QA2Explanation: Generating and Evaluating Explanations for Question Answering Systems over Knowledge Graph

Saeedeh Shekarpour
University of Dayton
Dayton, USA
`sshekarpour1`
`@udayton.edu`

Abhishek Nadgeri
RWTH Aachen &
Zerotha Research, Germany
`abhishek.nadgeri`
`@rwth-aachen.de`

Kuldeep Singh
Cerence GmbH &
Zerotha Research, Germany
`kuldeep.singh1`
`@cerence.com`

Abstract

In the era of Big Knowledge Graphs, Question Answering (QA) systems have reached a milestone in their performance and feasibility. However, their applicability, particularly in specific domains such as the biomedical domain, has not gained wide acceptance due to their "black box" nature, which hinders transparency, fairness, and accountability of QA systems. Therefore, users are unable to understand how and why particular questions have been answered, whereas some others fail. To address this challenge, in this paper, we develop an automatic approach for generating explanations during various stages of a pipeline-based QA system. Our approach is a supervised and automatic approach which considers three classes (i.e., success, no answer, and wrong answer) for annotating the output of involved QA components. Upon our prediction, a template explanation is chosen and integrated into the output of the corresponding component. To measure the effectiveness of the approach, we conducted a user survey as to how non-expert users perceive our generated explanations. The results of our study show a significant increase in the four dimensions of the human factor from the Human-computer interaction community.

1 Introduction

The recent advances of Question Answering (QA) technologies mostly rely on (i) the advantages of Big Knowledge Graphs which augment the semantics, structure, and accessibility of data, *e.g.,* Web of Data has published around 150B triples from a variety of domains[1], and (ii) the competency of contemporary AI approaches which train sophisticated learning models (statistical models (Shekarpour et al., 2015, 2013), neural networks (Lukovnikov et al., 2017), and attention models (Liu, 2019)) on a large size of training data, and given a variety of novel features captured from semantics, structure, and context of the background data. However, similar to other branches of AI applications, the state of the art of QA systems are *"black boxes"* that fail to provide transparent explanations about why a particular answer is generated. This black box behavior diminishes the confidence and trust of the user and hinders the reliance and acceptance of the black-box systems, especially in critical domains such as healthcare, biomedical, life-science, and self-driving cars (Samek et al., 2017; Miller, 2018). The running hypothesis in this paper is that the lack of explanation for answers provided by QA systems diminishes the trust and acceptance of the user towards these systems. Therefore, by implementing more transparent, interpretable, or explainable QA systems, the end users will be better equipped to justify and therefore trust the output of QA systems (Li et al., 2018).

Furthermore, **data quality** is a critical factor that highly affects the performance of QA systems. In other words, when the background data is flawed or outdated, it undermines the human-likeness and acceptance of the QA systems if no explanation is provided, especially for non-expert users. For example, the SINA engine (Shekarpour et al., 2015) failed to answer the simple question *"What is the population of Canada?"* on the DBpedia (Auer et al., 2007) version 2013, whereas it succeeded for similar questions such as *"What is the population of Germany?"*. The error analysis showed that the expected triple *i.e.,* `<dbr`[2]`:Canada dbo`[3]`:population "xxx">` is missing from DBpedia 2013. Thus, if the QA system does not provide any explanation about such failures, then the non-expert user concludes the QA system into the demerit points. Thus, in general, the errors or

[1] `http://lodstats.aksw.org/`

[2] dbr is bound to `http://dbpedia.org/resource/`.

[3] The prefix `dbo` is bound to `http://dbpedia.org/ontology/`.

Proceedings of the First Workshop on Interactive and Executable Semantic Parsing, pages 1–11
Online, November 19, 2020. ©2020 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17

failures of the QA systems might be caused by the inadequacies of the underlying data or misunderstanding, misinterpretation, or miscomputation of the employed computational models. In either case, the black-box QA system does not provide any explanations regarding the sources of the error. Often the research community obsesses with the technical discussion of QA systems and competes on enhancing the performance of the QA systems, whereas, on the downside of the QA systems, there is a human who plays a vital role in the acceptance of the system. The Human-Computer Interaction (HCI) community already targeted various aspects of the human-centered design and evaluation challenges of black-box systems. However, the QA systems over KGs received the least attention comparing to other AI applications such as recommender systems (Herlocker et al., 2000; Kouki et al., 2017).

Motivation and Approach: Plethora of QA systems over knowledge graphs developed in the last decade (Höffner et al., 2017). These QA systems are evaluated on various benchmarking datasets including WebQuestions (Berant et al., 2013), QALD (Unger et al., 2015), LC-QuAD (Trivedi et al., 2017), and report results based on global metrics of precision, recall, and F-score. In many cases, QA approaches over KGs even surpass the human level performance (Petrochuk and Zettlemoyer, 2018). Irrespective of the underlying technology and algorithms, these QA systems act as black box and do not provide any explanation to the user regarding 1) why a particular answer is generated and 2) how the given answer is extracted from the knowledge source. The recent works towards explainable artificial intelligence (XAI) gained momentum because several AI applications find limited acceptance due to ethical reasons (Angwin et al., 2016) and a lack of trust on behalf of their users (Stubbs et al., 2007). The same rationale is also applicable to the black-box QA systems. Research studies showed that representing adequate explanations to the answer brings acceptability and confidence to the user as observed in various domains such as recommender systems and visual question answering (Herlocker et al., 2000; Hayes and Shah, 2017; Hendricks et al., 2016; Wu and Mooney, 2018). In this paper, we argue that having explanations increases the trustworthiness, transparency, and acceptance of the answers of the QA system over KGs. Especially, when the QA systems fail to answer a question or provide a wrong answer, the explanatory output helps to keep the user informed about a particular behavior. Hence, we propose a template-based explanation generation approach for QA systems. Our proposed approach for explainable QA system over KG provides (i) *adequate justification:* thus the end user feels that they are aware of the reasoning steps of the computational model, (ii) *confidence:* the user can trust the system and has the willing for the continuation of interactions, (iii) *understandability:* educates the user as how the system infers or what are the causes of failures and unexpected answers, and (iv) *user involvement:* encourages the user to engage in the process of QA such as question rewriting.

Research Questions: We deal with two key research questions about the explanations of the QA systems as follows: **RQ1:** *What is an effective model and scheme for automatically generating explanations?* The computational model employed in a QA system might be extremely complicated. The exposure of the depth of details will not be sufficient for the end user. The preference is to generate natural language explanations that are readable and understandable to the non-expert user. **RQ2:** *How is the perception of end users about explanations along the human factor dimensions?*, which is whether or not the explanations establish confidence, justification, understanding, and further engagements of the user.

Our key contributions are: 1) a scheme for shallow explanatory QA pipeline systems, 2) a method for automatically generating explanations, and 3) a user survey to measure the human factors of user perception from explanations. This paper is organized as follows: In Section 2, we review the related work. Section 3 explains the major concepts of the QA pipeline system, which is our employed platform. Section 4 provides our presentation and detailed discussion of the proposed approach. Our experimental study is presented in Section 5, followed by a discussion Section. We conclude the paper in section 7.

2 Related Work

Researchers have tackled the problem of question answering in various domains including open domain question answering (Yang et al., 2019), biomedical (Bhandwaldar and Zadrozny, 2018), geospatial (Punjani et al., 2018), and temporal (Jia et al., 2018). Question answering over publicly available KGs is a long-standing field with over 62 QA systems developed since 2010 (Höffner et al., 2017). The implementation of various QA systems can be broadly categorized into three approaches (Singh, 2019; Diefenbach et al., 2018). The first is a semantic parsing based approach such as (Usbeck et al., 2015) that implements a QA system using several linguistic analyses (e.g., POS tagging, dependency parsing) and linked data technologies. The second approach is an end-to-end machine learning based, which uses a large amount of training data to map an input question to its answer directly (e.g., in (Yang et al., 2019; Lukovnikov et al., 2017)). The third approach is based on modular frameworks (Kim et al., 2017; Singh et al., 2018b) which aims at reusing individual modules of QA systems, independent tools (such as entity linking, predicate linking) in building QA systems collaboratively. Irrespective of the implementation approach, domain, and the underlying knowledge source (KG, documents, relational tables, etc.), the majority of existing QA systems act as a black box. The reason behind black box behavior is due to either the monolithic tightly coupled modules such as in semantic parsing based QA systems or nested and nonlinear structure of machine learning based algorithms employed in QA systems. The modular framework, on the other hand, provides flexibility to track individual stages of the answer generation process. The rationale behind our choice of the modular framework over monolithic QA systems is a flexible architecture design of such frameworks. It allows us to trace failure at each stage of the QA pipeline. We enrich the output of each step with adequate justification with supporting natural language explanation for the user. Hence, as the first step towards explainable QA over knowledge graphs, we propose an automatic approach for generating a description for each stage of a QA pipeline in a state-of-the-art modular framework (in our case: Frankenstein (Singh et al., 2018b)). We are not aware of any work in the direction of explainable question answering over knowledge graphs and we make the first attempt in this paper. Although, efforts have been made to explain visual question answering systems. Some works generate textual explanations for VQA by training a recurrent neural network (RNN) to mimic examples of human descriptions (Hendricks et al., 2016; Wu and Mooney, 2018) directly. The work by (Ngonga Ngomo et al., 2013) can be considered a closest attempt to our work. The authors proposed a template based approach to translate SPARQL queries into natural language verbalization. We employ a similar template-based approach to generate an automatic explanation for QA pipelines.

In other domains, such as expert systems, the earlier attempts providing explanations to the users can be traced back in the early 70s (Shortliffe, 1974). Since then, extensive work has been done to include explanations in expert systems followed by recommender systems to explain the system's knowledge of the domain and the reasoning processes these systems employ to produce results (for details, please refer to (Moore and Swartout, 1988; Jannach et al., 2010; Daher et al., 2017). For a recommender system, work by (Herlocker et al., 2000) is an early attempt to evaluate different implementations of explanation interfaces in "MovieLens" recommender system. Simple statements provided to the customers as explanations mentioning the similarity to other highly rated films or a favorite actor or actress were among the best recommendations of the MovieLens system compared to the unexplained recommendations. Furthermore, applications of explanation are also considered in various sub-domains of artificial intelligence, such as justifying medical decision-making (Fox et al., 2007), explaining autonomous agent behavior (Hayes and Shah, 2017), debugging of machine learning models (Kulesza et al., 2015), and explaining predictions of classifiers (Ribeiro et al., 2016).

3 QA Pipeline on Knowledge Graph

One of the implementation approaches for answering questions from interlinked knowledge graphs is typically a multi-stage process which is called *QA pipeline* (Singh et al., 2018b). Each stage of the pipeline deals with a required task such as Named Entity Recognition (NER) and Disambiguation (NED) (referred as Entity Linking (EL)), Relation extraction and Linking (RL), and Query Building (QB). There is an abundance of components performing QA tasks (Diefenbach et al., 2018). These implementations run on the KGs and have been developed based on AI, NLP, and Semantic Technologies, which accomplish one or more tasks of a QA pipeline (Höffner et al., 2017). Table 1 (Singh et al., 2018b) presents performance of best QA components on the LC-QuAD dataset, implementing QA tasks. The components are Tag Me API (Ferragina and Scaiella, 2010)) for NED, RL (Relation Linking) implemented by RNLIWOD[4] and SPARQL query builder by NLIWOD QB[5]). For example, given the question *"Did Tesla win a nobel prize in physics?"*, the ideal NED component is expected to recognize the keyword *"Tesla"* as a named entity and map it to the corresponding DBpedia resource, i.e. `dbr:Nikola_Tesla`. Similarly, the multi-word unit *"nobel prize in physics"* has to be linked to `dbr:Nobel_Prize_in_Physics`. Thereafter, a component performing RL finds embedded relations in the given question and links them to appropriate relations of the underlying knowledge graph. In our example, the keyword *"win"* is mapped to the relation `dbo:award`. Finally, the QB component generates a formal query (e.g. expressed in SPARQL) (i.e. `ASK {dbr:Nikola_Tesla dbo:award dbr:Nobel_Prize_in_Physics.}`). The performance values in Table 1 are averaged over the entire query inventory.

Table 1: Performance of QA components implementing various QA tasks on LC-QuAD dataset.

QA Component	QA Task	Precision	Recall	F-Score
TagMe	NED	0.69	0.66	0.67
RNLIWOD	RL	0.25	0.22	0.23
NLIWOD QB	QB	0.48	0.49	0.48

4 Approach

A full QA pipeline is required to answer a given question q. Such QA pipelines are composed of all the required components performing necessary tasks to transform a user-supplied natural language (NL) question into a formal query language (*i.e.,* SPARQL). We consider three generic classes for outputs of a full QA pipeline or individual components, namely $O_c = \{Success, NoAnswer, WrongAnswer\}$. Concerning a given question, a "success" class is when the QA pipeline (component) successfully provides a correct output, a "No Answer" class happens when the full QA pipeline (or an individual component) does not return any output and "Wrong Answer" class is when the provided output is incorrect.

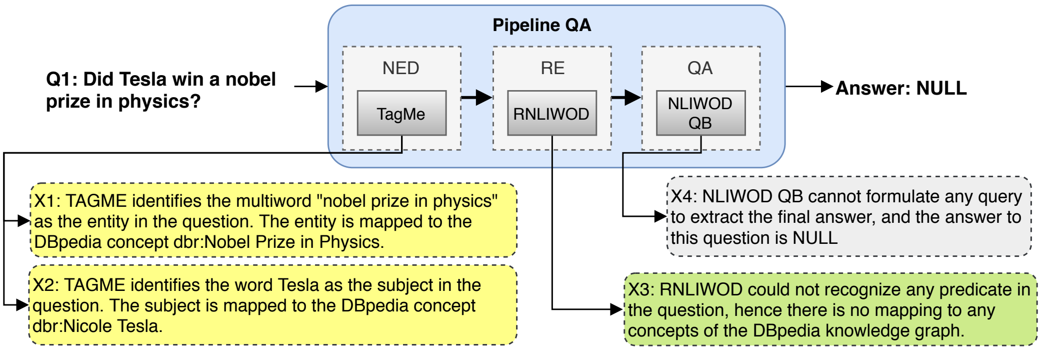
To address **RQ1**, we introduce a scheme for generating explanations for the QA pipeline system. This scheme produces shallow, however automatic explanations using a semi-supervised approach for generating individual explanations after running each integrated component. In our proposed model, the class of the output of each integrated component is predicted using a supervised learning approach. We train a classifier per component within the pipeline. Then based on the prediction of the classifier, an explanation template is chosen. The explanation template and the output of the component are incorporated to form the final representation of explanations. We have a repository of explanation templates for each component of the QA pipeline system. For example, the NED component corresponds to several explanation templates differing based on the number of the output entities. Precisely, the explanation template when the NED has one single entity is different from when it has two or three. Moreover, the templates vary based on the Part of Speach (POS) tag of the entities recognized in the input question. For example, Figure 1 shows a pipeline containing three components: 1) NED component: TagMe, 2) RL component: RNLIWOD QB, and 3) QB component: NLIWOD QB. Three classifiers were individually trained for each component. In this example, for the given question *"Did Tesla win a nobel prize in physics?"* the classifiers predicted the class of "Success" for NED and the class "No Answer" for RL and QB components. Thus, the explanation templates corresponding to the class of "success" for NED, and "No Answer" for RL and QB are filtered. Then since the NED component has two outputs, therefore, two explanations were generated for NED, whereas the remaining components show one explanation.

4.1 Predicting Output of Components

The set of necessary QA tasks formalized as $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ such as NED, RL, and QB. Each task $(t_i : q^* \rightarrow q^+)$ transforms a given representation q^* of a question q into another representation q^+. For example, NED and RL tasks transform the input representation *"What is the capital of Finland?"* into the representation *"What is the* `dbo:capital` *of* `dbr:Finland`*?"*. The entire set of QA components is denoted by $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$. Each component C_j solves one single QA task; $C_j^{t_i}$ corresponds to the QA task t_i in $\mathcal{T}$ implemented by C_j. For example, RNLIWOD implements the relation linking QA task, i.e. $RNLIWOD^{RL}$. Let $\rho(C_j)$ denote the performance of a QA component, then our key objective is to predict the likelihood of $\rho(C_j)$

[4] Component is similar to Relation Linker of https://github.com/dice-group/NLIWOD

[5] Component is based on https://github.com/dice-group/NLIWOD and (Unger et al., 2012).



X1: TAGME identifies the multiword "nobel prize in physics" as the entity in the question. The entity is mapped to the DBpedia concept dbr:Nobel Prize in Physics.

X2: TAGME identifies the word Tesla as the subject in the question. The subject is mapped to the DBpedia concept dbr:Nicole Tesla.

X3: RNLIWOD could not recognize any predicate in the question, hence there is no mapping to any concepts of the DBpedia knowledge graph.

X4: NLIWOD QB cannot formulate any query to extract the final answer, and the answer to this question is NULL

Figure 1: The QA pipeline generates the explanations in various stages of running; each explanation is generated per output of each integrated component. The demonstrated pipeline contains three components, i.e., NED, RL, and QB; the output(s) of each one is integrated into an explanation template and represented to the end user.

for a given representation q^* of q, a task t_i, and an underlying knowledge graph λ. This is denoted as $Pr(\rho(C_j)|q^*, t_i, \lambda)$. In this work, we assume a single knowledge graph (i.e. DBpedia); thus, λ is considered a constant parameter that does not impact the likelihood leading to:

$$Pr(\rho(C_j)|q^*, t_i) = Pr(\rho(C_j)|q^*, t_i, \lambda) \qquad (1)$$

Further, we assume that the given representation q^* is equal to the initial input representation q for all the QA components, i.e. $q^* = q$.

Solution Suppose we are given a set of NL questions $\mathcal{Q}$ with the detailed results of performance for each component per task. We can then model the prediction goal $Pr(\rho(C_j)|q, t_i)$ as a supervised learning problem on a training set, i.e. a set of questions $\mathcal{Q}$ and a set of labels $\mathcal{L}$ representing the performance of C_j for a question q and a task t_i. In other words, for each individual task t_i and component C_j, the purpose is to train a supervised model that predicts the performance of the given component C_j for a given question q and task t_i leveraging the training set. If $|\mathcal{T}| = n$ and each task is performed by m components, and the QA pipeline integrates all the $n \times m$ components, then $n \times m$ individual learning models have to be built up.

Question Features. Since the input question q has a textual representation, it is necessary to automatically extract suitable features, i.e. $\mathcal{F}(q) = (f_1, \dots, f_r)$. In order to obtain an abstract and concrete representation of NL questions, we reused question features proposed by (Singh et al., 2018b, 2019) which impact the performance of the QA systems. These features are: question length, answer type (list, number, boolean), Wh-word (who,what,which,etc.), and POS tags present in a question. Please note, our contribution is not the underlying Frankenstein framework, we reused it for the completion of the approach. Our contribution is to add valid explanation to each step of the QA pipeline, and empirical study to support our hypothesis.

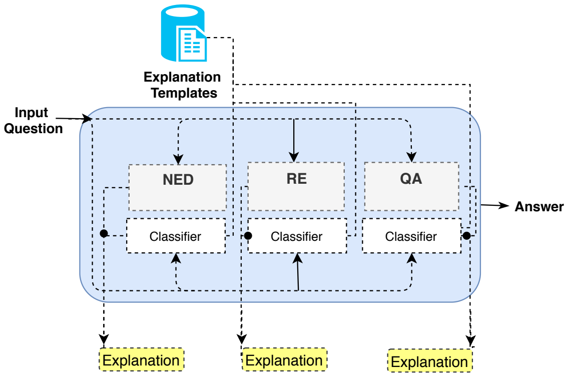


Figure 2: This figure sketches a top overview of our approach. There is a classifier for each component, which predicts the output of the associated component. Also, there is a repository of the explanation templates. Thus, based on the prediction of the classifier and the actual output of the component, a suitable template is filtered. For final explanation, the output of the component was incorporated into the template.

4.2 Methodology

Figure 2 shows the architecture of our approach. Initially, a pipeline for a QA system is built up; in our case we used Frankenstein platform (Singh et al., 2018b,a) to facilitate building up a pipeline. Please note, we do not aim to build a new QA system and reused an existing implementation. We extend the Frankenstein QA pipeline as illustrated in Figure 2. We rely on the best performing pipeline reported in (Singh et al., 2018b) over LC-QuAD dataset

(Trivedi et al., 2017). In addition, we manually populated a repository of explanation templates. For example, all the required explanation templates for NED components are created for cases such as templates for wrong answers, when components produce no answer, and in the case of correct answers. Similarly, the templates for other tasks such as RE an QB were handcrafted. Please note that these templates are generic, thereby they do not depend on the employed component. For example, if we integrate another NED component rather than TagMe, there is no need to update the template repositories. In the next step, we trained classifiers based on the settings which will be presented in the next section. Thus, when a new question arrives at the pipeline, in addition to running the pipeline to exploit the answer, our trained classifiers are also executed. Then the predictions of the classifiers lead us to choose appropriate templates from the repositories. The filtered templates incorporate the output of the components to produce salient representations for NL explanations. The flow of the explanations is represented to the end user besides the final answer.

Templates for Explanation To support our approach for explainable QA, we handcrafted 11 different templates for the explanation. We create placeholders in the predefined templates to verbalize the output of the QA components. Consider the explanation provided in Figure 1. The original template for explaining the output of TagMe component is: `TagMe identifies the multiword` **`X`** `as the entity in the question. The entity is mapped to the DBpedia concept` **`dbr:W`**`.` The placeholders **X** and **dbr:W** are replaced accordingly for each question if a classifier selects this template in its prediction.

5 Experimental Study

We direct our experiment in response to our two research questions (*i.e.,* RQ1 and RQ2) respectively. First, we pursue the following question *"How effective is our approach for generating explanations?"* This evaluation implies the demonstration of the success of our approach in generating proper explanations. It quantitatively evaluates the effectiveness of our approach. On the contrary, the second discourse of the experiment is an HCI study in response to the question *"How effective is the perception of the end user on our explanations?"* This experiment qualitatively evaluates user perception based on the human factors introduced earlier (cf. Section 1). In the following Subsections, we detail our experimental setups, achieved results, and insights over the outcomes of the evaluation.

5.1 Quantitative Evaluation

This experiment is concerned with the question *"How effective is our approach for generating explanations?"*. We measure the effectiveness in terms of the preciseness of the explanations. Regarding the architecture of our approach, choosing the right explanation template depends on the prediction of the classifiers. If classifiers precisely predict a correct output for the underlying components, then consequently, the right templates will be chosen.

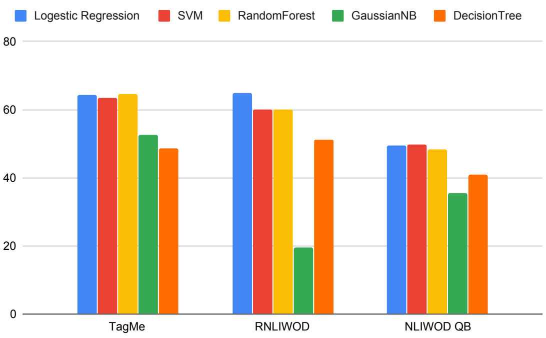


Figure 3: This figure illustrates the accuracy of five classifiers per QA component: TagMe, RNLIWOD, and NLIWOD QB. Logistic Regression classifier performs best for all the components.

In other words, any flaw in the prediction leads to a wrong template. Thus, here we present the accuracy of our classifiers per component. We consider three generic classes, namely $O_c = \{Success, NoAnswer, WrongAnswer\}$ (cf. section 4) for the outputs of individual components. A benchmarking approach has been followed to choose best classifier per task. We employ five different classifiers (SVM, Logistic Regression, Random Forest, Gaussian NB, and Decision Tree) and calculated each classifier's accuracy per component. To train the classifiers per component, we require to create a single dataset. The sample-set in training is formed by considering questions of the LC-QuAD dataset. To get the concrete representation of each question, we extracted the following features: question length, headword(who, what, how), answer types (boolean, number, list), and POS tags. If a particular feature is present, we consider the value 1; if not, then the value of that feature is 0 while representing the question. The label set of the training datasets for a given component was set up by measuring the micro F-Score of every given question for 3,253 questions from the LC-QuAD dataset. The F-score per question is calculated by adopting the same methodology proposed by (Singh et al., 2018b). We rely on 3,253 questions out of 5,000 questions of the LC-QuAD dataset because the gold standard SPARQL queries of the remaining 1,747 questions do not return any answer from DBpedia endpoint (also reported by (Azmy et al., 2018)). The classifier predicts if a component can answer the question or not, and trained using features extracted from the natural language questions against the F score per question. During the training phase, each classifier was tuned with a range of regularization on the dataset. We used the cross-validation approach with 10 folds on the LC-QuAD dataset. We employ a QA pipeline containing TagMe (Ferragina and Scaiella, 2010) for entity disambiguation, RNLIWOD[6] for relation linking, and NLIWOD QB[7] for SPARQL query builder. Figure 3 reports the accuracy of five classifiers (average of all classes). Furthermore, Table 2 reports the accuracy of the best classifier (Logistic Regression in our case) for each component.

Component	Accuracy
TagMe	0.64
RNLIWOD	0.60
NLIWOD WB	0.49

Table 2: **Accuracy of our multi-class classifier for predicting type of explanation for each component.**

Observations. We observe that the logistic regression classifier performs best for predicting the output of components. However, the accuracy of the classifier is low as depicted in the Table 2. (Singh et al., 2018b) report accuracy of *binary classifiers* for TagMe, RNLIWOD, and NLIWOD QB as 0.75, 0.72, and 0.65 respectively. When we train *multi-class classifiers* (*i.e.,* three classes) on the same dataset, we observe a drop in the accuracy. The main reason for the low performance of the classifiers is the low component accuracy (c.f. Table 1)

5.2 User Perception Evaluation

In the second experiment, we pursue the following research question: *"How is the perception of end user about explanations along the human factor dimensions?"* To respond to this question, we conduct the following experiment:

Experimental Setup: We perform a user study to evaluate how the explanations impact user perception. We aim at understanding user's feedback on the following four parameters inspired by (Ehsan et al., 2019; Ehsan and ark Riedl, 2019): 1) `Adequate Justification`: Does a user feel the answer to a particular question is justified or provided with the reasoning behind inferences of the answer? 2) `Education`: Does the user feel educated about the answer generation process so that she may better understand the strengths and limitations of the QA system? 3) `User involvement`: Does the user feel involved in allowing the user to add her knowledge and inference skills to the complete decision process? 4) `Acceptance`: Do explanations lead to a greater acceptance of the QA system in future interactions? With respect to the above criteria, we created an online survey to collect user feedback. The survey embraces random ten questions from our underlying dataset from a variety of answer types such as questions with the correct answer, incorrect answer, no answer (for which classifiers predict correct templates). The first part of the survey displays the questions to the user without any explanation. In the second part, the same ten questions, coupled with the explanations generated by our approach, are displayed to the user. The participants of the survey are asked to rate each representation of question/answer based on the four human factor dimensions (i.e., acceptance, justification, user involvement, and education). The rating scale is based on the Likert scale, which allows the participants to express how much they agree or disagree with a given statement (1:strongly disagree – 5:strongly agree). We circulated the survey to several channels of the co-authors' network, such as a graduate class of Semantic Web course, research groups in the USA and Europe, along with scientific mailing lists. Collectively we received responses from 80 participants. Please note, the number of participants is at par with the other explainable studies such as (Ehsan et al., 2019).

Results and Insights. Figure 4 summarizes the ratings of our user study. We evaluate the user responses based on the four human factor dimen-

[6] Component is similar to Relation Linker of `https://github.com/dice-group/NLIWOD`

[7] Component is based on `https://github.com/dice-group/NLIWOD` and (Unger et al., 2012).

sions: `Adequate Justification`, `Education`, `User involvement`, and `Acceptance`. The summary of ratings for each dimension was captured in one individual chart. The green bars show the feedback over questions with provided explanations, and on the contrary, red bars are aggregated over the question with no explanation. The x-axis shows the Likert scale. The Y-axis is the distribution of users over the Likert scale for each class independently- with explanation and without explanation. Overall it shows a positive trend towards the agreement with the following facts; the provided explanations helped users to understand the underlying process better, justify a particular answer, involve the user in the complete process, and increase the acceptability of the answers. The green bars are larger in positive ratings, such as strongly agree.

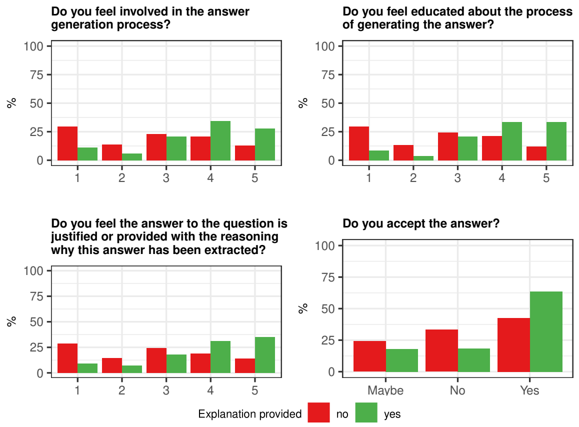


Figure 4: User perception Evaluation. The figure illustrates the comparative analysis of providing with and without explanation to the user. We consider the mean of all the responses. X-axis depicts the Likert scale (1 is strongly disagree, 5 is strongly agree). A clear trend in user responses shows that across all four parameters, there are many answers towards disagreement or neutral when no explanation is provided. In the case of explanation, users feel involved, and responses are shifted towards the agreement. Furthermore, users show more trust in the acceptance of the answer when provided with an explanation.

6 Discussion

In this paper, we focus on the challenge of explainable QA systems. We mainly target systems that consume data from the KGs. These systems receive a natural language question and then transform that to a formal query. Our primary aim is to take the initial steps to break down the full black-box QA systems. Thus, we reuse an existing QA pipeline systems since it already decompose the prominent tasks of the QA systems and then integrate individual implementations for each QA task. We based our approach and associated evaluation on the hypothesis that every component integrated into the pipeline should explain the output. It will educate and involve non-expert users and trigger them to trust and accept the system. Our findings in Section 5 support our hypothesis both on quantitative and qualitative evaluation. The limitation of our approach is that it heavily relies on the performance of the components. In the case of having low performing components, the accuracy of the classifiers is also downgraded. Although, on the one hand, this approach is shallow, one the other hand it avoids exposing the user to overwhelming details of the internal functionalities by showing succinct and user-friendly explanations. (Hoffman et al., 2017) noted that for improving the usability of XAI systems, it is essential to combine theories from social science and cognitive decision making to validate the intuition of what constitutes a "good explanation." Our work in this paper is limited to predefined template based explanations, and does not consider this aspect. Also, our work does not focus on the explainability of the behavior of the employed classifier, and the explanations only justify the final output of components.

7 Conclusion and Future Direction

In this paper, we proposed an approach that is automatic and supervised for generating explanations for a QA pipeline. Albeit simple, our approach intuitively expressive for the end user. This approach requires to train a classifier for every integrated component, which is costly in case the components are updated (new release) or replaced by a latest outperforming component. Our proposed approach induced in a QA pipeline of a modular framework is the first attempt for explainable QA systems over KGs. It paves the way for future contributions in developing explainable QA systems over KGs. Still, there are numerous rooms in this area that require the attention of the research community – for example, explanations regarding the quality of data, or metadata, or credibility of data publishers. Furthermore, recent attempts have been made to provide explanations of machine learning models (Guo et al., 2018). However, the inclusion of the explanations in neural approaches for question answering (such as in (Lukovnikov et al., 2017)) is still an open research question, and we plan to extend our work in this direction. The concerning domain of the system is also influential in explanations. for example, biomedical or marketing domains require various levels of details of explanations. In general, all of these concerns affect the acceptance and trust of the QA system by the end user. Our ambitious vision is to provide personalized and contextualized explanations, where the user feels more involved and educated.

References

Julia Angwin, Jeff Larson, Surya Mattu, and Lauren KirchnerIan Sample. 2016. Machine bias.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*

Michael Azmy, Peng Shi, Jimmy Lin, and Ihab Ilyas. 2018. Farewell freebase: Migrating the simplequestions dataset to dbpedia. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2093–2103.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1533–1544.

Abhishek Bhandwaldar and Wlodek Zadrozny. 2018. Uncc qa: A biomedical question answering system. *EMNLP 2018*, page 66.

Julie Daher, Armelle Brun, and Anne Boyer. 2017. A review on explanations in recommender systems.

Dennis Diefenbach, Vanessa López, Kamal Deep Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information Systems*, 55(3):529–569.

Upol Ehsan and ark Riedl. 2019. On design and evaluation of human-centered explainable ai system. In *Human-Centered Machine Learning Perspectives Workshop at CHI.*

Upol Ehsan, Pradyumna Tambwekar, Larry Chan, Brent Harrison, and Mark O. Riedl. 2019. Automated rationale generation: a technique for explainable AI and its effects on human perceptions. In *Proceedings of the 24th International Conference on Intelligent User Interfaces, IUI 2019, Marina del Ray, CA, USA, March 17-20, 2019*, pages 263–274.

Paolo Ferragina and Ugo Scaiella. 2010. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 1625–1628.

John Fox, David Glasspool, Dan Grecu, Sanjay Modgil, Matthew South, and Vivek Patkar. 2007. Argumentation-based inference and decision making–a medical perspective. *IEEE intelligent systems*, 22(6):34–41.

Wenbo Guo, Sui Huang, Yunzhe Tao, Xinyu Xing, and Lin Lin. 2018. Explaining deep learning models–a bayesian non-parametric approach. In *Advances in Neural Information Processing Systems*, pages 4514–4524.

Bradley Hayes and Julie A Shah. 2017. Improving robot controller transparency through autonomous policy explanation. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, pages 303–312. IEEE.

Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. 2016. Generating visual explanations. In *European Conference on Computer Vision*, pages 3–19. Springer.

Jonathan L Herlocker, Joseph A Konstan, and John Riedl. 2000. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250. ACM.

Robert R Hoffman, Shane T Mueller, and Gary Klein. 2017. Explaining explanation, part 2: empirical foundations. *IEEE Intelligent Systems*, 32(4):78–86.

Konrad Höffner, Sebastian Walter, Edgard Marx, Ricardo Usbeck, Jens Lehmann, and Axel-Cyrille Ngonga Ngomo. 2017. Survey on challenges of Question Answering in the Semantic Web. *Semantic Web*.

Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. 2010. *Recommender systems: an introduction*. Cambridge University Press.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018. Tequila: Temporal question answering over knowledge bases. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1807–1810. ACM.

Jin-Dong Kim, Christina Unger, Axel-Cyrille Ngonga Ngomo, André Freitas, Young-gyun Hahm, Jiseong Kim, Sangha Nam, Gyu-Hyun Choi, Jeong-uk Kim, Ricardo Usbeck, et al. 2017. OKBQA Framework for collaboration on developing natural language question answering systems.

Pigi Kouki, James Schaffer, Jay Pujara, John O'Donovan, and Lise Getoor. 2017. User preferences for hybrid explanations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 84–88. ACM.

Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. 2015. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th international conference on intelligent user interfaces*, pages 126–137. ACM.

Qing Li, Jianlong Fu, Dongfei Yu, Tao Mei, and Jiebo Luo. 2018. Tell-and-answer: Towards explainable visual question answering using attributes and captions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1338–1346.

Heguang Liu. 2019. Conditioning lstm decoder and bidirectional attention based question answering system. *arXiv preprint arXiv:1905.02019*.

Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. 2017. Neural network-based question answering over knowledge graphs on word and character level. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 1211–1220.

Tim Miller. 2018. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*.

Johanna D Moore and William R Swartout. 1988. Explanation in expert systems: A survey. Technical report, University of Southern California, USA.

Axel-Cyrille Ngonga Ngomo, Lorenz Bühmann, Christina Unger, Jens Lehmann, and Daniel Gerber. 2013. Sorry, i don't speak sparql: translating sparql queries into natural language. In *Proceedings of the 22nd international conference on World Wide Web*, pages 977–988. ACM.

Michael Petrochuk and Luke Zettlemoyer. 2018. Simplequestions nearly solved: A new upperbound and baseline approach. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 554–558.

Dharmen Punjani, K Singh, Andreas Both, Manolis Koubarakis, Ioannis Angelidis, Konstantina Bereta, Themis Beris, Dimitris Bilidas, T Ioannidis, Nikolaos Karalis, et al. 2018. Template-based question answering over linked geospatial data. In *Proceedings of the 12th Workshop on Geographic Information Retrieval*, page 7. ACM.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM.

Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*.

Saeedeh Shekarpour, Edgard Marx, Axel-Cyrille Ngonga Ngomo, and Sören Auer. 2015. SINA: semantic interpretation of user queries for question answering on interlinked data. *J. Web Semant.*, 30:39–51.

Saeedeh Shekarpour, Axel-Cyrille Ngonga Ngomo, and Sören Auer. 2013. Question answering on interlinked data. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 1145–1156.

Edward H Shortliffe. 1974. A rule-based computer program for advising physicians regarding antimicrobial therapy selection. In *Proceedings of the 1974 annual ACM conference-Volume 2*, pages 739–739. ACM.

Kuldeep Singh. 2019. Towards dynamic composition of question answering pipelines. *Doctoral Thesis, University of Bonn, Germany*.

Kuldeep Singh, Andreas Both, Arun Sethupat Radhakrishna, and Saeedeh Shekarpour. 2018a. Frankenstein: A platform enabling reuse of question answering components. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, pages 624–638.

Kuldeep Singh, Arun Sethupat Radhakrishna, Andreas Both, Saeedeh Shekarpour, Ioanna Lytra, Ricardo Usbeck, Akhilesh Vyas, Akmal Khikmatullaev, Dharmen Punjani, Christoph Lange, Maria-Esther Vidal, Jens Lehmann, and Sören Auer. 2018b. Why reinvent the wheel: Let's build question answering systems together. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1247–1256.

Kuldeep Singh, Muhammad Saleem, Abhishek Nadgeri, Felix Conrads, Jeff Pan, Axel-Cyrille Ngonga Ngomo, and Jens Lehmann. 2019. Qaldgen: Towards microbenchmarking of question answering systems over knowledge graphs. In *ISWC*.

Kristen Stubbs, Pamela J Hinds, and David Wettergreen. 2007. Autonomy and common ground in human-robot interaction: A field study. *IEEE Intelligent Systems*, 22(2):42–50.

Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. 2017. LC-QuAD: A Corpus for Complex Question Answering over Knowledge Graphs. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, pages 210–218. Springer.

Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and

Philipp Cimiano. 2012. Template-based question answering over RDF data. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, April 16-20, 2012*, pages 639–648. ACM.

Christina Unger, Corina Forascu, Vanessa López, Axel-Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano, and Sebastian Walter. 2015. Question Answering over Linked Data (QALD-5). In *Working Notes of CLEF 2015 - Conference and Labs of the Evaluation forum, Toulouse, France, September 8-11, 2015.* CEUR-WS.org.

Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Lorenz Bühmann, and Christina Unger. 2015. Hawk–hybrid question answering using linked data. In *European Semantic Web Conference*, pages 353–368. Springer.

Jialin Wu and Raymond J Mooney. 2018. Faithful multimodal explanation for visual question answering. *arXiv preprint arXiv:1809.02805*.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. *NAACL HLT 2019*, page 72.