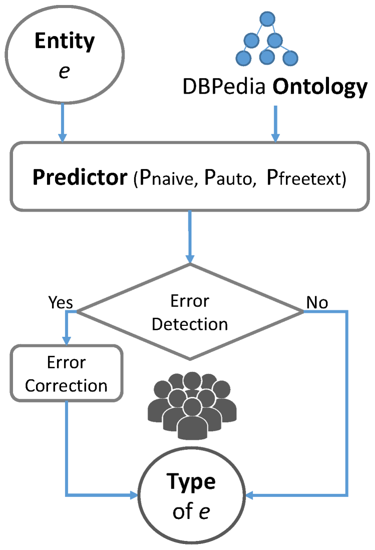


Fig. 1. Representation of a workflow with a semi-automatic prediction step, followed by the crowd based error detection and error correction steps.

Definition 2 (Candidate Nodes) *Given an entity $\boldsymbol{e}$ and ontology $\boldsymbol{O}$, particular node $\boldsymbol{N}$ is considered as the exact solution node $\boldsymbol{N_E}$* **iff** *$\boldsymbol{N}$ and not any of the children nodes of $\boldsymbol{N}$ is corresponding to the category of $\boldsymbol{e}$, i.e $\boldsymbol{N}$ represents the most specific category for $\boldsymbol{e}$ in $\boldsymbol{O}$. Any ancestor of $\boldsymbol{N_E}$ thereby is considered as a candidate node $\boldsymbol{N_C}$.*

Definition 3 (Predictor and Predicted Nodes) *Let a predictor $\boldsymbol{P}$ be a (semi-) automatic approach, able to select for given $\boldsymbol{e}$ a ranked set $\boldsymbol{S_{predicted}} = \{N_{p_1}, \ldots N_{p_n}\}$ of nodes from $\boldsymbol{O}$ as predicted candidates for $\boldsymbol{N_E}$.*

Our assumption is that the location of an $\boldsymbol{N_P}$ within $\boldsymbol{O}$ is close to $\boldsymbol{N_E}$ with high probability. In case no prediction can be made by $\boldsymbol{P}$ for a particular $\boldsymbol{e}$, we consider $\boldsymbol{S_{predicted}} = \{\boldsymbol{ROOT}\}$, as the $\boldsymbol{ROOT}$ is the closest to the optimal solution given the hierarchical structure definition in 1.

3.1.1. Human computation driven error detection and correction model

The error detection and correction process in this context is to traverse $\boldsymbol{O}$ starting from a set S consisting of top-ranked predicted nodes $\boldsymbol{N_P}$, until $\boldsymbol{N_E}$ is found and to break the process if no correct solution can be found. We model each traversal step as a microtask for a human assessor, where we ask, whether $\boldsymbol{N_C}$ exists in a list of options. The answer can be $\boldsymbol{true}$ (with indication of the corresponding node) or $\boldsymbol{false}$, in case no nodes from the list can be selected. Generally we assume that if a node is proven to be false, also all of its descendants are proven to be false.

Error detection algorithm: We employ a traversal algorithm 1 with a set S of top-scored node as a start. In case any of $\boldsymbol{N_P}$ from this set can be identified as $\boldsymbol{N_C}$, a check is done for each of its children nodes according to the definition 2.

Algorithm 1 Error detection

```
1: procedure CHECKSPECIFICTYPE( $e$, $S = \{N_{p1} \dots N_{pn}\}$)
2:     if $N_c \in S$ and $humanChoice() = N_c$ then
3:         if $\forall N_i \in children(N_c)$,
               $humanChoice() \neq N_i$ then
4:             return TRUE
5:     return FAIL;
```

Error correction algorithm: After the error is detected, it is possible to correct it using human assessors. Similarly to error detection, the microtask based correction algorithm 2 starts from the set S of candidate nodes. In case a $\boldsymbol{N_P}$ from S is identified as $\boldsymbol{N_C}$, its children nodes are traversed in a breadth first manner, until the node with the most specific type corresponding to $\boldsymbol{e}$ is found. Otherwise the algorithm continues with the parent node of $\boldsymbol{N_P}$. Every node on the way is touched only once. The algorithm stops when $\boldsymbol{N_C}$ is found without children corresponding to the type of $\boldsymbol{e}$. Note, in case no specific node can be found in $\boldsymbol{O}$, the algorithm will return $\boldsymbol{ROOT}$.

The costs of a particular algorithm are defined as the number of microtasks necessary to complete the algorithm run. Overall cost for entity annotation in our framework is constituted as the sum of prediction, error detection and error correction costs, more formally:

$$Cost(annotation) = Cost(prediction) + \\ Cost(detection) + Cost(correction)$$

3.2. Predictors

Currently all the predictors described in literature are automatic predictors, where given an entity a list of candidates along with the confidence of predictions is produced. In this work we employ a predictor P_{auto} as an example. We used DandelionAPI [5] which has an

[5] https://dandelion.eu/docs/api/

Algorithm 2 Error correction

```
1: procedure FINDSPECIFICTYPE( $e$, $S = \{N_{p1} \dots N_{pn}\}$)
2:     if $N_c \in S$ and $humanChoice() = N_c$ then
3:         $C = anyChild(e, N_c)$
4:         if $C = FAIL$ then
5:             return $N_c$
6:         else
7:             return $C$
8:     else
9:         $S_{parents} \leftarrow \{\}$
10:        for $N_i \in S$ do
11:            $S_{parents} \leftarrow S \cup parent(N_i)$
12:        return findSpecificType($e, S_{parents}$)
13: procedure ANYCHILD($e, N$)
14:     for $N_i \in children(N)$ do
15:         if $N_i = humanChoice()$ then
16:             $N_{child} = anyChild(e, N_i)$
17:             if $N_{child} = FAIL$ then
18:                 return $N_i$
19:             else
20:                 return $N_{child}$
21:     return FAIL;
```

add-on for Google Spreadsheet allowing entities uploaded in a spreadsheet to be analysed and typed with DBpedia types. The entity to be classified is the the text to be analysed, and the output we got is the types along with a confidence level for them. The types given from Dandelion API always include the parent types if a specific type is identified. For instance, if an entity's specific category is "Building", Dandelion gives Building, ArchitecturalStructure and Place in its output.

Additionally, we propose two human computation based prediction approaches. The first P_{naive} is a naive predictor approach that starts from the root of the ontology and traverses the tree by expanding the children in a bread first manner. This approach is very costly, but can be applied also for entities, where any of the other (semi-) automatic approaches fail. Our second human computation based prediction approach P_{free} allows for unconstrained input from the crowd, balancing microtask efficiency and annotation accuracy. The outcome depends on how many times the questions are asked and how the answers are aggregated. The main advantages of this approach are its simplicity and freedom of choice. Classification is one of the most common human cognitive skills and the crowd is not constrained by any unnecessary biases. In addition, restrictions on category input can sometimes increase the difficulty of the task [18] and hence impact the overall accuracy. The outcome depends on how many times the questions are asked and how the answers are aggregated. The more answers are collected, the more reliable the classification [31]. Top aggregated answers can be voted by the crowd in a second step to identify the most suitable candidate [32], or detect errors and correct the answers, as it is done in our case.

Considering the diversity of vocabulary of the crowd users, direct aggregation of answers is not effective. To solve this issue, we leverage the freetext input to automatically calculate the most close DBpedia types and present the top predicted categories to the crowd for the correction. We use the difflib[6] SequenceMatcher to compare the type name suggested by crowd with DBpedia classes to get a string match similarity score. For each entity, we calculate the similarity score of every input collected from the crowd with existing DBpedia type. Each DBpedia type then get a aggregated score as the indication of how close it is to the user proposed type. We then retrieve the top DBpedia types to form the list of output candidates to be used in error detection and error correction.

3.3. Microtask Design

Two of the most important factors that play into the success of a microtask approach to DBpedia entity typing are precision and costs. Precision refers to the ability of crowd contributors to submit answers that are useful, either in direct or aggregated form. Costs cover the effort a requester needs to invest to generate the microtasks and process crowd inputs, as well as the financial rewards for the accepted work. We distinguish between two types of microtasks based on their output: **T1** where the workers produce free text output; and **T2** where the worker can choose from a list of classes. In both cases, we generate descriptions of the input entities in form of labels or text summaries. Either way, the effort to generate the first type of tasks is comparatively lower, as there is no need to compute a list of potential candidates. However, this is compensated by overhead in sense making of the output and means to aggregate free text inputs into meaningful class suggestions, while in **T2** the answer domain is well-defined. Crowdsourcing literature recommends to use iterative tasks to deal with **T1** scenarios: in a first step the crowd is generating suggestions, while in the second it is asked to vote on the most promising ones [33,34,32].

T2 variant requires the strategy to generate the list of candidates. It can be achieved through automatic tools addressing a similar task (such as GATE,[7] NLTK,[8] ,Dandelion API,[9], Alchemy API,[10] and Open Calais[11] for entity annotation), however, they impose clear bounds on the accuracy of the crowd experiments, as not all input can be processed by the automatic tools (recall) and their output can be only as precise as the task input allows (precision). The choice of an appropriate threshold to accept or reject produced results is also subject of debate in related literature [35,36]. Another option is to provide the crowd all possible choices in a finite domain - in our case all classes of the DBpedia ontology. The challenge then is to find a meaningful way for the crowd to explore these choices.

3.4. Workflows

For the purpose of this study we considered three types of workflows, following the preliminary considerations presented so far. Each workflow incorporates a predictor, as well as error detection and correction steps.

- **(W_{free}) Freetext**: accept unconstrained input from the crowd to lable an entity. Iterate to identify the most promising suggestion. This workflow incorporates P_{free} predictor.
- **(W_{naive}) Naive**: ask the crowd to choose the DBpedia category by traversing from the root top-down until a specific category is selected. This is the workflow that employs P_{naive} predictor.
- **(W_{auto}) Automatic**: use an entity typing tool to generate candidates, then ask the crowd to choose from a shortlist. This workflow is based on the P_{auto} predictor.

In the remainder of this section we explain the workflows in more detail and their translation into microtasks.

[6]`https://docs.python.org/2/library/difflib.html`

[7]`https://gate.ac.uk/`
[8]`http://www.nltk.org/`
[9]`https://dandelion.eu/docs/api/`
[10]`http://www.alchemyapi.com/`
[11]`http://www.opencalais.com/`

3.4.1. *Option List Length and the Choice*

While classification is indeed one of our most common human skills, cognitive psychology established that people have troubles when too many choice are possible. This means both too many classes to consider [37,36] and too many relevant criteria to decide whether an item belongs to a class or not [38,39]. Miller's research suggests that in terms of the capacity limit for people to process information, 7 (plus or minus 2) options is a suitable benchmark [40]. For our workflows we hence would ideally use between 5 and 10 classes to choose from. This situation is given by the predictor P_{auto}, as the confidence of automatic entity typing algorithms decreases rapidly and only the top 10 are likely to be representative. However, in the P_{naive} condition we start from the DBpedia ontology, which has over 700 classes to choose from.[12] The problem persists even when browsing the ontology level by level, as some classes have many tens of subclasses (e.g., there are 21 different forms of organization, 50 child categories of person and 43 specific types of athlete). In our experiments therefore, we split the lists into subsets of 7 items or less to display per microtask.

The particular user choice from a list depends on her level of expertise and on the given situation, as the theory teaches. Rosch et al. proved in experiments that experts and newbies make very different classification decisions - people with little insight into a domain tend to feel comfortable with categories that are neither too abstract, not too specific, whereas experts are much more nuanced [41]. The same effect was observed by [42] or in games with a purpose [43]. Van Ahn et al. noted that ESP game players adapted their classification behavior to maximize returns. As the game was played in pairs, the best strategy for each annotator was to minimize the risk of suggesting a class label that their opponent would not and hence loosing points. As we are working with microtask platforms, we have to assume that the behavior of the crowd contributors will resemble newbies in the Rosch et al. experiments and casual gamers who interacted with GWAPs. Thus, we can not always expect from P_{naive} to identify the most specific class in the DBpedia ontology, which matches the input entity. Rosch et al. [41] define a so-called "basic level of abstraction" as the level at which the most distinguishable features to differentiate an item from another apply. This is the level non-experts tend to select most often.

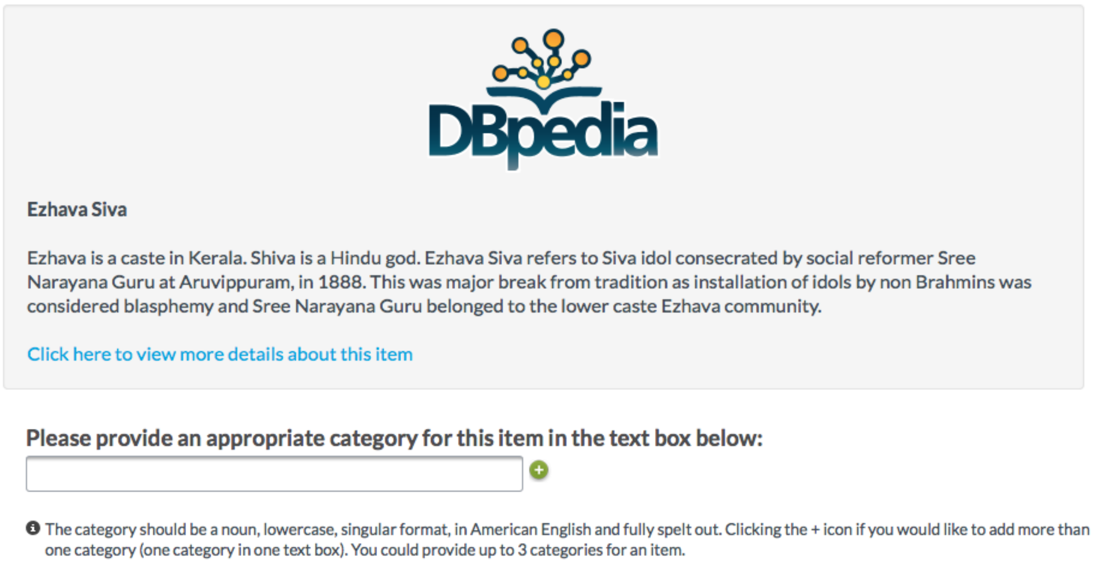
[12]`http://mappings.dbpedia.org/server/ontology/classes/(accessedonJan14,2016)`

3.5. *Workflow Implementation*

For each workflow we first queried the DBpedia endpoint via SPARQL to obtain the name, description, and link to Wikipedia of each entity. We created the gold standard data (see Section 4), set the required CrowdFlower parameters, launched the jobs, and monitored their progress. The Figures 2 and 3 depict the user interfaces for **T1** and **T2** types of tasks. For **T2** we always limited the number of options shown to a user in the list to maximum of 7 (6 class candidates and an $NoneOfAbove$ option). In case there are more candidates, we split into batches.

3.5.1. *CrowdFlower Parameters*

Task: Following the advises from the literature [2], we used 5 units/rows for each task/page for each of the three workflows. The worker completes a task by categorising five entities, including a control question.
Judgment: Snow et al. [20] claims that answers from an average of four non-experts could achieve a level of accuracy parallel to NLP experts on Mechanical Turk. We hence asked for 5 judgments per experimental data throughout our experiments. In W_{free}, we asked for 11 free text suggestions. This choice is in line with previous experiments from the literature [2,12].We also did empirical simulation with our previous experiments on entities that have been classified to find out the effective number of judegment to use. Figure 4 shows number of times the entity has been annotated and the corresponding accurate answers (based on 120 entities).
Payment: We paid 6 cents for each task consisting of 5 units. This setting took into account existing surveys [44], as well as related experiments which has similar complexity level. Task like reading an article and then asking the crowd to rate the article based on given criteria as well as providing a suggestion on the areas to be improved are paid 5 cents [45]. Tasks that have smaller granularity such as validating the given linked page display relevant image to the subject were paid 4 cents per 5 units [2]. In a similar vein, the complexity of our classification task is somewhere in between considering the time and knowledge it requires to complete the task.
Quality control: We created test questions for all jobs and required contributors to pass a specified minimum accuracy rate (we use 50%) before working on the units.


Ezhava Siva

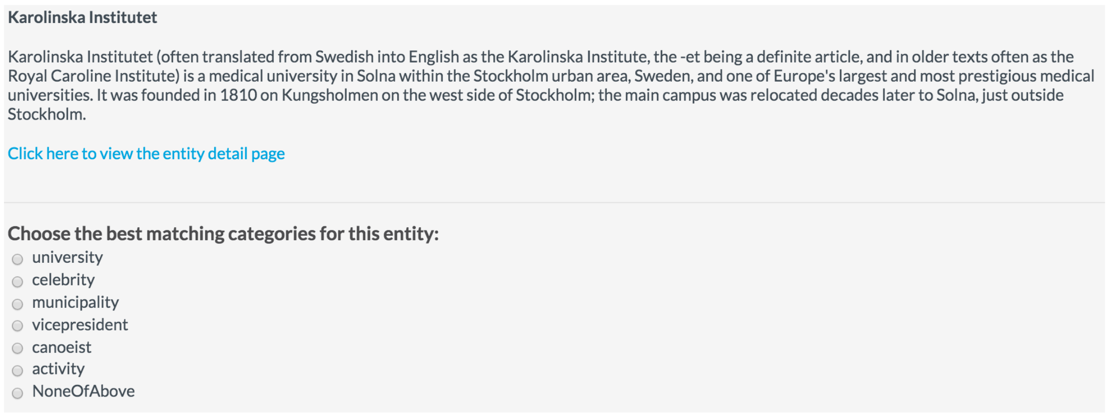
Ezhava is a caste in Kerala. Shiva is a Hindu god. Ezhava Siva refers to Siva idol consecrated by social reformer Sree Narayana Guru at Aruvippuram, in 1888. This was major break from tradition as installation of idols by non Brahmins was considered blasphemy and Sree Narayana Guru belonged to the lower caste Ezhava community.

Click here to view more details about this item

Please provide an appropriate category for this item in the text box below:

The category should be a noun, lowercase, singular format, in American English and fully spelt out. Clicking the + icon if you would like to add more than one category (one category in one text box). You could provide up to 3 categories for an item.

Fig. 2. The user interface for **T1** tasks based on free text input


Karolinska Institutet

Karolinska Institutet (often translated from Swedish into English as the Karolinska Institute, the -et being a definite article, and in older texts often as the Royal Caroline Institute) is a medical university in Solna within the Stockholm urban area, Sweden, and one of Europe's largest and most prestigious medical universities. It was founded in 1810 on Kungsholmen on the west side of Stockholm; the main campus was relocated decades later to Solna, just outside Stockholm.

Click here to view the entity detail page

Choose the best matching categories for this entity:

- university
- celebrity
- municipality
- vicepresident
- canoeist
- activity
- NoneOfAbove

Fig. 3. The user interface for **T2** tasks based on a list of candidate options

Contributors: CrowdFlower distinguishes between three levels of contributors based on their previous performance. The higher level of contributors required, the longer it takes to finish the task but might be with higher quality. In our experiment, we choose the default Level1 which allow all levels of contributors to participate in the classification task. We used this level for all three workflows.

Aggregation: For the **T2** tasks we used the default option (aggregation='agg')[13], as the task is to choose from a set of pre-defined options. For **T1**, we looked at the first three answers (aggregation='agg_3') based on 11 judgments. We also used aggregation='all' to collect the additional categories when the crowd felt the best match was not listed among the suggestions.

3.6. *Using the Crowdsourced Data*

The validated and aggregated results may be leveraged in several ways in the context of DBpedia. Free-text suggestions signal potential extensions of the DBpedia ontology (concepts and labels) and of the DBpedia Extraction Framework (mappings). Applying the shortlist workflow gives insights into the limitations of entity typing technology, while the way people interact

[13]`https://success.crowdflower.com/hc/en-us/articles/203527635-CML-Attribute-Aggregation`

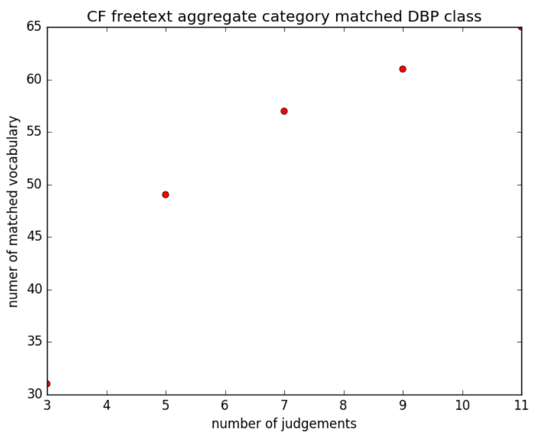


Fig. 4. Freetext: Judgement Number vs. Accuracy

with the naive workflow is interesting not only because it provides hints about the quality imbalances within the DBpedia ontology, but also for research on Semantic Web user interfaces and possible definition of new mapping rules for entity typing.

4. Evaluation

In this section, we will evaluate the performance of proposed predictors in terms of workflows depicted in Figure 1. Overall, we obtained three workflows, namely: W_{naive}, W_{auto} and W_{free}, respectively based on predictors P_{naive}, P_{auto} and P_{free}. We evaluate the accuracy of the predictors and compare the overall costs of different workflows.

4.1. Data

For our experiments we uniformly at random choose 120 entities which were not classified so far by the DBPedia. The authors of the paper annotated these entities manually to obtain a gold standard. The annotators worked independently and achieved an agreement of 0.72 measured using Cohen's kappa. According to one of the most commonly used interpretation by Landis and Koch(1977), kappa value in the range of 0.6-0.8 corresponds to a substantial agreement. Noted we are able to get 106 out of 120 entity typings agreed between two annotators. To achieve consensus, the annotators then collaboratively defined a set of rules to categorize the entities whose classes did not match and involved a third annotator for majority voting calculation. For example, an entity such as 'List of Buffy the Vampire Slayer novels' was eventually classified as List instead of Novel, while the 'Haute Aboujagane, New Brunswick', which describes a specific community, was defined as Community instead of GovernmentAdministrativeRegion.

Table 1 provides the overview on the composition of the resulting gold standard corpus with respect to general categories. "Place" and "Organisation" were the most present categories with respectively 20 and 16 entities in the dataset, corresponding to one of their child categories. For 18 entities no appropriate category in the ontology could be found by the experts, those were labeled as "owl:Thing".

4.2. Experiments

For each workflow we assessed their properties in our experiments with respect to quality, effectivity and costs. The accuracy of the predictor matching our gold standard will provide information about native predictor effectivity for real applications. The accuracy of the human based error correction with respect to our gold standard will provide insights into what quality can be to expected from the crowd in general.

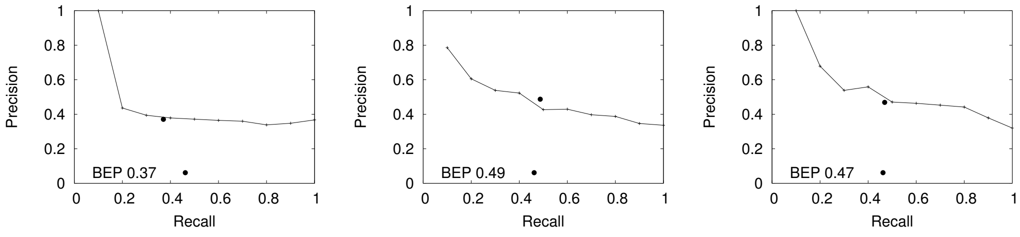
4.2.1. Prediction Quality

In our quality concerned experiments, we employ precision measure which measures the portion of correctly classified entities among all entities classified by the predictor. Our quality measures are the precision-recall curves as well as the precision-recall break-even points for these curves. The precision-recall graph provides insights in how much precision it is possible to gain in case only a certain amount of recall is required. Typically, the head of the distribution achieves better precision due to the fact that the values correspond to the higher confidence of the predictor. The break-even point (BEP) is the precision/recall value at the point where precision equals recall, which is equal to the F1 measure, the harmonic mean of precision and recall, in that case. The results of the experiments for predictors described in Section 3.2 are shown in Figure 5. The main observations are:

The precision of the human computation based methods was generally higher when compared to the automatic method. Also the automatic method did not produce any results for around 80% of the entities, whilst human computation based predictor provided suggestions for all entities with certain quality. P_{naive} was showing the best precision over the test set of BEP 0.49, followed by P_{free} with BEP 0.47. Surprisingly,

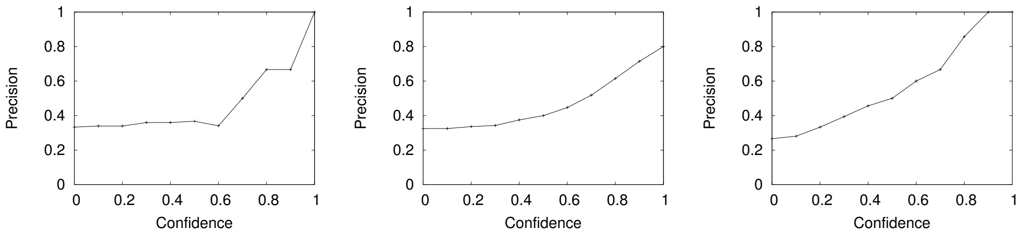
#Categories	20	18	16	10	9	7	5	4	2	1
Categories	Place	owl:Thing	Organisation	Work Agent	TopicalConcept	Group	List	EthnicGroup	Company PersonFunction Device UnitOfWork Food Species ChemicalSubstance	Name Activity MeanOfTransportation Medicine EducationalInstitution Event Language

Table 1
Overview of the **E1** Corpus



(a) P_{auto} (b) P_{naive} (c) P_{free}

Fig. 5. Precision-recall curves for the output of the different predictors with respect to our gold standard.



(a) P_{auto} (b) P_{naive} (c) P_{free}

Fig. 6. Precision at confidence level curves for the output of the different predictors with respect to our gold standard.

the precision of P_{naive} was lower than expected for a completely human computation based solution and may indicate that the classification task requires high expertise and can not rely on crowd alone. For about 10% of recall all methods provide very good results, especially the precision for P_{auto} and P_{free} was very high, making further steps in error detection and correction possibly unnecessary.

We standardized the confidence level of each predictor to the range [0.0-1.0] and plotted the output quality at different confidence levels in terms of precision in Figure 6. As expected from previous results, high confidence levels above 0.9 for P_{auto} and P_{free} indicate high quality results. For confidence levels below, the output can not be trusted and need error correction. To improve the prediction, quality research need to be done to develop better predictors, or improve the quality existing ones.

	W_{auto}	W_{naive}	W_{free}
Prediction costs	0	4.1	1
Error detection costs	2.9	0	1.6
Error correction costs	3.2	0	1.83
Sum	6.1	4.1	4.43

Table 2
Cost overview for different workflows

4.2.1.1 Prediction costs

In comparison to P_{auto}, the predictors P_{free} and P_{naive} required human input and therefore additional costs. Whilst in P_{free} the user had to execute exactly one free text task per entity, to obtain the results using P_{naive}, the crowd worker had to complete 4.1 tasks on average.

4.2.2. Error Detection Quality and Costs

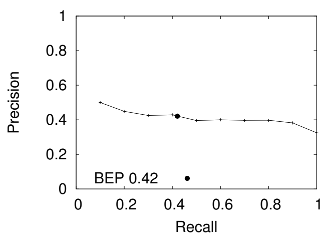
Having received the output of a predictor, we tested whether the crowd was able to identify prediction errors and empirically determined the costs for this detection in terms of the number of questions needed to be answered on average. The graphs provide insights into the quality of the predictor outputs with respect to crowd decisions and can provide decision support to include or exclude the error detection step in real applications. We do not test error detection for P_{naive} as its output already based on crowd and can not be expected to improve with the error detection or correction step. As defined in our crowdsourcing model, at each task we show 7 candidate options to the user. On average, error detection took 2.9 detection steps with W_{auto} and 1.6 steps with W_{free} as depicted in Table 2.

4.2.3. Error Correction Quality and Costs

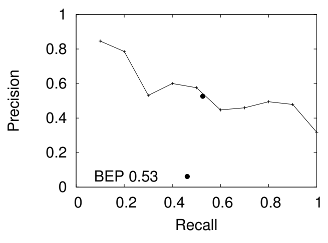
In the last step we apply our algorithm to correct the errors produced by the predictors. Similar to the previous section, we measured the costs for the correction as the number of of questions needed to be answered on average. The Table 2 provides an overview over the costs. W_{auto} required on average 3.2 steps to correct the prediction due to the fact that the predictor did not produce any results for the most entities and the whole tree had to be traversed to find the answer in such cases. 1.83 steps were required on average for W_{free}, indicating that in general the predictor pointed the user to the right area within the ontology. In summary, as depicted in Table 2, W_{auto} appeared the most costly with 6.1 steps on average and the other two workflows shown comparable results.

4.2.4. The Quality of Human Output

Finally, we will measure the quality of the workflows as a whole to estimate the effort to be invested by experts for post-processing. The Figure 7 shows the precision recall curves of the human based result correction for W_{auto} and W_{free}. We observe that in both workflows the result improved when compared with the prediction step alone. Our proposed workflow W_{free} reached a BEP of 0.53 - the highest result among all experiments.



(a) W_{auto}



(b) W_{free}

Fig. 7. The overall output quality of the workflows.

4.2.4.1 Crowdsourcing Tasks

We decided to limit the number of top-options shown to the user to 7 as recommended in the literature. Longer lists may contain the correct result with higher probability, however would also require more interaction and effort in complexity for a crowd worker. To show the possible influence of the options number on the prediction quality, we plot the correspondence in Figure 8 where we vary the list size on the logarithmic scale from top 1 to the maximum of 740 possible categories and measure the predictor BEPs' on basis of the gold standards. As we can observe, the precision improves only slightly with the growing list size, indicating no meaningful advantage of lists with more than 7-10 options.

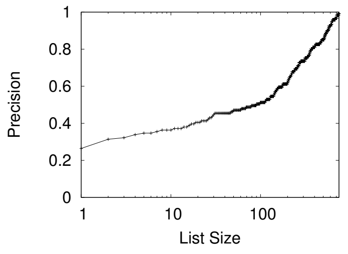


Fig. 8. Precision at different Options List Size with the W_{free} Predictor

5. Discussion and Lessons Learnt

5.1. Are unclassified entities enclassifiable

As noted earlier, there were significant differences in the performance scores achieved in the experiments using different workflows. Especially notable is that existing NLP tool could only classify 45% of the randomly selected entities (54 out of 120) with a precision of only 0.37. Human-based approach can achieve relatively higher 0.49 to 0.47 precision, but still a large portion of the entity are not classified to the correct specific type. To some extent, this indicates that untyped entities have certain characteristics which makes them difficult to be classified. Firstly, we observed that their types are quite diverse and not within the most popular classes[14]. For instance, "Standard", "SystemOfLaw", 'or "TopicalConcept" are not categories non-expert could easily distinguish. Secondly, the imbalanced structure of DBpedia ontology also makes the classification for untyped entities whose boundary between subtle categories are not well defined. For example, "Hochosterwitz Castle" is a "Castle" which would be the most specific category for this entity, however, Castle is a child category of Building which is a child type of "ArchitecturalStructure" which has many child types such as "Arena", "Venue" and "Pyramid", leading the user to choose none of the children of "ArchitecturalStructure" as they did not see any fits. Similarly, among all 18 entities that are instances of "Place", only 5 of them are correctly classified to the most specific category because of the unclear and overdefined sub-categories. "Place" and "Area" are both immediate first level type under "owl:Thing", which create a lot confusion in the first place as we observed from the crowd contributed classification. Also categories such as "Region", "Locality" and "Settlement" are difficult to be differentiated.

Lastly, ambiguous entities unsurprisingly caused disagreement [46]. This was the case with "List" and specific types such as the "1993 in Film",[15] which is an List (not a film), and the "1976 NBA Finals",[16] which is rather a Tournament (child of "Event", "SocietalEvent" and "SportsEvent", but not a "List"). In general, entities like these contain a context which sometimes make the entity itself ambiguous. In the similar way, "Provinces of the Dominican Republic"[17] is a list (not a place) while "Luxembourg at the Olympics" is a sports team. In other case, an entity with context is just difficult to fit in any existing DBpedia types. For instance, "Higher education in Hong Kong" and "Petroleum industry in Nigeria".

5.2. The outputs are only as good as the inputs

Taking naive workflow where we present maximum of 7 types (including a "NoneOfAbove" option) in one step as an example, the aggregated outcome shows 33 entities are categorised as "other" after traversing the DBpedia classes tree top-down from "owl:Thing", with none of the DBpedia categories being chosen. This also contributes to the ongoing debate in the crowdsourcing community regarding the use of miscellaneous categories [46,47]. In our case, using this option elicited a fair amount of information, even if it were to be used just to identify a problematic case. [47] discuss the use of instructions as a means to help people complete an entity typing task for microblogs. In our case, however, we believe that performance enhancements would be best achieved by studying the nature of unclassified entities in more depth and looking for alternative workflows that do not involve automatic tools in cases which we assume they will not be able to solve. A low hanging fruit is the case of containers, which can be identified easily. For the other cases, one possible way to move forward would be to compile a list of entity types in the DBpedia ontology, which are notoriously difficult, and ask the crowd to comment upon that shortlist instead of the one more or less 'guessed' by a computer program. Another option would be to look at workflows that involve different types of crowds. However, it is worth mentioning that for the 120 randomly chosen untyped entities from DBpedia, 18 of them don't fit in any DBpedia types based on our gold standard which indicate there is a need to enhance the ontology itself.

5.3. Popular classes are not enough

As noted earlier, entities which do not lend themselves easily to any form of automatic classification seem to be difficult to handle by humans as well. This

[14]`http://wiki.dbpedia.org/services-resources/datasets/data-set-39/data-set-statistics`

[15]`https://en.wikipedia.org/wiki/1993_in_film`

[16]`https://en.wikipedia.org/wiki/1976_NBA_Finals`

[17]`https://en.wikipedia.org/wiki/Provinces_of_the_Dominican_Republic`

is worrying, especially if we recall that this is precisely what people would expect crowd computing to excel at, enhancing the results of technology. However, we should also consider that a substantial share of microtask crowdsourcing applications address slightly different scenarios: (i) the crowd is either asked to perform the task on their own, in the absence of any algorithmically generated suggestions; or (ii) it is asked to create training data or to validate the results of an algorithm, under the assumption that those results will be meaningful to a large extent. The situation we are dealing with here is fundamentally new because the machine part of the process does not work very well and distorts the wisdom of the crowds. These effects did not occur when we use free annotations. Entity such as "Brunswick County North Carolina"[18] is a obviously a "County" and a child type of "Place". Freetext approach actually propose this category, although that category does not exist in DBpedia yet.This result is consistent with [18].

It became evident that in case the predicted categories are labeled in domain-specific or expert terminology, people tend not to select them. While in the unbound condition they are comfortable using differentiated categories, the vocabulary has to remain accessible. For example, Animal (sub-category of Eukaryote) is used more than Eukaryote. In all three workflows, if the more general category is not listed, participants were inclined towards the more specialized option rather than higher-level themes such as Person, Place, and Location. This could be observed best in the freetext workflow. Such aspects could inform recent discussions in the DBpedia community towards a revision of the core ontology.

5.4. *Spam prevention*

It has been observed that crowdsourcing microtasks sometimes generate noisy data which either is submitted deliberately from lazy workers or from the crowd whose knowledge of the task area is not sufficiently enough to meet certain accuracy criteria. Test question is a good way to help minimize the problems caused by both cases such that only the honest worker with basic understanding are involved in the tasks. Although the test question approach require about 10% additional judgments to be collected, it does give good inputs in which the definite spam are rare and negligible.

[18]`https://en.wikipedia.org/wiki/Brunswick_County,_North_Carolina`

6. Conclusion and Future Work

In this paper we are the first to propose a crowdsourcing based error detection and correction workflows for (semi-) automatic entity typing in DBpedia with selection of the most specific category. Though our framework is employing DBpedia ontology, in principle it can also be applied to other classification problems where the labels are structured as a tree. In our second contribution we propose a new microtask based free text approach for the prediction of entity type. This predictor provide good results even for entities where automatic machine learning based approach fails and naive full ontology scans are necessary. We empirically evaluated the quality of the proposed predictors as well as the crowd performance and costs for each of the workflows using 120 DBpedia entities chosen uniformly at random from the set of entities, not yet annotated by the DBPedia community.

While upper levels of the DBpedia ontology seem to match well the basic level of abstraction coined by Rosch and colleagues more than 30 years ago [41], contributors used more specific categories as well, especially when not being constrained in their choices to the (often un-balanced) DBpedia ontology. The experiments also call for more research into what is different about those entities which make the hard cases and in our discussion we gave some suggestions for improvement.

In our future work we plan to investigate how the quality of semi-automatic predictors can be further improved to reduce the costs for correction to a minimum. More work can be done in design of the microtasks especially in terms of the option choice displayed to the user. Finally, there is a lot to be done in terms of support for microtask crowdsourcing projects. The effort invested in our experiments could be greatly reduced if existing platforms would offer more flexible and richer services for quality assurance and aggregation (e.g., using different similarity functions).

References

[1] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 2014.

[2] Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Sören Auer, and Jens Lehmann. Crowdsourcing linked data quality assessment. *The Semantic Web–ISWC 2013*, pages 260–276, 2013.

[3] P Kreis. Design of a quality assessment framework for the dbpedia knowledge base. *Master's thesis, Freie Universität Berlin*, 2011.

[4] Amrapali Zaveri, Dimitris Kontokostas, Mohamed A Sherif, Lorenz Bühmann, Mohamed Morsey, Sören Auer, and Jens Lehmann. User-driven quality evaluation of dbpedia. In *Proceedings of the 9th International Conference on Semantic Systems*, pages 97–104. ACM, 2013.

[5] Christian Bizer. Dbpedia version 2014 released. http://blog.dbpedia.org/2014/09/09/dbpedia-version-2014-released/, 2014. [Online; Accessed: 8 Dec 2014].

[6] Katharina Siorpaes and Elena Simperl. Human intelligence in the process of semantic content creation. *World Wide Web*, 13 (1-2):33–59, 2010.

[7] Dimitris Kontokostas. Making sense out of the wikipedia categories (gsoc2013). blog.dbpedia.org, 2013. URL `http://blog.dbpedia.org/2013/11/29/making-sense-out-of-the-wikipedia-categories-gsoc2013/`. [Online; Accessed: 29 Dec 2014].

[8] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 313–322. ACM, 2010.

[9] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st international conference on World Wide Web*, pages 469–478. ACM, 2012.

[10] Kai Eckert, Mathias Niepert, Christof Niemann, Cameron Buckner, Colin Allen, and Heiner Stuckenschmidt. Crowdsourcing the assembly of concept hierarchies. In *Proceedings of the 10th annual joint conference on Digital libraries*, pages 139–148. ACM, 2010.

[11] Jonathan M Mortensen, Mark A Musen, and Natalya F Noy. Crowdsourcing the verification of relationships in biomedical ontologies. In *AMIA Annual Symposium Proceedings*, volume 2013, page 1020. American Medical Informatics Association, 2013.

[12] Cristina Sarasua, Elena Simperl, and Natalya F Noy. Crowdmap: Crowdsourcing ontology alignment with microtasks. In *The Semantic Web–ISWC 2012*, pages 525–541. Springer, 2012.

[13] Thomas Markotschi and Johanna Völker. Guesswhat?!–human intelligence for mining linked data. 2010.

[14] SemanticGames. Games for ontology building. http://semanticgames.org/category/games-for-ontology-building, 2012. URL `http://semanticgames.org/category/games-for-ontology-building/`.

[15] Katharina Siorpaes and Martin Hepp. Ontogame: Towards overcoming the incentive bottleneck in ontology building. In *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops*, pages 1222–1232. Springer Berlin Heidelberg, 2007.

[16] Stefan Thaler, Katharina Siorpaes, David Mear, Elena Simperl, and Carl Goodman. Seafish: a game for collaborative and visual image annotation and interlinking. In *The Semanic Web: Research and Applications*, pages 466–470. Springer, 2011.

[17] Stefan Thaler, Elena Paslaru Bontas Simperl, and Katharina Siorpaes. Spotthelink: A game for ontology alignment. *Wissensmanagement*, 182:246–253, 2011.

[18] David Vickrey, Aaron Bronzan, William Choi, Aman Kumar, Jason Turner-Maier, Arthur Wang, and Daphne Koller. Online word games for semantic data collection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 533–542. Association for Computational Linguistics, 2008.

[19] Natalya Fridman Noy, Jonathan Mortensen, Mark A. Musen, and Paul R. Alexander. Mechanical turk as an ontology engineer?: using microtasks as a component of an ontology-engineering workflow. In *Web Science 2013*, pages 262–271, 2013.

[20] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics, 2008.

[21] Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. Quality management on Amazon Mechanical Turk. *Proceedings of the ACM SIGKDD Workshop on Human Computation - HCOMP '10*, page 64, 2010. ISSN 145030222X. . URL `http://www.scopus.com/inward/record.url?eid=2-s2.0-77956245055&partnerID=tZOtx3y1`.

[22] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G. Ipeirotis, and Philippe Cudré-Mauroux. The Dynamics of Micro-Task Crowdsourcing: The Case of Amazon MTurk. pages 238–247, may 2015. URL `http://dl.acm.org/citation.cfm?id=2736277.2741685`.

[23] Stephanie L. Rosenthal and Anind K. Dey. Towards maximizing the accuracy of human-labeled sensor data. *Proceedings of the 15th international conference on Intelligent user interfaces - IUI '10*, page 259, 2010. . URL `http://dl.acm.org/citation.cfm?doid=1719970.1720006`.

[24] Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier Movellan. Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. *Advances in Neural Information Processing Systems*, 22 (1):1–9, 2009.

[25] Babak Loni, Jonathon Hare, Mihai Georgescu, Michael Riegler, Xiaofei Zhu, Mohamed Morchid, Richard Dufour, and Martha Larson. Getting by with a little help from the crowd: Practical approaches to social image labeling. *Proceedings of the 2014 International ACM Workshop on Crowdsourcing for Multimedia*, pages 69–74, 2014. .

[26] Jonathon S Hare, Maribel Acosta, Anna Weston, Elena Simperl, Sina Samangooei, David Dupplaw, and Paul H Lewis. An Investigation of Techniques that Aim to Improve the Quality of Labels provided by the Crowd. In Martha A Larson, Xavier Anguera, Timo Reuter, Gareth J F Jones, Bogdan Ionescu, Markus Schedl, Tomas Piatrik, Claudia Hauff, and Mohammad Soleymani, editors, *Proceedings of the MediaEval 2013 Multimedia Benchmark Workshop, Barcelona, Spain, October 18-19, 2013.*, volume 1043 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2013. URL `http://ceur-ws.org/Vol-1043/mediaeval2013_submission_44.pdf`.

[27] Edwin Simpson, Stephen Roberts, Ioannis Psorakis, and Arfon Smith. Dynamic bayesian combination of multiple imperfect classifiers. *Studies in Computational Intelligence*, 474:1–35, 2013. ISSN 1860949X. .

[28] Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceeding KDD '08 Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622, Las Vegas, Nevada, USA, 2008.

[29] Ece Kamar, Severin Hacker, and Eric Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 467–474. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[30] Chong Sun, Narasimhan Rampalli, Frank Yang, and AnHai Doan. Chimera: Large-scale classification using machine learning, rules, and crowdsourcing. *Proceedings of the VLDB Endowment*, 7(13):1529–1540, 2014. ISSN 2150-8097.

[31] Chris J Lintott, Kevin Schawinski, Anže Slosar, Kate Land, Steven Bamford, Daniel Thomas, M Jordan Raddick, Robert C Nichol, Alex Szalay, Dan Andreescu, et al. Galaxy zoo: morphologies derived from visual inspection of galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 389(3):1179–1189, 2008.

[32] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. Turkit: human computation algorithms on mechanical turk. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 57–66. ACM, 2010.

[33] Andrew W Brown and David B Allison. Using crowdsourcing to evaluate published scientific literature: methods and example. *PloS one*, 9(7):e100647, 2014.

[34] Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. Cascade: Crowdsourcing taxonomy creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1999–2008. ACM, 2013.

[35] Benjamin Scheibehenne, Rainer Greifeneder, and Peter M Todd. What moderates the too-much-choice effect? *Psychology & Marketing*, 26(3):229–253, 2009.

[36] Barry Schwartz. The paradox of choice. Ecco, 2004.

[37] Sheena S Iyengar and Mark R Lepper. When choice is demotivating: Can one desire too much of a good thing? *Journal of personality and social psychology*, 79(6):995, 2000.

[38] Leola A Alfonso-Reese, F Gregory Ashby, and David H Brainard. What makes a categorization task difficult? *Perception & Psychophysics*, 64(4):570–583, 2002.

[39] Jianping Hua, Zixiang Xiong, James Lowey, Edward Suh, and Edward R Dougherty. Optimal number of features as a function of sample size for various classification rules. *Bioinformatics*, 21(8):1509–1515, 2005.

[40] George A Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

[41] Eleanor Rosch, Carolyn B Mervis, Wayne D Gray, David M Johnson, and Penny Boyes-Braem. Basic objects in natural categories. *Cognitive psychology*, 8(3):382–439, 1976.

[42] James W Tanaka and Marjorie Taylor. Object categories and expertise: Is the basic level in the eye of the beholder? *Cognitive psychology*, 23(3):457–482, 1991.

[43] Luis Von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004.

[44] Panagiotis G Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21, 2010.

[45] Aniket Kittur, Ed H Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456. ACM, 2008.

[46] Lora Aroyo and Chris Welty. Crowd truth: Harnessing disagreement in crowdsourcing a relation extraction gold standard. *Web Science'13*, 2013.

[47] O. Feyisetan, E. Simperl, M. Luczak-Roesch, R. Tinati, and N.Shadbolt. Towards hybrid ner: a study of content and crowdsourcing-related performance factors. In *Proceedings of the ESWC2015 (to appear)*, 2015.