Microbenchmarks for Question Answering Systems Using QaldGen

Qaiser Mehmood[1]*, Abhishek Nadgeri[2] *, Muhammad Saleem[3], Kuldeep Singh[6], Axel-Cyrille Ngonga Ngomo[4], and Jens Lehmann[5]

[1] INSIGHT, NUIG, Ireland, qaiser.mehmood@insight-centre.org
[2] Service Lee technologies, India, abhishek.n@servify.in
[3] University of Leipzig, Germany, lastname@informatik.uni-leipzig.de
[4] University of Paderborn, Germany, axel.ngonga@upb.de
[5] Fraunhofer IAIS, Germany, jens.lehmann@iais.fraunhofer.de
[6] Nuance Communications, Germany, kuldeep.singh1@nuance.com

Abstract. Microbenchmarks are used to test the individual components of the given systems. Thus, such benchmarks can provide a more detailed analysis pertaining to the different components of the systems. We present a demo of the QaldGen [5], a framework for generating question samples for micro benchmarking of Question Answering (QA) systems over Knowledge Graphs (KGs). QaldGen is able to select customised question samples from existing QA datasets. The sampling of questions is carried out by using different clustering techniques. It is flexible enough to select benchmarks of varying sizes and complexities according to user-defined criteria on the most important features to be considered for QA benchmarking. We evaluate the usability of the interface by using the standard system usability scale questionnaire. Our *overall usability score of 77.25 (ranked* B+*)* suggests that the online interface is *recommendable*, easy to use, and well-integrated[7].

1 Introduction

General-purpose benchmarks are designed to test the overall performance of the system. These benchmarks evaluate the overall performance of the system by providing tests and performance metric. On the other hand, microbenchmarks are more specific, designed to test fine-grained components of the complete system [1]. The tests and performance metric used in such suggested benchmarks is more specific to the components on that is being tested. The proposed benchmark provides more detailed, use-case specific, and component-level evaluations to pinpoint the pro and cons of the system which cannot be achieved from the results of the general-purpose benchmarks.

[*] These two authors contributed equally

Various QA datasets such as LC-QuaD[8] and QALD[9] have been used to evaluate the performance of QA systems over RDF knowledge graphs. In these general-purpose benchmark evaluations, the global metrics of precision, recall, and F-score are used as a performance indicator. Although informative, these evaluations do not shed light on the strength and weakness of a particular component of the QA system. Furthermore, various *benchmarks features* such type of the benchmark questions (e.g., what, who etc.) and the corresponding answers (e.g. boolean, count, list etc.), the number of entities in the question, the number of triple patterns and joins in the corresponding SPARQL queries etc. have a significant impact on the performance of the existing QA systems [3,4]. For instance, the overall winner of the 6th edition of the Question Answering over Linked Data Challenge (QALD6) was CANALI, which suffered limitations when the question started with `"Give me"`. CANALI is outperformed by another QA systems UTQA for such type of questions [3], the problems were highlighted as a result of performing micro analysis on QA system.

To fill this gap, we propose QaldGen [5], a framework for automatic selection of components-level microbenchmarks for QA systems over knowledge graphs. The framework is able to generate question samples customized by for a user in terms of the different QA-related important *benchmarks features*. The framework generates the desired question samples from existing QA datasets (LC-QUAD and QALD9) by using different clustering methods, while considering the customized selection criteria specified by the user.

2 QaldGen Question Sampling Framework

2.1 QaldGenData and Important Benchmark Features

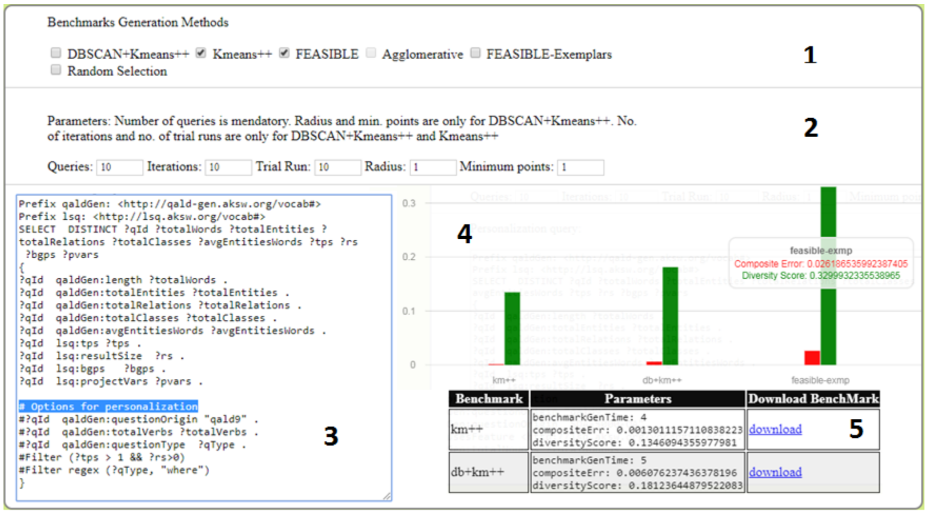
As mentioned before, our framework creates question samples for micro benchmarking from existing well-known QA over KGs datasets LC-QUAD and QALD9. LC-QuAD contains a total of 5000 questions while QALD9 contains a total of 408 questions which also include questions from QALD1-QALD8. We automatically annotate total 5408 questions from QALD9 and LC-QuAD datasets with 51 important QA related features. We then convert the annotated questions into RDF format and name the resulting RDF datasets as *QaldGenData*. This dataset can be reused in training machine learning approaches related to question answering.

2.2 Question Sampling Generation

The benchmark generation is carried out the following four main steps: (1) Select all the questions along with the required features from the input QaldGenData dataset, (2) Generate the feature vectors and normalise them for the input questions, (3) Generate the required number of clusters by using distance-based clustering techniques, (4) Select the single most representative question from each cluster to be included in the final benchmark.

[8] http://lc-quad.sda.tech/

[9] http://qald.aksw.org/



Benchmarks Generation Methods

□ DBSCAN+Kmeans++ ☑ Kmeans++ ☑ FEASIBLE □ Agglomerative □ FEASIBLE-Exemplars
□ Random Selection

Parameters: Number of queries is mendatory. Radius and min. points are only for DBSCAN+Kmeans++. No. of iterations and no. of trial runs are only for DBSCAN+Kmeans++ and Kmeans++

Queries: 10 Iterations: 10 Trial Run: 10 Radius: 1 Minimum points: 1

```
Prefix qaldGen: <http://qald-gen.aksw.org/vocab#>
Prefix lsq: <http://lsq.aksw.org/vocab#>
SELECT  DISTINCT ?qId ?totalWords ?totalEntities ?
totalRelations ?totalClasses ?avgEntitiesWords ?tps ?rs
 ?bgps ?pvars
{
?qId  qaldGen:length ?totalWords .
?qId  qaldGen:totalEntities ?totalEntities .
?qId  qaldGen:totalRelations ?totalRelations .
?qId  qaldGen:totalClasses ?totalClasses .
?qId  qaldGen:avgEntitiesWords ?avgEntitiesWords .
?qId  lsq:tps ?tps .
?qId  lsq:resultSize  ?rs .
?qId  lsq:bgps   ?bgps .
?qId  lsq:projectVars ?pvars .

# Options for personalization
#?qId  qaldGen:questionOrigin "qald9" .
#?qId  qaldGen:totalVerbs ?totalVerbs .
#?qId  qaldGen:questionType  ?qType .
#Filter (?tps > 1 && ?rs>0)
#Filter regex (?qType, "where")
}
```

feasible-exmp
Composite Error: 0.026186535992387405
Diversity Score: 0.3299932335538965

Benchmark	Parameters	Download BenchMark
km++	benchmarkGenTime: 4 compositeErr: 0.0013011157110838223 diversityScore: 0.1346094355977981	download
db+km++	benchmarkGenTime: 5 compositeErr: 0.006076237436378196 diversityScore: 0.18123644879522083	download

Fig. 1. The QaldGen online interface

3 QaldGen Online

The online demo and source code of the QaldGen is available at the QaldGen homepage `http://qaldgen.aksw.org/`. Figure 1 shows the online interface of the QaldGen, which is comprised of five main steps:

1. **Selection of clustering method**: The first step is to select the question sample generation method(s). Currently, our framework supports 6 well-known clustering methods namely DBSCAN+Kmeans++, Kmean++, Agglomerative, Random selection, FEASIBLE and FEASIBLE-Exemplars.
2. **Parameters selection**: The second step is the selection of clustering method-related parameters like the number of queries in the resulting benchmark etc.
3. **Question Sample personalization**: The third step allows to further customize the resulting question sample which can be used for micro benchmarking QA systems. This can be done by using a single SPARQL query.
4. **Results**: The diversity score and the similarity errors for the selected methods will be shown as bar graphs.
5. **Question Sample download**: The resulting micro benchmarks can be finally downloaded to be used in the evaluation.

4 Evaluation

An evaluation of the QaldGen can be found in [5]. To assess the usability of our system, we have used the standardized, ten-item Likert scale-based *System*

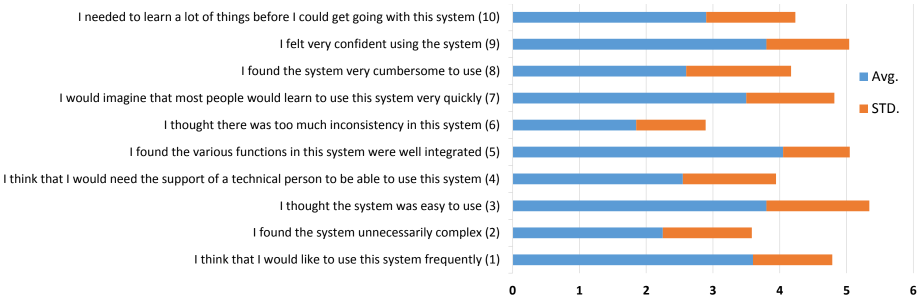


Fig. 2. Result of usability evaluation using SUS questionnaire.

Usability Scale (SUS) [2] questionnaire[10] which can be used for global assessment of systems usability. The survey was posted through Twitter with the ISWC_conf hashtag and was filled by 20 users.[11] The results of SUS usability survey is shown in Figure 2. We achieved a mean usability score of **77.25**[12] indicating that the online interface is recommendable, easy to use, and well-integrated.

Acknowledgment

This work has been supported by the project LIMBO (Grant no. 19F2029I), OPAL (no. 19F2028A), KnowGraphs (no. 860801), Science Foundation Ireland (Grant no. SFI/12/RC/2289), and SOLIDE (no. 13N14456)

References

1. C. Laaber, J. Scheuner, and P. Leitner. Software microbenchmarking in the cloud. how bad is it really? *Empirical Software Engineering*, pages 1–40, 2019.
2. J. R. Lewis and J. Sauro. The factor structure of the system usability scale. In *HCD*. 2009.
3. M. Saleem, S. N. Dastjerdi, R. Usbeck, and A. N. Ngomo. Question Answering Over Linked Data: What is Difficult to Answer? What Affects the F scores? In *NLIWoD*, 2017.
4. K. Singh, I. Lytra, A. S. Radhakrishna, S. Shekarpour, M.-E. Vidal, and J. Lehmann. No one is perfect: Analysing the performance of question answering components over the dbpedia knowledge graph. *arXiv preprint arXiv:1809.10044*, 2018.
5. K. Singh, M. Saleem, A. Nadgeri, F. Conrads, J. Pan, A.-C. N. Ngomo, and J. Lehmann. Qaldgen: Towards microbenchmarking of question answering systems over knowledge graphs. In *ISWC*, 2019.

[10] Our survey can found at: `https://forms.gle/etaUcgRHHH3ima9u5`

[11] As of June 28th, 2019. Responses summary: `https://bit.ly/2XfMy1y`

[12] Please see `https://bit.ly/2xiukgl` for the interpretation of the score