DB-quiz: a DBpedia-backed knowledge game

Jindřich Mynarz
Department of Information and Knowledge Engineering, University of Economics
W. Churchill Sq. 4
130 67 Prague, Czech Republic
jindrich.mynarz@vse.cz

Václav Zeman
Department of Information and Knowledge Engineering, University of Economics
W. Churchill Sq. 4
130 67 Prague, Czech Republic
vaclav.zeman@vse.cz

ABSTRACT

We developed a quiz game that uses questions automatically generated from the Czech and English DBpedia. The game uses class membership and categorization to split DBpedia resources into domains of questions, such as persons or films. Difficulty of questions is estimated by using the indegree graph metric based on links between Wikipedia pages. Surface forms of the DBpedia resources including inbound link labels and select properties are used to remove potential spoilers from questions. We evaluate the relation of indegree to answer success rate and the contribution of semantic features for spoiler removal using online evaluation during game sessions with actual users. Our findings indicate that answer success rate correlates with indegree, but it has a stronger correlation with page views. The evaluation also suggests that additional surface forms improve spoiler detection.

CCS Concepts

•**Information systems** → *Resource Description Framework (RDF);* •**Applied computing** → *Computer games;*

Keywords

games, knowledge bases, DBpedia, RDF

1. INTRODUCTION

DB-quiz is a knowledge-based game that is backed by data from DBpedia [1]. The game uses questions automatically generated from descriptions of DBpedia resources, while the resource labels are used as the answers. In this paper we present the game's design, describe the problems associated with automated generation of questions, discuss our solutions to the problems along with their implementation, and finally evaluate the solutions using data gathered in online user testing.

Surveying the related work, we find that a crucial problem of games built on DBpedia is poor data quality. This issue is mentioned both by [2] and [4], who reports that negative feedback for their game was mostly caused by inconsistencies in DBpedia. The DBpedia extraction framework has improved since the time of these reports but, as we have experienced too, DBpedia's data quality is still poor, so careful data pre-processing and normalization is required. The work of [4] also uses a PageRank-based approach for estimating how well are DBpedia resources known, which is similar to our approach.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SEMANTiCS 2016, September 12 - 15, 2016, Leipzig, Germany

© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ISBN 978-1-4503-4752-5/16/09…$15.00

DOI: http://dx.doi.org/10.1145/2993318.2993339

2. GAME DESIGN

The game design of DB-quiz is based on AZ-kvíz; which is a Czech TV show.[1] The game is played on a triangular board made of hexagonal fields. Each turn players select an available field they want to acquire. Players seize fields when they correctly answer the questions for the fields. Each question is posed as a description of the thing the players must guess. The goal of the game is to get to own fields that continuously spen all 3 sides of the board. Figure 1 shows a screenshot of the game.

When a player picks a field, a question is shown. The player sees an abbreviation of the answer (e.g., *"TBL"* for *"Tim Berners-Lee"*) and a description of the answer (e.g., *"TBL is an English computer scientist, best known as the inventor of the World Wide Web."*). Based on the description the player attempts to guess what the abbreviation stands for. If the guess is correct, the player wins the field, otherwise the field is marked as missed and can be directly acquired by a player without answering a question. When a player submits a guess, it is compared with known surface forms of the correct answer and a verdict is shown informing if the guess was correct. The guess does not need to match the answer exactly. For example, character case, insignificant typos, missing punctuation or diacritics usually does not make the guess incorrect.[2] When half of the player's time for making a guess elapses, a hint is shown that reveals a few letters from the correct answer.[3]

3. PROBLEMS

The design of DB-quiz poses two main problems proceeding from the automated generation of game questions. First, while in AZ-kvíz the difficulty of questions is determined manually, we need to estimate it in an automated way. The

[1]https://cs.wikipedia.org/wiki/AZ-kv%c3%adz

[2]For example, *"tim berners lea"* still matches the answer *"Tim Berners-Lee"*.

[3]For example, if the answer is *"Tim Berners-Lee"*, the hint may be *"T__ Be_n___ L__"*.

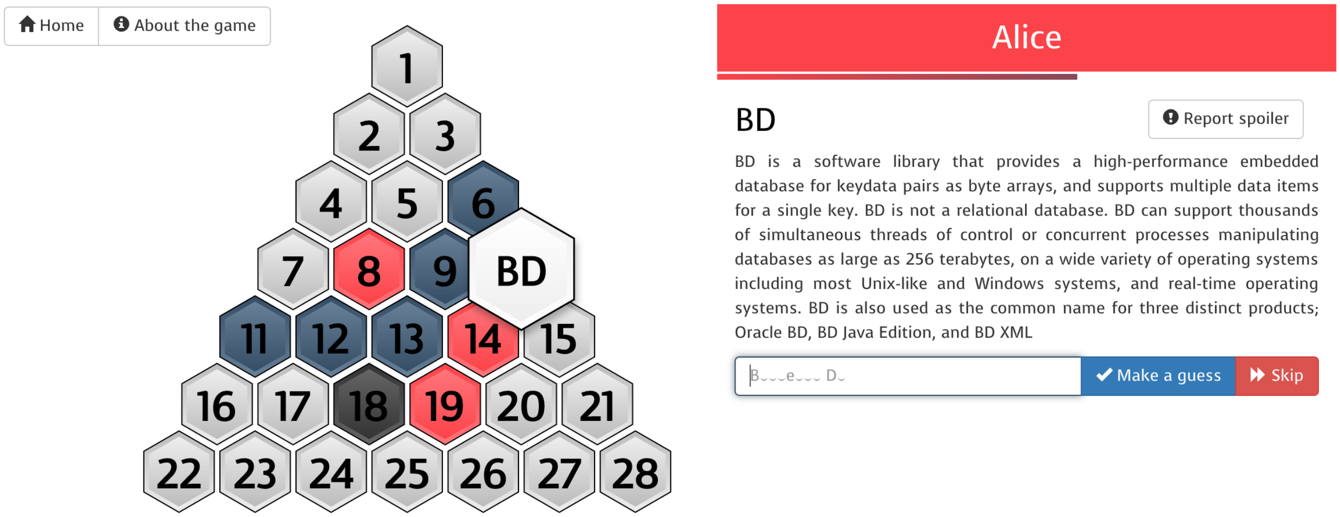

Figure 1: In-game screenshot

second problem is how to clean questions from spoilers that give away the answers.

DB-quiz uses descriptions of DBpedia resources generated from Wikipedia pages as questions. We use indegree of a Wikipedia page as a proxy to its difficulty as a question. Indegree is *"the number of inward directed graph edges from a given graph vertex in a directed graph"* [5]. Indegree of a DBpedia resource can be computed as the number of links to the resource via the `dbo:wikiPageWikiLink`[4] property. This property represents links to Wikipedia pages from other Wikipedia pages. We assume that indegree correlates with how well the resources described in Wikipedia pages are known.

DB-quiz offers three game difficulties: easy, normal, and hard. The problem is how to split questions into these difficulties based on indegree. Using a naïve approach we can split questions ordered by indegree into thirds with even number of members. However, the indegree distribution has a long-tail shape and is dominated by resources with low indegree, as can be seen in Figure 2. If we adopted this approach, many questions assigned with easy or normal difficulty would in fact be hard. Instead of the naïve approach we compute cumulative sums of indegrees of the resources ordered by indegree and make a split when the sum reaches the third of the total sum of indegrees. This way effectively partitions the indegree distribution into thirds of the area under the distribution's curve.

To address the problem of removing spoilers from questions we use their surface forms available in DBpedia. Most surface forms are obtained via the `dbo:wikiPageWikiLinkText` property that contains the anchor texts in links pointing to a resource from other Wikipedia pages. In this way we obtain various spellings or declinations of the resource's label.[5] Other properties used to identify surface forms of a given resource include `dcterms:title` or `dbo:birthName`.[6] Once we obtain surface forms of a resource, we replace them in the resource's description with the abbreviation of the resource's label.

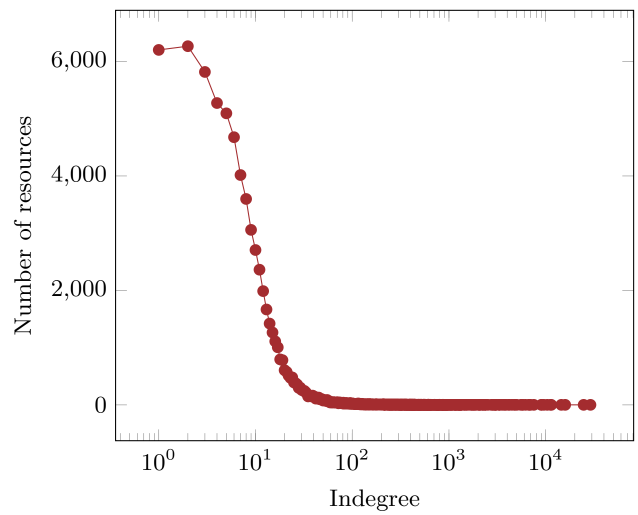

Figure 2: Histogram of indegrees in the Czech DBpedia (indegrees use a logarithmic scale)

4. IMPLEMENTATION

The game supports two kinds of data sources: DBpedia (Czech and English) for automated generation of questions and Google Spreadsheets for manually provided questions. The game uses the RDF properties `rdf:type` and `dcterms:subject` to partition DBpedia resources by domain, keyed by class (e.g., `dbo:Film`) or category (e.g., http://cs.dbpedia.org/resource/Kategorie:Narození_v_Brně for people born in Brno) respectively. The selected subset of resources

[4]Prefixes in this paper can be resolved to their corresponding namespace IRIs by http://prefix.cc.

[5]For example, for *"Tim Berners-Lee"* we get the Czech declination *"Tima Bernerse-Leea"*).

[6]For instance, we can learn that the birth name of the Czech opera singer Emmy Destinn was Emilie Pavlína Věnceslava Kittlová.

is further filtered to match the game's criteria. We exclude resources labelled with abbreviations (e.g., *"R.E.M."*) and we only retrieve those with longer descriptions (> 140 characters) and shorter labels (< 40 characters), so that players have enough hints to make a correct guess without having to type long labels. For resources in each manually selected domain we compute difficulties using the method described in 3. Note that resources can belong into multiple domains, so that they can have multiple difficulties.[7]

Once we obtain the resources for generating questions, their descriptions and labels are normalized. Normalization includes trimming and collapsing whitespace characters or deleting parenthesized substrings, which frequently contain spoilers. We abbreviate the resource's label to its initial letters (e.g., *"Tim Berners-Lee"* becomes *"TBL"*). Subsequently, we replace any occurrence of the label or other surface forms of the resource in its description by this abbreviation. When we compare a guess with the correct answer, both are normalized to increase the tolerance of the match. The normalization includes lower-casing, replacing diacritical characters with their ASCII counterparts, and removing punctuation. Instead of exact match between the normalized guess and the correct answer we compare them by using the Jaro-Winkler string similarity metric with constant scaling factor $p = 0.1$ and a high similarity threshold (0.94, determined by trial), so that a few mismatching characters in the guess are tolerated. We chose this metric because it penalizes the mismatches near the beginning of the guess more than those near the end and it also takes into account the length of the compared strings, so that more errors are accepted for longer strings.

DB-quiz is implemented in Clojurescript[8] as a client-side web application backed by a SPARQL endpoint that provides pre-processed data from DBpedia. Currently it lacks a multiplayer mode, so that players need to take turns playing in the same browser. The source code of the game is openly available at https://github.com/jindrichmynarz/db-quiz. The game can be played at http://mynarz.net/db-quiz/.

5. EVALUATION

We used online user testing to evaluate the computation of question difficulty and the removal of spoilers; the problems addressed in Section 3. We wanted to investigate if correctness of answers correlates with the computed difficulty such that it can justify the method chosen for computing difficulty of questions. To gather data for testing this hypothesis we logged whether answers to questions were correct or not. To test the effectiveness of removing spoilers from questions we allowed users to report the encountered spoilers. We used A/B testing to assess the difference between the number of spoiler reports for questions with spoilers removed and the reports for questions without spoilers removed. While 80 % of players were presented with questions from which surface forms corresponding to potential spoilers were removed, the remaining 20 % were given questions from which only the sought answers were removed. We used Google Analytics[9] to gather data from user sessions. We employed custom events to log data about the correctness of answers and the numbers of reported spoilers per setting for removing spoilers. Using the correctness of answers we derived answer success rate as the ratio of correct answers to all answers to a question. We tested correlations of answer success rates with indegrees and numbers of page views.[10]

Since we do not have enough answers for individual questions, we group the questions with one or more answers to intervals by indegree or page views, such that each interval contains enough questions, and then count mean average success rate for each interval. For this purpose we used a bootstrap method [3] for deriving the success rate within an interval. We used Kendall's tau-b to measure correlation between the averaged success rates and the selected variables. We chose this correlation coefficient because our data contains a large share of ties in answer success rates and this coefficient is adjusted for tied data.

Min. number of questions per interval	**Kendall's rank correlation coefficient between answer success rate and**	
	indegree	**page views**
20	0.37	0.84
50	0.45	0.89
80	0.43	1
100	0.73	0.87

Table 1: Correlations for intervals of questions generated via a bootstrap method

5.1 Results

Data for evaluation was collected during 6 months from October 22, 2015 to April 22, 2016. During this period the application received 1401 visits from 892 users who provided 3514 answers for 2315 questions, reported 242 spoilers, and finished 138 game sessions. The game was advertised via social networks of the authors and on a 2015 Czech Wikipedia conference.

Spoilers were reported 114 times for questions with spoilers removed and 128 times for questions in which spoilers were not removed. Given the 1 to 4 ratio of the sample sizes, this suggests that spoilers are reported approximately 4.5 times more for questions without spoilers removed, and thus indicates that the spoiler removal has an effect.

Resulting correlation coefficients obtained via the described bootstrap method as shown in the Table 1. We discovered indegree has a weak to moderate correlation with answer success rate. However a stronger correlation can be found between answer success rate and number of page views. As can be seen in Figures 3 and 4 there are more outliers in indegrees than in page views, which makes page views a better source for estimating question's difficulty.

The overall correctness of answers per difficulty is depicted in Figure 5. It shows that the aggregated answer success rate corresponds with the question difficulty as expected.

[7]For example, a person may be easy to guess as a person, but relatively unknown as a musician.

[8]https://github.com/clojure/clojurescript

[9]https://analytics.google.com

[10]Via the Pageview API (https://wikitech.wikimedia.org/wiki/Analytics/PageviewAPI) from July 1, 2015 to March 1, 2016.

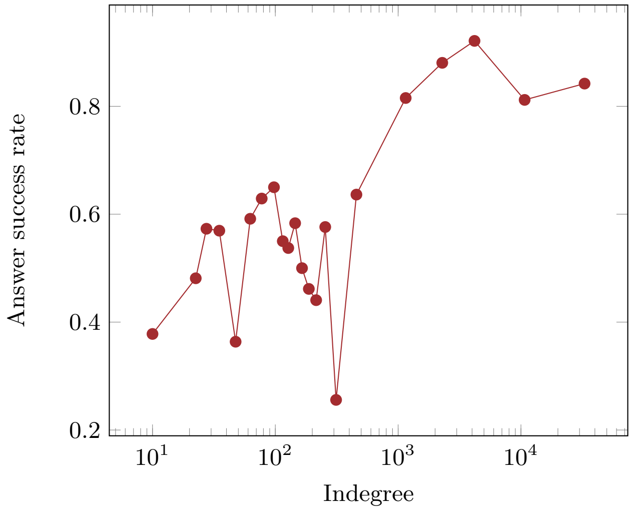

Figure 3: Relationship between answer success rates and indegrees where questions are grouped by indegree into equifrequent intervals

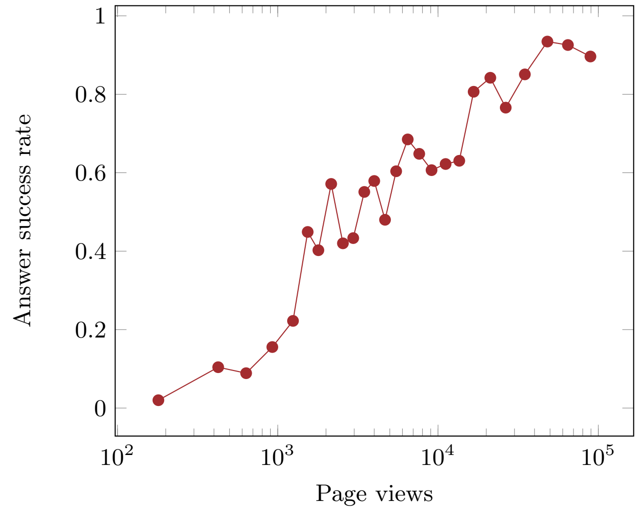

Figure 4: Relationship between answer success rates and page views where questions are grouped by page views into equifrequent intervals

6. CONCLUSION

We presented DB-quiz, a knowledge game that uses questions automatically generated from DBpedia. Automated generation of questions poses two main challenges: estimation of question's difficulty and removal of spoilers from question's description. We proposed to estimate a question's difficulty from its indegree and to remove spoilers from its description based on the question's surface forms found in DBpedia. We evaluated the game via online user testing and found that question difficulty based on indegree positively correlates with answer success rate, however an even stronger correlation can be found for the number of page views of the question's Wikipedia page. The effectiveness of spoiler removal was evaluated using A/B testing and shown that users are more likely to report a spoiler for questions to which the spoiler removal was not applied.

As part of our future work, we plan to use page views to estimate question's difficulty. A key missing feature of the game is a multiplayer mode, which the game lacks due to its implementation complexity. Additionally, in order for the game to be a faithful rendition of the AZ-kvíz it is based on, it should provide yes–no questions for missed fields. This feature poses an interesting challenge for automated generation of questions from knowledge bases such as DBpedia. We plan to carry on with user testing that proved to be useful in evaluation of our hypotheses.

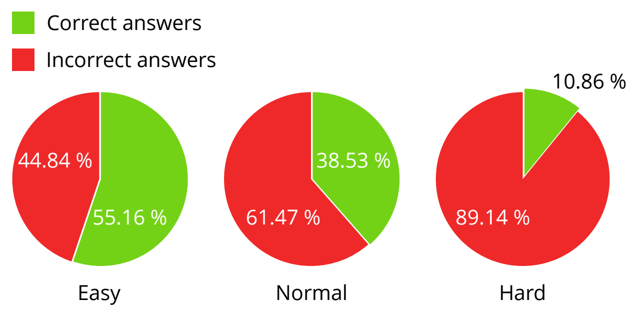

Figure 5: Correctness of answers per difficulty

7. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A nucleus for a web of open data. In K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, J. Goldbeck, P. Mika, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *The Semantic Web: 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11–15, 2007. Proceedings*, pages 722–735, Berlin, Heidelberg, 2007. Springer.

[2] C. Bratsas, D. E. Chrysou, A. Eftychiadou, D. Kontokostas, P. Bamidis, and I. Antoniou. Semantic web game based learning: An i18n approach with greek DBpedia. In *Proceedings of the 2nd International Workshop on Learning and Education with the Web of Data*, volume 840 of *CEUR workshop proceedings*, Aachen, 2012. RWTH Aachen University.

[3] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1994.

[4] J. Waitelonis, N. Ludwig, M. Knuth, and H. Sack. WhoKnows? evaluating linked data heuristics with a quiz that cleans up DBpedia. *International Journal of Interactive Technology and Smart Education*, 8(4):236–248, 2011.

[5] E. W. Weisstein. Indegree. *MathWorld*, 2016.