Generating Questions from Wikidata Triples

Kelvin Han, Thiago Castro Ferreira, Claire Gardent

Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 277–290
Marseille, 20-25 June 2022

Generating Questions from Wikidata Triples

Kelvin Han[1], **Thiago Castro Ferreira**[2], **Claire Gardent**[1]
[1]CNRS/LORIA, Université de Lorraine
[2]aiXplain, inc., Federal University of Minas Gerais
[1]Nancy, France, [2] Belo Horizonte, Brazil
huiyuan.han@loria.fr, thiagocf05@ufmg.br, claire.gardent@loria.fr

Abstract

Question generation from knowledge bases (or knowledge base question generation, KBQG) is the task of generating questions from structured database information, typically in the form of triples representing facts. To handle rare entities and generalize to unseen properties, previous work on KBQG resorted to extensive, often ad-hoc pre- and post-processing of the input triple. We revisit KBQG – using pre-training, a new (triple, question) dataset and taking question type into account – and show that our approach outperforms previous work both in a standard and in a zero-shot setting. We also show that the extended KBQG dataset (also helpful for knowledge base question answering) we provide allows not only for better coverage in terms of knowledge base (KB) properties but also for increased output variability in that it permits the generation of multiple questions from the same KB triple. Our code and dataset can be found at: `https://gitlab.inria.fr/hankelvin/wikidataqg`

Keywords: question generation, knowledge bases, KBQG, Wikidata

1. Introduction

With the rise of large scale knowledge bases (KBs) such as Wikidata (Vrandečić and Krötzsch, 2014), DBpedia (Auer et al., 2007) and Cyc (Lenat and Guha, 1993), large amounts of factual data has become available which can be used to answer factual questions. In that context, teaching machines to generate a question from a KB item (question generation from KB, KBQG) has become an important issue with multiple potential applications. By translating a KB fact (e.g., (HENRY_POINCARÉ , BIRTHPLACE , FRANCE)) into a natural language (NL) question (e.g., *Where was Henri Poincaré born?*), KBQG facilitates access to KBs by non experts. It could help improve the ability of dialog models to ask factual questions and support the development of tutoring systems that ask the user a series of questions about some KB entity. Finally, it is useful for creating or augmenting the sets of (KB content, NL question) pairs necessary to train Question Answering (QA) systems on KBs (KBQA). However the scale of these knowledge bases, the high number of rare entities they contain and the lack of NL aliases for KB relations still leave this task a challenging problem.

The state of the art in KBQG have mainly focused on how to address these rare entity and unknown relation issues. Typically, the KB input is enriched with lexicalization information extracted from the KB (semantic type of the entities, domain and range of the relations) or/and using distant supervision from comparable KB/NL data (Elsahar et al., 2018; Liu et al., 2019; Serban et al., 2016). Delexicalization has also been commonly used, where KB entities are replaced with placeholders both in the input and in the output text (see table 13). The model is trained on the delexicalized data and at inference time, post-processing replaces placeholders with the corresponding values (Elsahar et al., 2018; Liu et al., 2019; Serban et al., 2016). Yet even if these approaches have yielded good results, they require extensive, often ad-hoc, pre- and post-processing techniques to be effective, increasing the complexity of the model. Moreover, these additional methods might not be generic enough to scale up to new databases with other schema and broader signature. Delexicalization for instance, which requires matching KB entities (e.g., Barack Obama) in the input with their corresponding NL mentions in the output text (e.g., the former President of the United States) may be quite complex and may also result in incorrect or incomplete delexicalizations when applied to a new KB. Similarly, distant supervision is only possible given some comparable data and might only provide partial information. In fact, (Liu et al., 2019) notes that (Elsahar et al., 2018)'s distant supervision approach only provides textual information for 44% of the predicates present in the SimpleQuestion dataset they use for training. Finally, the presence and coverage of type, domain and range information that are relevant for the generation of NL questions varies depending on the database and might not be sufficient to support the verbalization of unknown entities or relations (i.e., entities and relations which have not been seen at training time).

In recent years, pre-training has been shown to be effective for providing neural models with additional information about the structure of natural language and improving generative tasks (Dong et al., 2019; Song et al., 2019; Lawrence et al., 2019). In this paper, we leverage pre-training to provide a model for KBQG which requires neither delexicalizing the training and test data nor enriching the KB input with additional information. We use BART, a Transformer-based encoder-decoder pre-trained using a denoising objective on large quantities of text, and we propose an approach to the KBQG task which differs from pre-

1.	**Input**	rdf: (HENRI_POINCARE , BIRTHPLACE , FRANCE)	qfocus-pos: obj qtype: which
	Output	Which country was Henri Poincaré born in ?	
2.	**Input**	rdf: (HENRI_POINCARE , BIRTHPLACE , FRANCE)	qfocus-pos: obj qtype: where
	Output	Where was Henri Poincaré born ?	

Figure 1: Input/Output Examples: 1 and 2 show how the same input triple may map to multiple questions with different question types.

vious work in two ways. First, we use the question type (e.g., *what, which, where, when, etc.* see Section 4.2) to guide generation. This helps capture the fact that, as illustrated by Examples 1 and 2 in figure 1, a given KB fact may give rise to multiple questions. Second, we provide a novel dataset for KBQG using Wikidata (Vrandečić and Krötzsch, 2014) as a KB and deriving (KB fact, question) pairs from three existing datasets, namely, SimpleQuestions (Bordes et al., 2015), ZeroShotRE (Levy et al., 2017) and WebNLG (Gardent et al., 2017a). This novel dataset is both more up to date (replacing Freebase which is no longer available with Wikidata which has become one of the largest and most prominent collections of open data on the web) and linguistically richer (contrary to the SimpleQuestions dataset which maps each input to a single question, the dataset derived from the WebNLG data allows for a one-to-many input/question mapping).

We show that our approach outperforms previous approaches in a zero-shot setting for KB properties and entity types (i.e., for KB facts whose property/entity type does not occur in the training data); that additional data increases coverage (more KB properties can be accounted for); and that controlling generation using question type helps improve diversity (one KB triple can be used to produce multiple questions).

2. Related Work

Early work (Olney et al., 2012; Seyler et al., 2015; Song and Zhao, 2016; Seyler et al., 2017) on KBQG used hand-crafted templates which requires significant human effort, generalizes poorly and is difficult to scale up. Recently, neural models have been proposed which are trained on corpora of (KB triple, NL question) pairs and do not require manual intervention. (Reddy et al., 2017) used an RNN sequence-to-sequence model to convert a set of keywords about a Freebase subgraph into a question. Within the Semantic Web community, (Kumar et al., 2019) introduced a neural question generator over knowledge graph where the complexity of the output can be controlled. (Serban et al., 2016) first trained a recurrent encoder-decoder network with attention on SimpleQuestions dataset (Bordes et al., 2015). To handle unseen entities, they used a placeholder for the subject entity in the question and train on the delexicalized data. (Elsahar et al., 2018) focused on generalisation in a zero-shot setting. To handle unseen entities and properties, they enriched the KB input with a lexicalisation of the input KB property obtained through distant supervision and with the Freebase type of the input subject and object entities. They also delexicalized the data by replacing matching terms in this additional information and the output questions with placeholders, replacing these by their value after inference. The RDF triples are initialized with learned TransE (Bordes et al., 2013) embeddings. Two separate encoders are used for the RDF and the textual context and the decoder attends to both. (Liu et al., 2019) expanded the contextual information used by (Elsahar et al., 2018) with information about the domain and the range of the input property. To improve question specificity, they propose an answer-aware loss by optimizing the cross-entropy between the generated question and the answer type words. Finally, (Bi et al., 2020) developed an encoder-decoder question generator, which also enriches the input facts with additional information, and constrain the decoder with word types to preserve the adequacy of the generated question.

These approaches require extensive pre- and post-processing of the input. In contrast, we propose a simpler approach using pre-training to learn the mapping between KB items and their NL counterpart.

3. Task and Terminology

RDF (Rich Description Framework, (Lassila et al., 1998)) is a semantic web standard for encoding knowledge. In an RDF KB, facts are encoded as triples of the form (s, p, o), where s and o are RDF entities (also called resources) and p is a property.

Given an RDF triple of the form (s, p, o), the KBQG task consists in generating an NL question about the object o (or the subject s) of the triple. Some example input and output are shown in figure 1.

We call the questioned part of the RDF triple, the **question focus** and we refer to the semantic type of the question focus which can be extracted from the KB as the **question focus type**. We use the term **question focus position** to refer to the position (subject or object) of the question focus in the RDF triple.

4. The WKDQG Dataset

To evaluate our approach using the BART model, we created WKDQG which is the compilation of three KBQG datasets: SimpleQuestions (Bordes et al., 2015), ZeroshotRE (Levy et al., 2017) and WebNLG-Q. Its creation was motivated by the lack of standard between the three datasets, since each one verbalizes natural questions based on different knowledge-bases, some not even maintained anymore. To unify these datasets, WKDQG maps their knowledge bases and aligns their natural questions to the WikiData format

(Vrandečić and Krötzsch, 2014), currently one of the largest and most prominent collection of open data on the web. Moreover, we also enrich the standardized datasets with additional typing and lexicalization information, not present in the original versions to allow comparison with previous approaches.

In this section, we first describe the creation and content of these three datasets and how they were mapped to Wikidata. We then explain how these datasets were enriched with additional typing and lexicalization information to help guide generation.

4.1. (RDF, Question) Datasets

SQ. SimpleQuestions (SQ, (Bordes et al., 2015)) is a benchmark of KBQG models. It comprises 108K pairs between a triple from Freebase (Bollacker et al., 2008), a KB which is no longer maintained, and an NL question collected by crowdsourcing.

We transformed the original SQ dataset (SQ-FB) into its Wikidata version (SQ hereafter) by mapping its entity-pairs into Wikidata using the P646 property (Free-Base ID). We noticed that a small set of Freebase entities map to more than one entity in Wikidata; to resolve these subject entity ambiguities we used token overlap with the target question.[1]

Next, we identified the set of all Wikidata properties between each entity pair. The entity pairs (in Wikidata format), are then grouped by their Freebase property (FB cluster). First, we consider all the one-to-one relations; if all entity-pairs in a Freebase cluster with Freebase property p^i_{fb} share the same WKD property p^j_{wkd}, we map p^i_{fb} to p^j_{wkd}. If on the other hand not all pairs of the FB cluster are related by the same WKD property,[2] we check whether there exists a Wikidata property shared by at least 75% of the FB cluster. If this is the case, we assign this Wikidata property to all pairs of the FB cluster. Otherwise, we manually inspect the set of Wikidata properties associated with the FB cluster to decide whether to keep (assign one property to the group or from a set of properties for members of the group), or discard the found Wikidata properties.

This was done for the forward direction first, and for the remaining unassigned Freebase entity pairs, we repeated the process in the backward direction. Finally, for the remaining entity-pairs without any Wikidata relations between them, we use the Freebase-to-Wikidata property mappings already found.

We note that in the original corpus, the question focus is always the object of the triple. When converting from FreeBase to Wikidata, it sometimes shifted from being the object of the RDF triple to being the subject. We ensured that the relevant information in such samples are appropriately reversed. The final Wikidata RDF triple is represented by the English-language labels for the entities and property of the triple.

ZQ. ZeroshotRE is a KBQA corpus consisting of several questions generated based on 1,192 crowdsourced question templates (e.g., "Where did x graduate from?", "In which university did x study?" and "What is x's alma mater?") for 120 Wikidata properties. We used the version of it that is included in the KILT benchmark (Petroni et al., 2020), publicly available in the HuggingFace `datasets` library.

Although the dataset is already in the Wikidata format, its test set comprises of only RDF triples, whose question focus is missing. For example, the question 'Which award did Hrant Melkumyan get?' is paired with the incomplete RDF triple (HRANT MELKUMYAN , AWARD RECEIVED , _). To circumvent this, we used SPARQL queries to retrieve the answer set for these questions in Wikidata. For those with more than one possible answer, we instantiated new samples in the corpus.[3].

WQ. The WebNLG dataset (Gardent et al., 2017b) is another popular data-to-text benchmark, with natural language assertions to 3,790 unique RDF triples as well as their combinations. We collected a *data-to-question* version of this corpus, comprising 11,664 NL questions to 3,625 (95.6%) of the single RDF triples of version 2.1 of the original corpus.

The questions were collected using the AMT crowdworking platform. Participants were asked to produce a question for an RDF triple given a question type and a question focus to ensure wide coverage of such questions which is our aim for WKDQG. In terms of question focus, they were given both subject and object parts of the RDF triples to ask for (e.g., (**ALBENNIE_JONES** , ACTIVEYEARSENDYEAR , 1950) → *Which singer closed out her career in 1950?* and (ALBENNIE_JONES , ACTIVEYEARSENDYEAR , **1950**) → *When did Albennie Jones' career come to a close?* respectively) .[4]

As the WebNLG triples come from the DBpedia knowledge base, we mapped their entities and properties into Wikidata using a process similar to that for the SQ dataset. Details are given in the appendices.

[1]For instance, the Freebase entity 08fc1w is linked to two Wikidata entities, Q14798562 and Q153441 as at January 2022. They have entity labels 'Margarita Luti' and 'La fornarina' respectively. The entity Q153441 was selected given its label's uncased word token overlap with the question in the SQ sample: "What artist created "La Fornarina"?"

[2]For example, in the WikiData KB, the FB property `www.freebase.com/music/artist/origin` sometimes map to the Wikidata relation "place of birth" (P19) and sometimes to "location of formation" (P740).

[3]A small set of 217 of these incomplete RDFs (534 samples) remained without answers (from the time ZeroshotRE was created to present, the subject in the RDF no longer holds the property); they are marked "NoAnswer" in the dataset and excluded from our experiments

[4]The question focuses were classified using a coarse-grained set of KB types (Location, Organization, Person, and Event/Date, Measure, Number as well as an Other category) and mapped to the corresponding question types from *What, When, Where, Which, Who, How many*.

	# (RDF,Q) pairs	# qtype/RDF (avg/min/max)	# RDF prop.	# RDF ent.	S/O/S&O	Vocab. Size	Question Size (avg/min/max)
SQ	53,624	1.0/1.0/2.0	196	62,088	0.77/0.20/0.03	16,602	7.96/1.0/36.0
WQ	10,272	2.26/1.0/4.0	184	2,713	0.11/0.71/0.18	6,084	10.08/5.0/42.0
ZQ	282,543	1.22/1.0/4.0	120	193,576	0.74/0.24/0.01	22,970	8.99/4.0/38.0
TOTAL (union)	346,439	1.19/1.0/4.0	342	247,720	0.75/0.22/0.03	30,607	8.86/1.0/42.0

Table 1: Datasets (S/O/S&O denotes the proportion of entities occupying the subject, object or subject as well as object positions of the triples in the data. Vocab. Size is the number of distinct subword-tokens in the NL question part of the data, Question Size is the number of word tokens in each NL question.)

Due to differences in the data models of DBpedia and Wikidata, 412 out of the 3,625 WebNLG original single triples could not be mapped into Wikidata as their properties have no or multiple counterparts in WikiData. Another set of 90 single triples map into a smaller set of 45 Wikidata triples.[5] As a result only 10,272 RDF-Q pairs remained in WQ after the mapping.

4.2. Adding typing and lexicalization information

For all datasets, we enrich each RDF-question pair with question type and the semantic type of its question focus. This information was obtained from the Wikidata public SPARQL endpoint in the second half of 2021. For SQ, we also include the additional information released by (Elsahar et al., 2018).

Question type. SQ, WQ and ZQ contain *What, When, Where, Which and Who* questions, whereas WQ also contains quantity-seeking questions (e.g. 'How many pages is the novel A Long Long Way?'). We detect these questions types with regular expressions/string match.[6] Besides these question types, SQ and ZQ contain 'inform-me' questions (e.g. 'The date of birth of Glyn Pardoe is? and 'Name a modern jazz singer.') and polar questions ('Is highly refined pirates a post-rock album?') as well; in our work, we label these questions as being of the *Other* type.

Question focus type. For SQ, we use the Freebase entity type information contained in the version of the original dataset released by (Elsahar et al., 2018) (see Section 4.2) to allow for a comparison with Elsahar et al's model. For WQ and ZQ, we retrieved the set of Wikidata supertypes for every entity in the dataset from Wikidata using the 'instance of' (P31) and 'subclass of' (P279) properties. If an entity has multiple possible supertypes, we select the one that is the most common across the training split of the dataset for our experiments. Table 1 shows some statistics about each dataset and Table 2 about the question type distribution.

Question type	SQ	WQ	ZQ
What	58.3%	42.3%	57.7%
When	<0.1%	1.0%	3.0%
Where	9.6%	7.7%	1.6%
Which	14.1%	38.3%	20.8%
Who	13.1%	8.5%	15.5%
How many	-	2.2%	-
Other	4.9%	-	1.4%

Table 2: Distribution of question types in the SQ, ZQ and WQ datasets.

SQ additional information. When comparing our approach with (Elsahar et al., 2018) on the SQ dataset, we use the additional lexical information they released. This comprises verbalizations of the RDF property, obtained by distant supervision, as well as the Freebase semantic type of the RDF entities.[7] For SQ instances without such additional information, we used a special token to represent the missing information.[8]

5. Approach

Instead of enriching the input with NL information as was done in previous work, we leverage advances in pre-training and use the WKDQG dataset to adapt the BART pre-trained model to KBQG. BART (Lewis et al., 2020) is a Transformer-based encoder-decoder using sub-word tokenization (byte-pair encoding) and was trained using a generative denoising objective on a combination of news, Wikipedia and books.

In adapting BART for question generation from RDF input, we explore two main options: one where only the RDF is used as input (BART$_{rdf}$) and another (for the SQ dataset only) where the RDF input is enriched with the additional NL information provided as a support for the lexicalization of the RDF content by (Elsahar et al., 2018) (BART$_{rdf+nl}$).

[5] For instance the DBpedia triples (BACON EXPLOSION , MAIN INGREDIENT , BACON) and (BACON EXPLOSION , INGREDIENT , BACON) both map to (BACON EXPLOSION , HAS PART , BACON) in Wikidata.

[6] First with regular expressions at the start of the question, and if no question type word was detected there, a fallback to string match inside the question (for embedded questions, e.g. ''Name an artist who plays rock music.').

[7] The semantic type information for the entities in SQ were obtained by (Elsahar et al., 2018) from the FB5M version of Freebase using the 'fb:type/instance' property.

[8] As indicated in Section 1, Elsahar's distant supervision approach only provides lexicalisation information for 44% of the predicates present in SQ.

We also explore variants regarding the question type: (BART$_{rdf}$), (BART$_{rdf,qt}$) and (BART$_{rdf,mtl}$) which we describe below; Table 13 in the appendices provide examples of the inputs provided to each of these models.

BART$_{rdf}$. This is a BART model without modification which takes as input an RDF triple and a token indicating the question focus position. The RDF is represented linearly in the (s, p, o) order with a special token separator ('|') between each of them.

BART$_{rdf+nl}$. This is the same as BART$_{rdf}$ except that we enrich the input with lexicalization information for the RDF property provided by (Elsahar et al., 2018).[9] We separate this additional information from the rest of the input using a special token.

BART$_{rdf,qt}$. The input to BART$_{rdf,qt}$ is the concatenation of the input RDF triple, a token representing the question focus position, the semantic type of the question focus (e.g., musical artist, location) and a special control token for the target question type. We separated each of these four fields with markers in the input. The addition of the question type information is equivalent to an oracle setting when evaluating on the SQ test set. Our interest in it is motivated by its usefulness for generating varied questions (see Section 7.3).

BART$_{rdf,mtl}$. BART$_{rdf,qt}$ requires that the type of the question that can be generated from an RDF triple be known (since the question type is part of its input). We also explore a setting where this requirement is lifted by using multi-task learning where QG is the main task and predicting the question type is an auxiliary task. The input to both tasks is the concatenation of the triple with the question focus position and the question focus type. The model is trained by minimizing the weighted sum of the loss from the main KBQG task and this auxiliary task. We found that a 0.3 weight for the auxiliary task and 0.7 for the QG task to perform best.

6. Experiments

Training Details For all our experiments, we used the `bart-base` model from the HuggingFace `transformers` library. The `bart-base` model has six layers each in its Transformer encoder and decoder blocks with a hidden size of 768. We used the bart-base tokenizer with a vocabulary size of 50,282 tokens (having added special tokens for the RDF separator, question and question focus types). We fine-tune all of the models by minimizing the standard cross-entropy loss of the outputs against the targets. We use the ADAMW optimizer with a learning rate of 0.0002, and a learning rate scheduler with a linear warm-up of 10% of the training steps. All of the `bart-base` models were trained for 10 epochs, and tuned on the BLEU-4 score of the development set.

Elsahar's Model. We compare our approach with the BART model with (Elsahar et al., 2018), an RNN based encoder-decoder model with attention as well as delexicalization. The model is trained with the delexicalized output, at inference time the decoder output is relexicalized. In the `RDF-only` setting, only information about the RDF (in the form of TransE pretrained embeddings) is provided to the model. Word-based tokenization was used, and word tokens were represented with pretrained 100-D GLOVE embeddings (Pennington et al., 2014). For the RDF triple, pretrained Wikidata embeddings released by (Han et al., 2018)[10] were used to represent the elements of the triples.
In the `RDF+NL` setting, the additional lexicalization information about the property and the entities (see Section 4.1) is added to the RDF input using separate encoders and GLOVE embeddings for the lexicalization part of the input. We used the publicly released code by (Elsahar et al., 2018), making only changes to load the pretrained Wikidata KB embeddings and ensuring that their and our decoders are not constrained by a max length in order to allow comparability.

Automatic Evaluation To evaluate the models' outputs, we used the BLEU-4 (Papineni et al., 2002), ROUGE-L (Lin, 2004) and METEOR (Denkowski and Lavie, 2014) automatic metrics. These n-gram based measures are widely used as indicators of the surface similarity (BLEU-4 and ROUGE-L) and paraphrase (METEOR) between a model's generated output and a reference. We also report the BERTScore (Zhang et al., 2020) metric, which using a BERT model, provides an indicator of the embedding-based semantic similarity between output and reference. To ensure direct comparability on the outputs from the BART model (subword-based tokenization) and Elsahar's (word-based), we applied the MOSES detokenizer on all the models' outputs and references, as well as a set of regular expression rules on Elsahar's model outputs to detokenize contractions and possessives (e.g. don't) not handled by the MOSES detokenizer, before scoring with the automatic metrics.

Human Evaluation We also conducted a human evaluation to assess to what extent the question type control token and the variability present in the WQ dataset (one triple can be mapped to multiple questions with different question type) can help generate questions of different types from the same triple. In this evaluation, we compare our best model trained on SQ only with the same model trained on all three datasets (SQ,WQ,ZQ). Our hypothesis is that WQ variability will help the second model learn to generate different types of questions for the same triple. We collect

[9] For e.g. the RDF (ADLER SCHOOL OF PROFESSIONAL PSYCHOLOGY , COUNTRY , UNITED STATES OF AMERICA) and type information for the question focus, we reused the lexicalization "is location of" from (Elsahar et al., 2018) and inserted the entities in their correct argument position to give "United States of America is location of Adler School of Professional Psychology".

[10] `http://139.129.163.161/index/toolkits#pretrained-embeddings`

from the output of both models 50 randomly selected input triples covering different properties. We show the annotators the input triple, the reference question and the output of the two models, and we ask them which of the two outputs verbalizes the input property best (A: Adequacy); differs most from the reference question (D:Difference); is most natural (N:Naturalness) and verbalizes the entities best (E:Entity). A third option is possible for cases where both output score similarly. We measure the percentage of time one model was chosen over the other, taking the majority agreement between three evaluators.

7. Results and Discussion

We first compare our approach with Elsahar's on SQ considering four settings: with and without NL information, on seen data (RDF properties present in the test data have been seen at training time), and in a property zero-shot setting (RDF properties present in the test data have not been seen at training time). We also examine the seen and zero-shot settings for entity types. We then examine the impact of the additional datasets in particular, the WQ dataset which allows for the same triple to be mapped to multiple questions with different question types.

7.1. With and without additional NL information on Seen data

Table 3 shows the results for the standard setting, where RDF properties present in the test data have been seen at training time.

Pre-training helps bridge the gap between RDF properties and their lexicalization. Our approach outperforms Elsahar's in both settings (with and without additional lexicalization information in the input). Moreover, BART$_{rdf}$, which uses no additional information, yield comparable results to Elsahar$_{nl}$'s model which uses additional NL information. This indicates that pre-training and word pieces suffice to bridge the gap between the name of the RDF properties and the way they are lexicalized in text. It also shows that despite the high ratio of named entities in RDF triples (the subject and object, which make up two thirds of an RDF triple, usually are named entities), delexicalization (used by Elsahar's) can successfully be replaced by these two methods: both pre-training and word pieces help the model generate names that might not have been seen at training time.

Question type helps improve performance. Whether it is implicitly learned through multi-task learning (BART$_{rdf,mtl}$, BART$_{rdf+nl,mtl}$) or explicitly input to the model (BART$_{rdf,qt}$, BART$_{rdf+nl,qt}$), informing the model with the target question type improves performance.

7.2. Zero-Shot Learning

We used the same cross validation approach as (Elsahar et al., 2018) to approximate a zero-shot property setting. Specifically, we split the SQ dataset into 10 folds, with mutually exclusive sets of RDF properties (no fold contains RDF properties found in another fold) and draw two of these folds in turn for use as the test set in each cross-validation run. At the start of each run, we reload the original parameter weights for the pre-trained BART model. Following Elsahar, we repeat the same zero-shot setting on entity types (which is stricter than a zero-shot entity set-up), taking care to account for the fact that a SimpleQuestions triple in Freebase may have a different order when mapped to Wikidata. Table 4 includes the mean and standard deviation of the automatic metrics from cycling through these data splits for a zero-shot property setting. Table 5 shows the same but for the zero-shot entity type settings.

Model	B-4	BSc	R-L	M
RDF-only				
Elsahar	34.01	64.85	61.51	32.67
BART$_{rdf}$	37.05	69.42	65.12	34.22
BART$_{rdf,mtl}$	37.91	69.68	65.20	34.55
BART$_{rdf,qt}$	**41.95**	**73.51**	**71.21**	**36.78**
RDF+NL				
Elsahar$_{nl}$	38.13	68.63	65.48	34.74
BART$_{rdf+nl}$	38.38	70.00	65.67	34.87
BART$_{rdf+nl,mtl}$	38.10	70.17	65.57	34.73
BART$_{rdf+nl,qt}$	**42.67**	**73.78**	**71.50**	**37.28**

Table 3: Results on the SQ dataset under a **SEEN** setting, i.e. no zero-shot constraints (B-4: BLEU-4, BSc: BertScore, R-L: Rouge-L and M: Meteor).

Pre-training outperforms a delexicalization model whose input is enriched with lexicalization information. Regardless of a zero-shot property or zero-shot entity type setting, our approach outperforms Elsahar's whether or not the input is enriched with lexicalization information. Notably, the BART model **without** NL information (BART$_{rdf}$) performs on par with Elsahar's model **with** NL information (Elsahar$_{nl}$). This illustrates the capacity of pretrained decoders based on subword units to handle unseen units: while the RDF properties of the test data have not been seen at training time, their subword units probably have been and can be used by the decoder to generate the corresponding NL expressions. There is however, a 10-BLEU point gap between the zero-shot property and zero-shot entity type settings for the top performing model (BART$_{rdf+nl,qt}$), indicating the importance of lexicalisation data for KB properties.

7.3. Additional data

The main contribution of the extended dataset WKDQG (in particular WQ) is that it helps generate multiple questions from the same KB triple. In SQ, each RDF is only paired with a question of a single type as well as a single question focus (either the subject or the object). Accordingly, a model trained (and evalu-

Model	B-4	BSc	R-L	M
RDF-only				
Elsahar	14.24	47.94	44.30	24.43
	(±2.48)	(±2.23)	(±2.66)	(±0.92)
BART$_{rdf}$	22.63	59.12	53.14	27.02
	±2.85	±2.55	±2.64	±1.35
BART$_{rdf,mtl}$	26.63	62.24	56.52	28.91
	±3.03	±1.20	±2.21	±1.19
BART$_{rdf,qt}$	**28.35**	**64.40**	**60.84**	**30.46**
	±3.33	±2.98	±2.67	±1.62
RDF+NL				
Elsahar$_{nl}$	20.54	55.73	51.11	27.32
	(±3.68)	(±2.34)	(±2.96)	(±1.45)
BART$_{rdf+nl}$	23.42	59.52	53.74	27.46
	± 3.38	±2.66	±2.73	±1.39
BART$_{rdf+nl,mtl}$	25.25	61.29	55.04	28.27
	±3.35	±2.49	±2.42	±1.34
BART$_{rdf+nl,qt}$	**28.74**	**64.18**	**60.62**	**30.38**
	±3.38	±3.27	±2.90	±1.45

Table 4: Results on the SQ dataset under a **zero-shot** setting for RDF properties.

Model	Sub-type		Obj-type	
	B4	R-L	B4	R-L
RDF-only				
Elsahar	29.96	58.46	23.94	53.54
	(±2.10)	(±2.29)	(±4.34)	(±3.23)
BART$_{rdf}$	32.90	61.59	30.40	60.05
	(±1.90)	(±1.79)	(±3.09)	(±2.34)
BART$_{rdf,mtl}$	33.21	62.11	31.07	60.49
	(±1.50)	(±1.74)	(±2.65)	(±2.33)
BART$_{rdf,qt}$	**37.30**	**67.48**	**35.05**	**66.11**
	(±1.68)	(±1.38)	(±3.03)	(±1.97)
RDF+NL				
Elsahar$_{nl}$	32.92	61.43	28.58	58.42
	(±2.77)	(±1.91)	(±4.48)	(±2.98)
BART$_{rdf+nl}$	34.23	62.29	30.96	59.87
	(±2.33)	(±2.28)	(±3.27)	(±3.02)
BART$_{rdf+nl,mtl}$	34.32	62.31	31.24	60.13
	(±2.01)	(±1.98)	(±3.23)	(±2.55)
BART$_{rdf+nl,qt}$	**38.51**	**68.08**	**35.76**	**66.62**
	(±1.68)	(±1.51)	(±3.12)	(±2.13)

Table 5: Results on the SQ dataset under a **zero-shot** setting for RDF entities (subject/object for Elsahar, question focus and the other entity in the triple for the BART models).

ated to perform well) on SQ data alone is unlikely to be able to generate questions of a different form (paraphrased, or with a different question type). On the other hand, although WQ is five times smaller in size, it has a wide coverage of question types for each RDF in it and there is also variation in the question focus. While the questions in ZQ were instantiated from templates and their question focus are all on the object of the RDF, all of its questions are of a high quality in terms of specificity. We included them in WKDQG for these reasons. Using the additional parallel data created with WKDQG we also explored different ways of combining it (training on all data or training and fine-tuning) but did not find it to improve results over training and testing on each of the three datasets (cf. Table 12 in the appendices) likely because the datasets have very different properties in terms of question type/input ratio and vocabulary size.

Model — Metric	B-4	BSc	R-L	M
$Test_O$				
BART$_{rdf,qt}$	41.95	73.51	71.21	36.78
BART$_{rdf,qt,wkdqg}$	41.31	72.63	70.28	36.62
$Test_A$				
BART$_{rdf,qt}$	26.60	60.15	49.53	29.27
BART$_{rdf,qt,wkdqg}$	26.37	59.48	49.29	29.30

Table 6: Results on the SQ dataset under a **SEEN** setting. BART$_{rdf,qt,wkdqg}$: model fine-tuned on the WKDQG data. Test$_A$ (for alternative) is the SQ test set with a different question type provided to the model. Test$_O$ (for original) is the SQ test set with the original question type.

Choice — Measure	D	A	N	E
BART$_{rdf,qt}$Test$_0$	14%	12%	18%	2%
BART$_{rdf,qt,wkdqg}$Test$_A$	76%	4%	10%	4%
Same	8%	80%	62%	92%
No Majority Vote	2%	4%	10%	2%

Table 7: Human evaluation on 50 outputs (D:Difference from the reference, A:Semantically Adequate, N:Naturalness, E:Entity Lexicalizations). For each criteria, the first two lines of the columns indicate which model is preferred. E.g., BART$_{rdf,qt,wkdqg}$'s output is judged more different from the reference 76% of the time than BART$_{rdf,qt}$'s.

Finetuning with varied data permits generating questions with different question type from the same triple. Using SQ as test set, we show that BART$_{rdf,qt}$ fine-tuned on the WKDQG data (BART$_{rdf,qt,wkdqg}$) is able to generate questions that are paraphrases of the reference. We do this by replacing the question type in the input with one for a different, but semantically plausible, question type.[11] We found that in this setting (Test$_A$), BART$_{rdf,qt,wkdqg}$ is able to almost always faithfully generate a question of this new type: the model output only deviated from the provided question type seven times (out of the 10,725

[11]We used a BERT model to predict the distribution of question types given an entity type and a question focus position as input. This model is trained on data from the train and development sets of SQ, WQ and ZQ.

instances in the test set).[12] Tables 6 (automatic evaluation) and 7 (human evaluation) show the results of this experiment, and Table 10 in the appendices contains examples of the generated outputs here. While the automatic metrics show significantly lower scores for the $\texttt{Test}_A$ setting since the question type of the generated questions are different from the reference, the human evaluation indicates that the $\text{BART}_{rdf,qt,wkdqg}$ model generates questions with a comparable level of adequacy (A), naturalness (N) and entities (E) than in a $\texttt{Test}_O$ setting but a greater difference with the reference. The agreement among the three annotators was 0.521 (Fleiss' kappa). This demonstrates that by controlling for the question type and using training data with greater variability, KBQG models can be used to generate varied questions of high quality which differ from the reference.

Model	B-4	BSc	R-L	M
$\text{BART}_{rdf,qt}$	41.95	73.51	71.21	36.78
-question type	37.73	69.72	65.41	34.51
-question focus	41.43	73.17	70.81	36.54

Table 8: Ablation Study: each line indicates the (non cumulative) removal of the corresponding component from $\text{BART}_{rdf,qt}$on the SQ dataset

7.4. Downstream QA Evaluation

We evaluated the utility of our generated varied questions on the performance of two downstream QA systems – KEQA (Huang et al., 2019) and BuboQA (Mohammed et al., 2018), leveraging the approach and code of (Han et al., 2020). While we generate questions using Wikidata triples to enrich the SQ dataset, all of the QA experiments described here use Freebase data. The results of these experiments are summarised in Table 9 here and details are provided in Table 11.

Using the same $\texttt{Test\_}A$ set as in Section-7.3 while leaving the train and development sets unchanged, we show that simply changing the distribution of the question types at inference time results in a 3.5% drop in top-1 accuracy (see **SQ_w1** and **SQ_w2** in Table 11).

We were able to reverse this drop in performance on $\texttt{Test\_}A$ by using an enriched training and development set. We do this using the same question type prediction model above,[13] and using the obtained set of plausible question type tokens as controls for the BART model, we generated the set of paraphrased questions of (different question types) for each SQ sample.

We also show that this enrichment approach enables robust QA performance – in the face of a shift in the distribution of the question types in the test data. The same models trained on the enriched training and validation set, perform as well on the original test set (i.e. $\texttt{Test\_}O$ in Section-7.3, compare (**SQ_o+e**:**SQ_o** and **SQ_w3**:**SQ_w0**). Consistent with the findings of (Liu et al., 2019), we find that enriching the training data with generated questions leads to a minor decline ($\approx$ 0.5 percentage points) in top-1 accuracy.

Model	**OOO**	**OOA**	**EEA**	**EEO**
BuboQA Acc @ 1	**85.12**	81.42	85.08	84.57
KEQA Acc @ 1	**86.85**	81.15	83.79	86.44

Table 9: Results of QA systems with and without questions generated with our approach. The column headers denote the train-dev-test set compositions. **O** denotes original, **A** alternative and **E** enriched sets of questions.

7.5. Ablation

We also performed an ablation study using the SQ dataset (cf. table 8) and find that removing the question focus from the input has limited negative impact (-0.52 BLEU-4) but that removing the question type control token leads to a strong decrease in performance across all automatic metrics (-4.22 BLEU-4). This indicates that the benefits brought about by pretraining, through knowledge about the question focus embedded in the model, is limited relative to question type information, at least for the goal of generating questions faithful to the type in the reference.

8. Conclusion

We revisited the task of KBQG and introduced a novel approach of generating questions from KB triples using a fine-tuned large language model, without the ad-hoc processing required of earlier work. Experimental evidence on WKDQG reveals that pre-training and question type control contributes to improved performance both in a standard and in a zero-shot setting. We investigate the capacity of the BART model and extended dataset to generate questions whose type is distinct from that of the gold truth, and find that it leads to better or similar quality questions of varied types in our human evaluation. Additionally, we quantified the decline in current QA systems' performance at inference time when the question type distribution is shifted from that seen at training, and show that by enriching the training data with the set of possible questions generated by our approach, these systems' performances are restored. We hope that WKDQG – which extends an existing benchmark (SQ) by six-fold, introduces a wider coverage of properties and question type per triple, and is updated to an actively maintained KB – contributes towards advances in QG and QA in general.

[12] In all seven cases, the new question type provided to the model was 'Other' and the generated questions were of the 'What' (4) and 'Which' (1), and 'Where' (2) types.

[13] Except that, instead of picking a single question type for each sample, we returned the set of all possible question types capped at four (including the original).

9. Acknowledgements

This research was supported by ANR Project QUANTUM (Project-ANR-19-CE23-0025). Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see `https://www.grid5000.fr`).

10. Bibliographical References

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Bi, S., Cheng, X., Li, Y.-F., Wang, Y., and Qi, G. (2020). Knowledge-enriched, type-constrained and grammar-guided question generation over knowledge bases. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2776–2786, Barcelona, Spain (Online), December. International Committee on Computational Linguistics.

Bollacker, K. D., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In Jason Tsong-Li Wang, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In C.J. Burges, et al., editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Bordes, A., Usunier, N., Chopra, S., and Weston, J. (2015). Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Denkowski, M. and Lavie, A. (2014). Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA, June. Association for Computational Linguistics.

Dong, R., Smith, D., Dudy, S., and Bedrick, S. (2019). Noisy neural language modeling for typing prediction in BCI communication. In *Proceedings of the Eighth Workshop on Speech and Language Processing for Assistive Technologies*, pages 44–51, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Elsahar, H., Gravier, C., and Laforest, F. (2018). Zero-shot question generation from knowledge graphs for unseen predicates and entity types. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 218–228, New Orleans, Louisiana, June. Association for Computational Linguistics.

Gardent, C., Shimorina, A., Narayan, S., and Perez-Beltrachini, L. (2017a). Creating training corpora for NLG micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188, Vancouver, Canada, July. Association for Computational Linguistics.

Gardent, C., Shimorina, A., Narayan, S., and Perez-Beltrachini, L. (2017b). The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain, September. Association for Computational Linguistics.

Han, X., Cao, S., Lv, X., Lin, Y., Liu, Z., Sun, M., and Li, J. (2018). OpenKE: An open toolkit for knowledge embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 139–144, Brussels, Belgium, November. Association for Computational Linguistics.

Han, N., Topic, G., Noji, H., Takamura, H., and Miyao, Y. (2020). An empirical analysis of existing systems and datasets toward general simple question answering. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5321–5334, Barcelona, Spain (Online), December. International Committee on Computational Linguistics.

Huang, X., Zhang, J., Li, D., and Li, P. (2019). Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, page 105–113, New York, NY, USA. Association for Computing Machinery.

Kumar, V., Hua, Y., Ramakrishnan, G., Qi, G., Gao, L., and Li, Y.-F. (2019). Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer.

Lassila, O., Swick, R. R., et al. (1998). Resource description framework (rdf) model and syntax specification.

Lawrence, C., Kotnis, B., and Niepert, M. (2019). Attending to future tokens for bidirectional sequence generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1–10, Hong Kong, China, November. Association for Computational Linguistics.

Lenat, D. and Guha, R. (1993). Building large knowledge-based systems: Representation and inference in the cyc project. *Artificial Intelligence*, 61(1):4152.

Levy, O., Seo, M., Choi, E., and Zettlemoyer, L. (2017). Zero-shot relation extraction via reading

comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada, August. Association for Computational Linguistics.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July. Association for Computational Linguistics.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July. Association for Computational Linguistics.

Liu, C., Liu, K., He, S., Nie, Z., and Zhao, J. (2019). Generating questions for knowledge bases via incorporating diversified contexts and answer-aware loss. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2431–2441, Hong Kong, China, November. Association for Computational Linguistics.

Mohammed, S., Shi, P., and Lin, J. (2018). Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 291–296, New Orleans, Louisiana, June. Association for Computational Linguistics.

Olney, A. M., Graesser, A. C., and Person, N. K. (2012). Question generation from concept maps. *Dialogue & Discourse*, 3(2):75–99.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Petroni, F., Piktus, A., Fan, A., Lewis, P., Yazdani, M., De Cao, N., Thorne, J., Jernite, Y., Karpukhin, V., Maillard, J., et al. (2020). Kilt: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252*.

Reddy, S., Raghu, D., Khapra, M. M., and Joshi, S. (2017). Generating natural language question-answer pairs from a knowledge graph using a RNN based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 376–385, Valencia, Spain, April. Association for Computational Linguistics.

Serban, I. V., García-Durán, A., Gulcehre, C., Ahn, S., Chandar, S., Courville, A., and Bengio, Y. (2016). Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598, Berlin, Germany, August. Association for Computational Linguistics.

Seyler, D., Yahya, M., and Berberich, K. (2015). Generating quiz questions from knowledge graphs. In *Proceedings of the 24th International Conference on World Wide Web*, pages 113–114.

Seyler, D., Yahya, M., and Berberich, K. (2017). Knowledge questions from knowledge graphs. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pages 11–18.

Song, L. and Zhao, L. (2016). Question generation from a knowledge base with web exploration. *arXiv preprint arXiv:1610.03807*.

Song, Y., Jiang, D., Zhao, W., Xu, Q., Wong, R. C.-W., and Yang, Q. (2019). Chameleon: A language model adaptation toolkit for automatic speech recognition of conversational speech. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 37–42, Hong Kong, China, November. Association for Computational Linguistics.

Vrandečić, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2020). Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A. Appendices

A.1. Mapping the WebNLG DBPedia triples to Wikidata

The approach for the migration is similar to that for SQ. We leveraged the "schema:about" property to map DBpedia entities into Wikidata. Specifically, we started by constructing SPARQL queries to return the set of all properties between each entity-pair (where both subject and object are entities as well as where the object is a literal in the forward and backward directions).

For the set of entity-pair/entity-literal pair with at least one Wikidata property found between them, we group them by their original DBpedia property (we refer to one of these as a DBP property cluster). Given WQ's smaller size, we directly identified the most common Wikidata property found in each DBP property cluster. We used a similar manual inspection approach as SQ above (namely, manually inspect: (a) one entity-pair from the cluster, (b) its DBpedia triple; and (c) the Wikidata property label.) and only mapped the DBpedia property to this most common Wikidata property if (a), (b) and (c) above are semantically aligned. This process was also repeated in the backwards direction. Next we used the mapping from these assignments (i.e. complete WQ triples found in Wikidata) for all other instances the set of their entities and properties appear in WQ.

For the remaining unmapped WQ DBpedia properties, we wrote a SPARQL query with a contain function for the DBpedia property (with camel case removed and lowercased) on the English label and alternative labels for all the properties in Wikidata (approx 8,000) to identify candidate mappings. We manually inspected these candidates and assigned using the same criteria (steps a,b,c above) if it is semantically aligned with the DBpedia property. We took care to check the direction of the Wikidata property and reversed entity-pair/entity-literal-pairs that have this DBpedia property. A further set of properties in WQ ($<$25) remained unmapped after this and a manual search of the Wikidata site was done to identify a suitable mapping.

For the remaining WQ triples not fully mapped (but with their DBpedia property mapped), we used "schema:about" to map their entities into Wikidata to complete the triple's mapping, with a fallback to search for matches of the entity (replacing the underscores with spaces and maintaining case) against the English labels and alternative labels for all entities in Wikidata. In the latter case, where there are multiple candidates, we select the first matched Wikidata entity (Q-code sorted by lexicographic order). Finally, for remaining unmapped entities, we conducted manual searches (with the help of the `wikipedia` package to identify candidates) on the Wikipedia and Wikidata sites to attempt to map these entities. Unlike SQ above - where we only mapped the triple into Wikidata if both entities in the SQ triple are found in Wikidata, we accepted DBpedia entities that could not be found in Wikidata; for these, we removed the camel case of the DBpedia entity and used them as the entity's mapping. Although the entity may not currently be present in Wikidata, its presence in DBpedia suggests that it is attested in Wikipedia and could be added to Wikidata.

A.2. Example outputs with varied question type control

	SimpleQ Sample	
1.	Reference	what category of celestial object is 7624 gluck
(O)	Input	(7624 GLUCK , INSTANCE OF , ASTEROID), **WHAT**, ANSOBJ, category
	Generated	what type of celestial object is 7624 gluck
(A)	Input	(7624 GLUCK , INSTANCE OF , ASTEROID), **WHICH**, ANSOBJ, category
	Generated	which type of celestial object is 7624 gluck
2.	Reference	what is maurizio calvesi's profession
(O)	Input	(MAURIZIO CALVESI , OCCUPATION , CINEMATOGRAPHER), **WHAT**, ANSOBJ, profession
	Generated	what is maurizio calvesi's profession
(A)	Input	(MAURIZIO CALVESI , OCCUPATION , CINEMATOGRAPHER), **OTHER**, ANSOBJ, profession
	Generated	is maurizio calvesi a cinematographer or a technician
3.	Reference	who was born in compton
(O)	Input	(CLARENCE DUREN , PLACE OF BIRTH , COMPTON), **WHO**, ANSSUBJ, american football player
	Generated	who was born in compton, queens
(A)	Input	(CLARENCE DUREN , PLACE OF BIRTH , COMPTON), **WHAT**, ANSSUBJ, american football player
	Generated	what former football player was born in compton, illinois

Table 10: Examples of the outputs of BART$_{rdf,qt}$(O) on Test$_O$ compared with those of BART$_{rdf,qt,wkdqg}$on Test$_A$ for the same sample (except for a different question type provided to the model). In bold face are the question type markers provided to the model.

A.3. Downstream evaluation on QA systems

The results of our experiments using generated questions from our approach to enrich the SQ dataset on current QA systems can be found in Table 11 below.

A.4. Combining the three datasets

Table 12 contains the results of our experiments where we fine-tune on the combination of the three datasets (SQ, WQ and ZQ) and evaluate the model on each's test set.

A.5. Examples: models' inputs and generation outputs

Table 13 provides an overview of the format of the inputs to each of the models (Elsahar and ours), under various settings. The original SQ input and reference is provided, as well as examples of each model's generated output.

Split/Model	**SQ_o**	**SQ_o+e**	**SQ_w0**	**SQ_w1**	**SQ_w2**	**SQ_w3**
Train	O, (75,722)	O+E, (173,063)	O_w, (37,521)	O_w, (37,521)	O_w+E, (149,710)	O_w+E, (149,710)
Dev	O, (10,815)	O+E, (24,664)	O_w, (5,360)	O_w, (5,360)	O_w+E, (21,380)	O_w+E (21,380)
Test	O, (21,687)	O, (21,687)	O_w, (10,726)	O_w-A, (10,726)	O_w-A, (10,726)	O_w (10,726)
BuboQA Acc@1	74.63	74.03	**85.12**	81.42	85.08	84.57
KEQA Acc@1	75.30	74.76	**86.85**	81.15	83.79	86.44

Table 11: Results of QA system performance on SQ with and without generated varied questions in the train, dev and/or test sets. In the table above, **O** denotes use of original SQ questions, **E** denotes enrichment (of O) with generated varied questions in train and dev, **A** denotes a question of an alternative (to O's) question type for each sample in test. **_w** denotes the set of SQ samples successfully mapped to Wikidata. In brackets are the sizes of the dataset splits.

Model	SQ	WQ	ZQ	ALL
BLEU-4				
BART$_{rdf,qt}$	41.95	40.55	49.72	48.24
BART$_{rdf,qt,wkdqg}$	41.31	37.60	49.71	48.05
BERTScore				
BART$_{rdf,qt}$	73.51	68.92	75.72	75.17
BART$_{rdf,qt,wkdqg}$	72.63	68.06	75.57	74.89
ROUGE-L				
BART$_{rdf,qt}$	71.21	63.86	71.26	71.03
BART$_{rdf,qt,wkdqg}$	70.28	62.80	71.01	70.65
METEOR				
BART$_{rdf,qt}$	36.78	34.95	40.78	39.99
BART$_{rdf,qt,wkdqg}$	36.62	33.40	40.59	39.76

Table 12: Results (automatic metrics) for RDF-only models. BART$_{rdf,qt,wkdqg}$: model fine-tuned on the WKDQG data, and evaluated on each of the SQ, WQ and ZQ test sets. ALL shows the results proportionally averaged on the three test sets.

SimpleQ Sample	**Input:** (M/03Y2SVR , ASTRONOMY/CELESTIAL_OBJECT/CATEGORY , M/0JVQ) **Reference** what category of celestial object is 7624 gluck		
Model	**Training/inference input format**	**Training target format**	**Generated**
Elsahar	(M/03Y2SVR , ASTRONOMY/CELESTIAL_OBJECT/CATEGORY , M/0JVQ), celestial object, category, PLACEHOLDEROBJ designated as PLACEHOLDERSUB	what OBJTYPE of SUBTYPE is PLACEHOLDERSUB	what is 7624 gluck (relexicalised from system decoder output: what is PLACEHOLDERSUB)
BART$_{rdf}$	(7624 GLUCK , INSTANCE OF , ASTEROID), ANSOBJ	what category of celestial object is 7624 gluck	what type of celestial object is 7624 gluck
BART$_{rdf,mtl}$	(7624 GLUCK , INSTANCE OF , ASTEROID), ANSOBJ, category	ditto	what type of celestial object is (7624) 1979 gd1
BART$_{rdf,qt}$	(7624 GLUCK , INSTANCE OF , ASTEROID), WHAT, ANSOBJ, category	ditto	what type of celestial object is 7624 gluck
BART$_{rdf+nl}$	(7624 GLUCK , INSTANCE OF , ASTEROID), ANSOBJ, asteroid designated as 7624 Gluck	ditto	what kind of celestial body is 7624 gluck
BART$_{rdf+nl,mtl}$	(7624 GLUCK , INSTANCE OF , ASTEROID), ANSOBJ, category, asteroid designated as 7624 Gluck	ditto	what type of celestial body is 7624 gluck
BART$_{rdf+nl,qt}$	(7624 GLUCK , INSTANCE OF , ASTEROID), WHAT, ANSOBJ, category, asteroid designated as 7624 Gluck	ditto	what kind of celestial object is 7624 gluck
BART$_{rdf,qt}$, zero-shot property	(7624 GLUCK , INSTANCE OF , ASTEROID), WHAT, ANSOBJ, category	ditto	what is the category of 7624 gluck
BART$_{rdf+nl,qt}$, zero-shot property	(7624 GLUCK , INSTANCE OF , ASTEROID), WHAT, ANSOBJ, category, asteroid designated as 7624 Gluck	ditto	what is the category of 7624 gluck

Table 13: Examples of system outputs (**Generated**), used in the automatic evaluation. Other columns compare the formats for the input and target between the models (Elsahar and ours).