A Platform for Commonsense Knowledge Acquisition Using Crowdsourcing

Christos T. Rodosthenous[1], Loizos Michael[1,2]
[1]Open University of Cyprus
[2]Research Center on Interactive Media, Smart Systems, and Emerging Technologies
P.O Box 12794, 2252, Nicosia, Cyprus
{christos.rodosthenous, loizos}@ouc.ac.cy

Abstract

In this article, we present our work on developing and using a crowdsourcing platform for acquiring commonsense knowledge aiming to create machines that are able to understand stories. More specifically, we present a platform that has been used in the development of a crowdsourcing application and two Games With A Purpose. The platform's specifications and features are presented along with examples of applying them in developing the aforementioned applications. The article concludes with pointers on how the crowdsourcing platform can be utilized for language learning, referencing relevant work on developing a prototype application for a vocabulary trainer.

Keywords: Games With A Purpose, Crowdsourcing, cloze tests, commonsense knowledge

1. Introduction

Human computation (Law and von Ahn, 2011) or crowdsourcing (von Ahn and Dabbish, 2008) is applied in cases where machines are not able to perform as good as humans can. In this paper, we focus on our work for developing a platform which utilizes crowdsourcing for acquiring knowledge about our world, i.e, commonsense knowledge. This platform was used to develop crowdsourcing applications, including Games With A Purpose (GWAPs) for acquiring commonsense knowledge suitable for understanding stories. More specifically, we present how the various platform features were used for the creation of two GWAPs: "Knowledge Coder" (Rodosthenous and Michael, 2014) and "Robot Trainer" (Rodosthenous and Michael, 2016) and a crowdsourcing application for acquiring knowledge that can be used in solving cloze tests, i.e., an exercise where a word from a passage or a sentence is removed and readers are asked to fill the gap.

The two games were designed to help the acquisition of commonsense knowledge in the form of rules. The first game implements a four-step methodology, i.e, acquiring, encoding, generalizing knowledge and verifying its applicability in other domains than the one used to generate it. The second game uses a hybrid methodology, where both human players and an automated reasoning system, based on the STory comprehension through ARgumentation (STAR) system (Diakidoy et al., 2015), are combined to identify and verify the contributed knowledge. Knowledge gathered is tested on answering questions on new unseen stories using the STAR system. Both games use a number of ready-made gamification elements from the platform to increase player contribution and interest to the task. Furthermore, the crowdsourcing platform's back-end interface was employed for real-time monitoring of the acquisition process and presentation of metrics and statistics in an intuitive dashboard.

For the crowdsourcing application, a three-step methodology was used, where contributors first find the missing word in a story, then they identify the words that lead to selecting the missing word and finally they verify the applicability of the contributed knowledge on filling a gap in a story where similar words are present. The process is repeated using a story which contains the previously identified words but with the missing word not explicitly present in the text. This application can also find use in language learning, since generated cloze tests can be delivered to language learners, while crowdsourcing the answers.

In the following sections, we present the developed crowdsourcing platform and its features, along with examples of how the platform was used in real scenarios for acquiring commonsense knowledge. In the penultimate section, we present related work in using crowdsourcing and discuss the differences with our approach for acquiring commonsense knowledge. In the final section, we give an overview of our work, provide insights on future directions and present a relevant extension of the crowdsourcing application in developing a vocabulary trainer for language learning.

2. The Crowdsourcing Platform

Following our vision for acquiring commonsense knowledge using crowdsourcing, we designed a platform which offers features and services that can be used to facilitate commonsense knowledge gathering from a number of paradigms, such as games, crowdsourcing tasks and mini applications. Most of the platform's specifications are applied in the majority of crowdsourcing platforms and applications and some of them are specific for the task of acquiring commonsense knowledge.

2.1. Platform Specifications

For developing the platform, we considered the following key design options: 1. the selection of a suitable technology for delivering task-based applications and GWAPs, 2. the handling of contributors' profiles, and 3. the representation of knowledge in a structured form that can be reused and verified. The platform should also allow monitoring of the acquisition process both in terms of contributors and acquired knowledge.

Furthermore, the platform should be able to offer a number of design elements needed in games and educational applications. These include but are not limited to: 1. leader boards, 2. contributors' ranking, 3. medals and awards, 4. progress-bars, 5. live feedback with notifications (both synchronous and asynchronous) for the events, and other gamification elements needed to provide the user with a pleasant experience while contributing.

On the back-end, the platform should be able to provide tools for designing a crowdsourcing application and managing contributors. These tools should provide developers the ability to easily change parameters of the application, e.g., number of raters for acquired knowledge to be valid, dynamic loading and changing of datasets (testing and validation) and export statistics on the system usage.

We chose to develop a web-based system using the Joomla[1] content management system (CMS) framework. The specific CMS inherently covers a lot of the aforementioned features in its core and it has a plethora of extensions for users to install, such as a community building component for creating multi-user sites with blogs, forums and social network connectivity. Additionally, the CMS provides a very powerful component development engine, for developers to deploy additional elements that can be reused in multi-domain applications.

There are many cases where crowdsourcing applications require functionality from other systems or knowledge bases, e.g., automated reasoning engines, datasets and natural language processing systems. For the crowdsourcing platform we constructed an Application Programming Interface (API) to the Web-STAR system (Rodosthenous and Michael, 2018) for story understanding related processing and we offer a direct integration to the Stanford CoreNLP (Manning et al., 2014) system. It is also able to retrieve and process factual knowledge, from ConceptNet (Speer et al., 2016), YAGO (Suchanek et al., 2007) and WordNet (Fellbaum, 2010). Developers can integrate other SPARQL-based (Quilitz and Leser, 2008) knowledge bases since the methodology used is generic.

The crowdsourcing platform offers a number of features for promoting the application to groups of users, either in social media or user forums. Contributors can share their contribution status/points/awards to social media groups. This tactic can increase user retention to the application. Moreover, developers can enable the "invitations" functionality, where contributors gain extra points when they invite other people to contribute.

2.2. Steps for Designing a Crowdsourcing Application Using the Platform

In this section, we showcase the steps needed for a developer to design and deploy a crowdsourcing application. These steps are also depicted in Figure 1. First, a template must be selected to match the application domain. There are a number of templates available to match a number of crowdsourcing paradigms (e.g., GWAPs, language learning applications) which can be customized according to the specific needs of the task.

[1]`https://www.joomla.org/`

Developers need to prepare the main functionality of their system by coding it in PHP, or any other language and encapsulate its executable in the platform and deliver the result using HTML, CSS and JavaScript. During this process, they need to prepare a list of parameters that can be used in the experiments and code it in XML format. These parameters can be incorporated in the code and control how various elements are displayed (e.g., display/hide web tour and guidance, choose what knowledge is presented for verification, etc.).

The next steps involve the selection of knowledge acquisition tasks. Developers can select among acquisition, verification and knowledge preference identification tasks and map the methodology steps to application screens or game missions (depending on the chosen paradigm). The knowledge preference selection task involves the ability of a human contributor to choose pieces of knowledge that are used in a given situation and discard the ones that are not. For example, when reading a story about birds, readers can infer that birds can fly. From a similar story, where it is explicitly mentioned that birds are penguins, readers can infer that penguins cannot fly.

For each task, a data stream is required. The data stream can be anything from text inserted directly from contributors, i.e, a dedicated task in the application, a pre-selected dataset such as Triangle-COPA (Maslan et al., 2015) or ROCStories (Mostafazadeh et al., 2016), or the outcome of another task.

Developers are free to design and code the logic behind each task as they see fit to achieve their goals. The platform has a number of pre-defined functions for storing commonsense knowledge in the form of rules or facts, both in natural language and in a logic-based format, e.g., `hug(X,Y)` implies `like(X,Y)` where `X` and `Y` are arguments and intuitively means if a person `X` hugs a person `Y` then person `X` likes person `Y`.

Moreover, the platform incorporates a number of visualization libraries (e.g., d3.js[2], Cytoscape.js[3], chart.js[4]) to provide live feedback to the contributor.

For each application, developers need to choose how contributed knowledge is selected and what are the criteria for storing this knowledge in the accepted knowledge pool. Developers can choose among a number of strategies or a combination of them, such as selecting knowledge that was contributed by at least n number of persons, knowledge that is simple (e.g., rules with at most n predicates in their body), knowledge that is evaluated/rated by at least n raters and knowledge that is evaluated by an automatic reasoning engine. Depending on the type of application, developers also need to choose a marking scheme that fits the logic behind the application and reward contributors, e.g., points and medals for games.

When the design of the various tasks is completed, the developer needs to choose how contributors will have access to the platform (e.g., anonymously, through registration or social networks) and what details need to be filled in their profiles.

[2]`https://d3js.org/`
[3]`http://js.cytoscape.org/`
[4]`https://www.chartjs.org/`

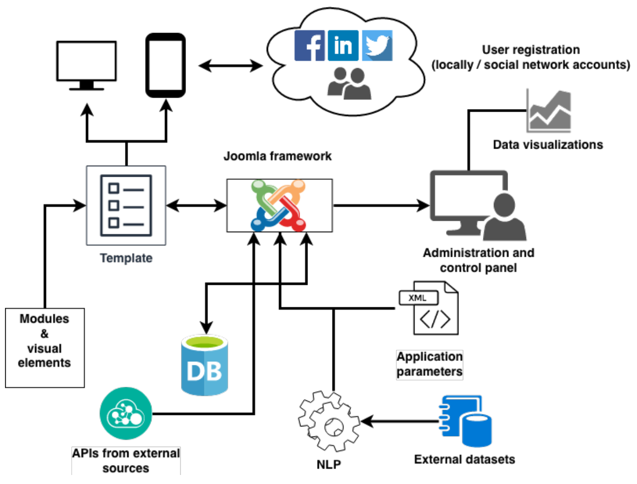


Figure 1: The architectural diagram of the Crowdsourcing platform, presenting the main components of the platform and the data flow.

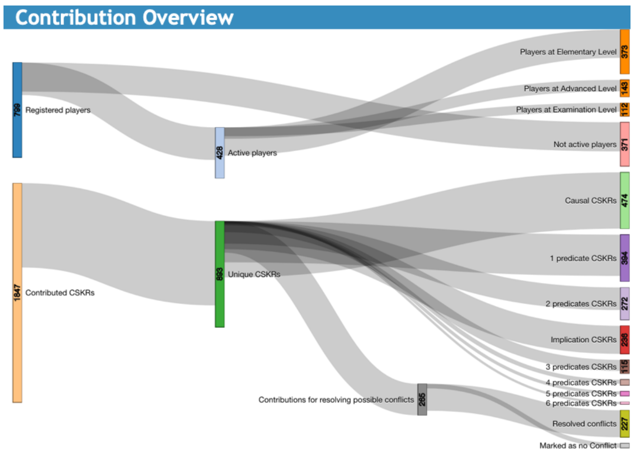


Figure 2: In this figure, a screenshot of a data visualization diagram is depicted where readers can follow the data flow in the system for both players (top stream) and commonsense knowledge (bottom stream).

2.3. Technological Infrastructure

In terms of technological infrastructure, the platform relies on a web-server with Linux-Apache-MariaDB-PHP (LAMP) stack and on the Joomla framework. The platform also utilizes the JQuery[5] and the bootstrap frameworks both for designing elements and for application functionality.
The platform employs the Joomla Model-View-Controller (MVC)[6] framework that allows the development of components by separating the data manipulation functions from the view controls. The controller is responsible for examining the request and determining which processes will be needed to satisfy the request and which view should be used to return the results back to the user. This architecture allows the usage of both internal (e.g., database) and external data sources (e.g., APIs, files) and of course deliver these services in an abstraction layer that can be used by other applications.
For user authentication, both the Joomla internal mechanisms are used and the Oauth[7] authentication methods that permit the seamless integration of social network authentication with the platform.

2.4. Data Visualization

It is important for application developers to be able to visualize acquired knowledge for better understanding what and how users behaved during the crowdsourcing experiment. In Figure 2 an example of a Sankey type graph is presented for the Robot Trainer game where results for both the contributors and the acquired knowledge are depicted on the same diagram. This type of functionality is possible by using the D3.JS library with data feed from the database and a graph theory (network) library for visualization and analysis called Cytoscape.js. The latter was also used for representing and contributing commonsense knowledge rules in a graphical manner in WebStar and was evaluated positively by novice users in conjunction with using a text-based editor for the same task.

3. An Example of Developing a Crowdsourcing Application

In its current state, the platform was used to develop two GWAPs and a crowdsourcing application. There is an extensive presentation of the two GWAPs in our previous work (Rodosthenous and Michael, 2014; Rodosthenous and Michael, 2016) and readers are directed there to learn more about the design, the various elements employed and the experiments performed to acquire commonsense knowledge.
In this section, we focus on how the platform was used for the task of acquiring knowledge in the form of natural language rules for solving cloze tests. For running this task, first we retrieved stories from the ROCStories dataset in a tabular format and loaded them in the platform's database table. Then we parsed each story sentence through the CoreNLP system and got the Parts-Of-Speech (POS) for each word and its base form (lemma). The aforementioned, were stored in a database table. For each story, a noun word was obscured and more than 1000 cloze tests were created. For each test at least 5 possible answers were generated and stored, including synonyms and antonyms retrieved from Wordnet. This workflow was developed in the back-end by reusing components from the two GWAPs and by adding new functionality specifically used for that workflow.
The task was separated in three subtasks and for the front-end design, each one of these tasks is presented on a separate screen (see Figure 3). Each screen comprises an instruction area on top, the active task area below that on the left and the visual representation area on the right. The visual representation area is dynamically updated with every user action. Directly below these two are the task controls. This template, based on bootstrap, was chosen for its simplicity, since we wanted to avoid users paying attention to unnecessary elements.
To start contributing, users need to create an account using

[5]https://jquery.com/
[6]https://docs.joomla.org/Model-View-Controller
[7]https://oauth.net/2/

either the registration form or one of the social media connected account methods inherently present in the platform. After entering their credentials, contributors are firstly presented with a test (see Figure 2a) which they solve and state how confident they are in solving it, in a scale of 0 to 100%. Secondly, the contributors are asked to highlight the words in the text that helped them decide the missing word (see Figure 2b), and thirdly, they are presented with a new test where both the correct answer selected in the 1st step and the highlighted words selected in the 2nd step are present (see Figure 2c). The contributor is asked to verify if the highlighted words are used to find the missing word. Finally, a new test appears which includes the highlighted words from the 2nd step but not the selected missing word from the 1st step. Contributors are asked if the missing word is implied in the story. Each contributor is also presented with a task to verify if the chosen words selected by another contributor are useful for solving the cloze test (see Figure 2d).

Each test is retrieved randomly from the database and for the verification task, tests are selected randomly at first, and by prioritizing selection of tests that have at least one contribution. That way, we give priority to verifying contributions. This is set before running the experiment in the parameters screen on the back-end. All user contributions are recorded and stored in a database table recording both the task data (e.g., missing word, highlighted words, verification response) and metadata (e.g., response time). Recording is possible using the JQuery AJAX libraries and APIs, which allow dynamic update of the content without refreshing the browser webpage and make the contributor to loose focus on the task.

Through these tasks, we are able to acquire knowledge both for cases where the word is explicitly stated in the text and for cases that it is implied. The crowdsourcing application was tested with a small crowd and initial experiments showed that acquired rules can be used both for solving cloze tests and for generating inferences from a story. For example the following two rules were generated and verified on unseen stories:

- when words (or their lemmas) `friends` and `high` exist in a story then the missing word is probably `school`
- when words (or their lemmas) `player` and `scored` exist in a story then the missing word is probably `team`

4. Related Work

Currently, there are many attempts to harness the power of the crowd for several tasks such as image tagging, knowledge gathering, text recognition, etc. The motives for people contributing, are categorized between intrinsic and extrinsic (Kaufmann et al., 2011). Intrinsic motivation includes enjoyment and community based contributions and extrinsic includes immediate and delayed payoffs and social motivation.

For the purpose of acquiring commonsense knowledge there are examples of games and frameworks such as Verbosity (von Ahn et al., 2006), i.e., a GWAP for Collecting Commonsense knowledge facts, Common Consensus (Lieberman et al., 2007), i.e., a GWAP for gathering commonsense goals, GECKA (Cambria et al., 2015), i.e., a game engine for commonsense knowledge acquisition, the Concept Game (Herdagdelen and Baroni, 2010), i.e, a GWAP for verifying commonsense knowledge assertions, the FACTory Game (Lenat and Guha, 1990) where players are asked to verify facts from Cyc, the Virtual Pet and the Rapport games (Kuo et al., 2009) for commonsense data collection and many other.

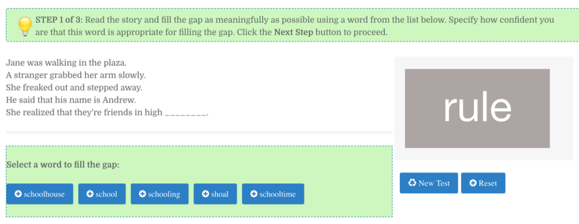
There are also approaches where contributors are motivated by money such as the Amazon Mechanical Turk (Buhrmester et al., 2011) and Figure Eight[8] and others, where motivation is geared towards contributing to science or other noble causes. These approaches rely on citizen science frameworks for crafting crowdsourcing tasks, such as Pybossa[9] and psiTurk[10].

The aforementioned systems and games are very interesting and provide a lot of features, but their design is focused on targeting a single task, rather than a series of tasks that are chained. Furthermore, the majority of systems is limited to the templates and standard workflow processes offered in order to accommodate the most common and most popular crowdsourcing tasks. The task of commonsense knowledge acquisition is more complex and requires more complex workflows to be used, e.g, contribution and then verification.

There are cases where crowdsourcing only is not the best option for acquiring a specific type of knowledge and hybrid solutions, i.e., solutions that employ both human contributors and machine processing, should be used towards that direction. Such an example is the acquisition of commonsense knowledge in the form of rules, where we compared a pure crowdsourcing approach ("Knowledge Coder" game) with a hybrid one ("Robot Trainer" game). The results suggest, that the hybrid approach is more appropriate for gathering general commonsense knowledge rules, that can be used for question-answering. This is one of the reasons we chose to develop a custom made platform in order to have more flexibility in developing such tasks. Ready-made templates offered by the mainstream platforms cannot give you this flexibility, since they aim in a broader set of experimenters. Of course, this comes at the cost that some development should be made from the experimenter.

The crowdsourcing platform has internal mechanisms for knowledge representation in the form of rules, which can be reused in many different applications that serve a similar purpose. Using one of the mainstream platforms requires handling the knowledge rule representation using external tools that need to be developed beforehand. The crowdsourcing platform also engulfs natural language processing tools for treating datasets, before requesting crowd workers to process them. There are also modules for direct integration with knowledge bases (e.g., ConceptNet, YAGO) that can be used in conjunction with the crowd tasks, either for knowledge verification or to reduce the ambiguities in language. The aforementioned features cannot be found in

[8]`https://www.figure-eight.com/`
[9]`https://pybossa.com/`
[10]`https://psiturk.org/`

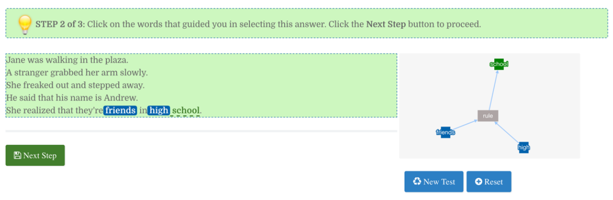


STEP 1 of 3: Read the story and fill the gap as meaningfully as possible using a word from the list below. Specify how confident you are that this word is appropriate for filling the gap. Click the **Next Step** button to proceed.

Jane was walking in the plaza.
A stranger grabbed her arm slowly.
She freaked out and stepped away.
He said that his name is Andrew.
She realized that they're friends in high ________.

Select a word to fill the gap:

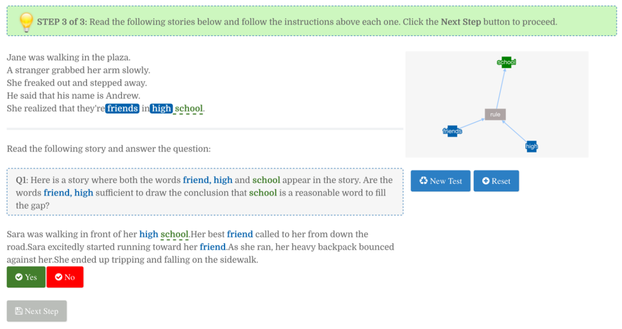
(a) Screenshot of the 1st step where contributors first select a word to fill the gap from one of the possible answers.



STEP 2 of 3: Click on the words that guided you in selecting this answer. Click the **Next Step** button to proceed.

Jane was walking in the plaza.
A stranger grabbed her arm slowly.
She freaked out and stepped away.
He said that his name is Andrew.
She realized that they're friends in high school.

(b) Screenshot of the 2nd step where contributors highlight the words that led in selecting the missing word.



STEP 3 of 3: Read the following stories below and follow the instructions above each one. Click the **Next Step** button to proceed.

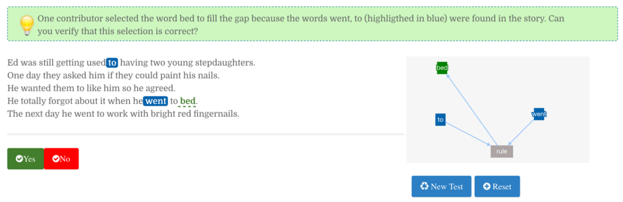
Jane was walking in the plaza.
A stranger grabbed her arm slowly.
She freaked out and stepped away.
He said that his name is Andrew.
She realized that they're friends in high school.

Read the following story and answer the question:

Q1: Here is a story where both the words **friend, high** and **school** appear in the story. Are the words **friend, high** sufficient to draw the conclusion that **school** is a reasonable word to fill the gap?

Sara was walking in front of her **high** school.Her best **friend** called to her from down the road.Sara excitedly started running toward her **friend**.As she ran, her heavy backpack bounced against her.She ended up tripping and falling on the sidewalk.

(c) Screenshot of the 3rd step where contributors verify if the highlighted words from the 2nd step can be used to identify the same missing word with that of the 1st step on a new cloze test.



One contributor selected the word bed to fill the gap because the words went, to (highligthed in blue) were found in the story. Can you verify that this selection is correct?

Ed was still getting used to having two young stepdaughters.
One day they asked him if they could paint his nails.
He wanted them to like him so he agreed.
He totally forgot about it when he went to bed.
The next day he went to work with bright red fingernails.

(d) Screenshot of the verification step where contributors verify other contributors highlighted words used for solving a cloze test.

Figure 3: Screenshots of the online environment for gathering knowledge and delivering cloze tests. The three steps of the process are depicted followed by the verification step.

platforms such as PYBOSSA or psiTurk which concentrate on designing crowdsourcing experiments nor in GECKA which is focused in designing GWAPs.

5. Discussion and Future Work

In this article we presented an overview of the crowdsourcing platform developed to facilitate the development of crowdsourcing applications and GWAPs focused on acquiring commonsense knowledge. Examples of how the platform was used to acquire commonsense knowledge were depicted along with how the various platform elements were used to achieve the goal of the application.

The key features of the crowdsourcing platform include the ability to design complex workflows for acquiring commonsense knowledge, a storage and handling mechanism for acquired knowledge and numerous tools for dataset processing and integration with large semantic knowledge bases and reasoning engines. Moreover the platform offers a wide range of visualizations and analytics to the experimenters that can be customized to facilitate the monitoring and reporting needed during crowd experiments.

In terms of results, from the first GWAP we implemented, i.e., "Knowledge Coder" game, we gathered 93 knowledge rules from 5 contributors. These rules were too specific on the story that was used to generate them and did not offer any value for understanding or answering questions on other stories. When the crowdsourcing platform was used for the "Robot Trainer" game we were able to recruit 800 players from Facebook and some popular game forums in a period of 153 days. These players contributed 1847 commonsense knowledge rules (893 unique). Contributed rules were general enough to be used in other domains, e.g., the symbolic rule `hit(X,Y) IMPLIES angry(X)` meaning that a person X hits a person Y implies that person X is angry. Through the game, 1501 commonsense knowledge rule evaluations were gathered and the interesting part is that players added a "Positive" evaluation to simple rules instead of more complex ones.

We are currently investigating how this work can be used in the context of language learning by using commonsense knowledge databases for creating exercises such as cloze tests, "find synonyms, antonyms, etc." and delivering them to students. The platform can be used to deliver vocabulary exercises, generated from commonsense knowledge databases and ontologies such as ConceptNet. The responses can be used to expand the knowledge bases that the exercises originated from. A prototype implementation of this (Rodosthenous et al., 2019), was developed during the CrowFest organized by the European Network for Combining Language Learning with Crowdsourcing Techniques (EnetCollect) COST Action (Lyding et al., 2018).

The crowdsourcing platform can also be used on our research for identifying the geographic focus of a story. We have developed a system called GeoMantis (Rodosthenous and Michael, 2019) that reads a story and returns the possible countries of focus for that story. GeoMantis uses commonsense knowledge from ConceptNet and YAGO to perform this task. We plan to launch a crowdsourcing task where users will be presented with knowledge about a country, e.g., `parthenon atLocation Greece` and will be asked to evaluate if it is a good argument to identify the geographic focus of a story to that specific country, aiming to add weights on each argument and test if the system yields better results.

6. Bibliographical References

Buhrmester, M., Kwang, T., and Gosling, S. D. (2011). Amazon's Mechanical Turk. *Perspectives on Psychological Science*, 6(1):3–5.

Cambria, E., Rajagopal, D., Kwok, K., and Sepulveda, J. (2015). GECKA: Game Engine for Commonsense Knowledge Acquisition. In *Proceedings of the 28th International Flairs Conference*, pages 282–287.

Diakidoy, I.-A., Kakas, A., Michael, L., and Miller, R. (2015). STAR: A System of Argumentation for Story Comprehension and Beyond. In *Working Notes of the 12th International Symposium on Logical Formalizations of Commonsense Reasoning (Commonsense 2015)*, pages 64–70.

Fellbaum, C. (2010). *Theory and Applications of Ontology: Computer Applications*. Springer Netherlands, Dordrecht.

Herdagdelen, A. and Baroni, M. (2010). The Concept Game: Better Commonsense Knowledge Extraction by Combining Text Mining and a Game with a Purpose. *AAAI Fall Symposium on Commonsense Knowledge, Arlington*, (2006):52–57.

Kaufmann, N., Schulze, T., and Veit, D. (2011). More than fun and money. Worker Motivation in Crowdsourcing-A Study on Mechanical Turk. In *AMCIS*, volume 11, pages 1–11.

Kuo, Y., Lee, J., Chiang, K., and Wang, R. (2009). Community-based Game Design: Experiments on Social Games for Commonsense Data Collection. In *Proceedings of the 1st ACM SIGKDD Workshop on Human Computation (HCOMP 2009)*, pages 15–22, Paris, France.

Law, E. and von Ahn, L. (2011). *Human Computation*. Morgan & Claypool Publishers, 1st edition.

Lenat, D. B. and Guha, R. V. (1990). *Building Large Knowledge-based Systems: Representation and Inference in the Cyc Project*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

Lieberman, H., Smith, D. A., and Teeters, A. (2007). Common Consensus: A Web-Based Game for Collecting Commonsense Goals. In *Proceedings of the Workshop on Common Sense and Intelligent User Interfaces*, Honolulu, Hawaii, USA.

Lyding, V., Nicolas, L., Bédi, B., and Fort, K., (2018). *Introducing the European NETwork for COmbining Language LEarning and Crowdsourcing Techniques (enetCollect)*, pages 176–181. Research-publishing.net.

Manning, C. D., Bauer, J., Finkel, J., Bethard, S. J., Surdeanu, M., and McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Maslan, N., Roemmele, M., and Gordon, A. S. (2015). One Hundred Challenge Problems for Logical Formalizations of Commonsense Psychology. In *Proceedings of the 12th International Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford, California, USA.

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., and Allen, J. (2016). A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. In *Proceedings of the 2016 North American Chapter of the ACL (NAACL HLT)*.

Quilitz, B. and Leser, U., (2008). *Querying Distributed RDF Data Sources with SPARQL*, pages 524–538. Springer Berlin Heidelberg, Berlin, Heidelberg.

Rodosthenous, C. T. and Michael, L. (2014). Gathering Background Knowledge for Story Understanding through Crowdsourcing. In *Proceedings of the 5th Workshop on Computational Models of Narrative (CMN 2014)*, volume 41, pages 154–163, Quebec, Canada. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Rodosthenous, C. and Michael, L. (2016). A Hybrid Approach to Commonsense Knowledge Acquisition. In *Proceedings of the 8th European Starting AI Researcher Symposium*, pages 111–122.

Rodosthenous, C. T. and Michael, L. (2018). Web-STAR: A Visual Web-based IDE for a Story Comprehension System. *Theory and Practice of Logic Programming*, pages 1–43.

Rodosthenous, C. and Michael, L. (2019). Using Generic Ontologies to Infer the Geographic Focus of Text. In Jaap van den Herik et al., editors, *Agents and Artificial Intelligence*, pages 223–246, Cham. Springer International Publishing.

Rodosthenous, C., Lyding, V., König, A., Horbacauskiene, J., Katinskaia, A., ul Hassan, U., Isaak, N., Sangati, F., and Nicolas, L. (2019). Designing a Prototype Architecture for Crowdsourcing Language Resources. In *Proceedings of the 2nd Language, Data and Knowledge (LDK) Conference (to appear)*. CEUR-WS.

Speer, R., Chin, J., and Havasi, C. (2016). ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, pages 697–706.

von Ahn, L. and Dabbish, L. (2008). Designing Games With a Purpose. *Communications of the ACM*, 51(8):57.

von Ahn, L., Kedia, M., and Blum, M. (2006). Verbosity: A Game for Collecting Common-Sense Facts. In *Proceedings of the 25th SIGCHI Conference on Human Factors in Computing Systems (CHI 2006)*, page 75, Montréal, Québec. ACM.