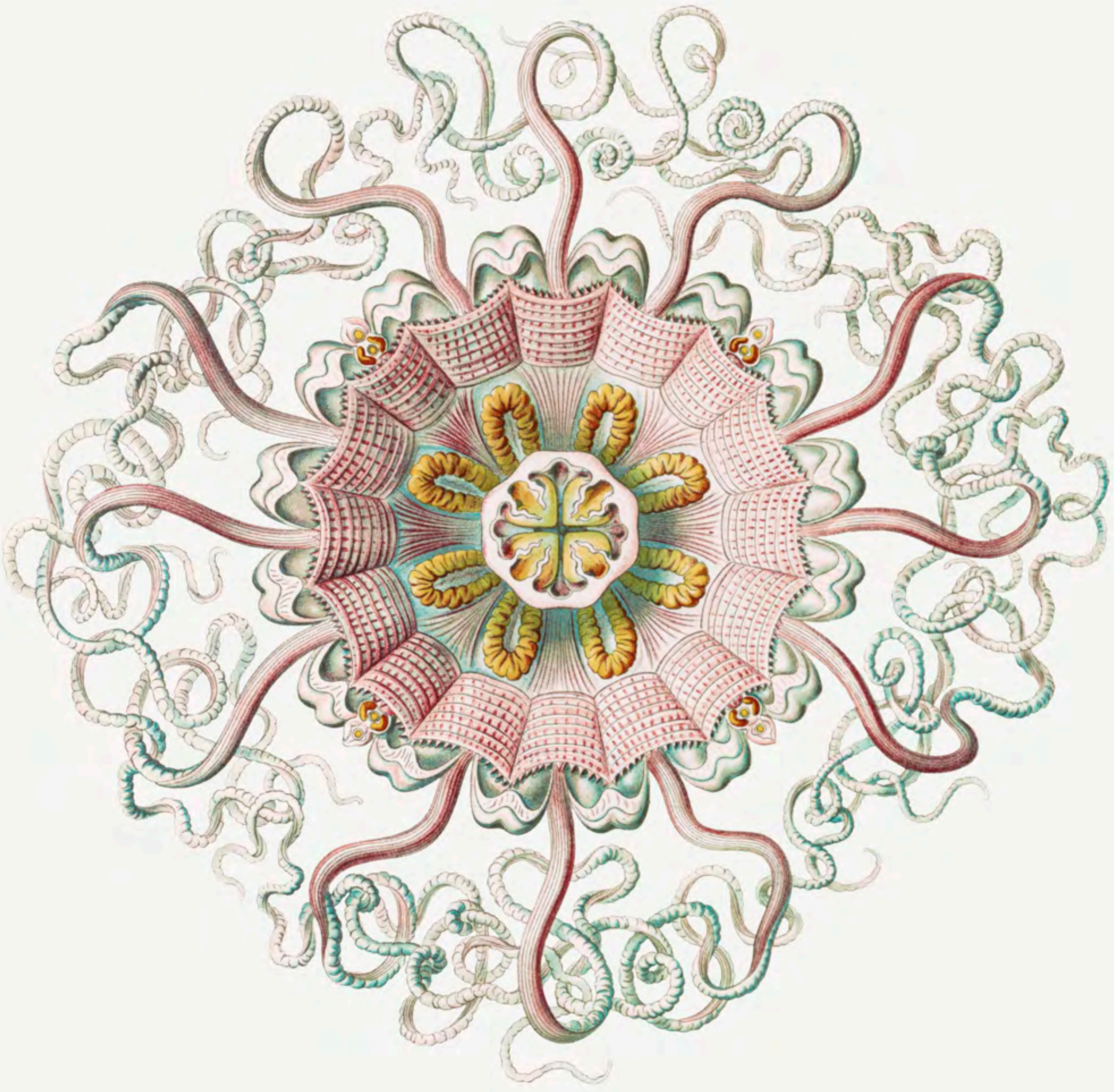
TROND LINJORDET
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

A Critical Look at the Evaluation of Knowledge Graph Question Answering

PhD Thesis — Computer Science — November 2022



A Critical Look at the Evaluation of Knowledge Graph Question Answering

by

Trond Linjordet

A dissertation submitted in fulfilment of
the requirements for the degree of
PHILOSOPHIAE DOCTOR (PhD)

University of
Stavanger

Faculty of Science and Technology
Department of Electrical Engineering and Computer Science
November 18, 2022

University of Stavanger
N-4036 Stavanger
NORWAY
www.uis.no

ISBN 978-82-8439-130-4
ISSN 1890-1387

PhD Thesis UiS no. 671

Acknowledgements

My work during this PhD would not have been possible without many types of support from many people; more than I can thank individually in a reasonable amount of space. The following then, is necessarily a non-exhaustive list of people who deserve special thanks. For those unfairly left out I assert: I am short on memory, not gratitude.

First of all, I wish to thank my supervisor Prof. Krisztian Balog for being a great collaborator and academic role model. I remain very happy with my selection of Krisztian as my PhD supervisor! I especially appreciate the personable but effective style of communication, as well as our shared sense of humor.

In addition, I wish to thank my co-supervisors Prof. Kjersti Engen and Prof. Trygve Eftestøl for their support.

At UiS, I have enjoyed good conversations with, and kindness from, every member of the Department of Electrical Engineering and Computer Science and many employees outside the department, as well. In particular I wish to thank the head of department, Prof. Tom Ryen for being able to care so much for so many people. I also want to thank Prof. Chunming Rong, for bringing me into the fold many years ago.

Thanks to members and visitors of IAI, past and present, with whom I've had the good fortune to interact: Shuo, Darío, Faegheh, Vinay, Martin, Magnus, Rahul, Ivica, Jan, Christin, Nolwenn, Weronika, Ivan.

Other good friends and colleagues at UiS who deserve my thanks include: Kaja, Russel, Sven Ole, Monika, Tomasz, Damiano, Theo, Aryan, Antorweep, Ståle, Julian, Nejm, Anna, Luca, Prachi, Hein, Tormod, Kristian, Morten, Karl, John Håkon, Romuald, Per, Jon, Mina, Faraz, Arian & Yeganeh, Daniel, Anissa, Nan, Boyu, Jakob, Ahmad, Jorge, Thilina, Neel, Didrik, Saul, Svein, Eivind, Racin, Rodrigo, Nikita, Dhanya, Rituka, Jayachander, and many more.

In the autumn of 2018, I was able to visit the research group of Prof. Maarten de Rijke, my “academic grandparent,” which was an inspiring and impactful experience, and for which I am very grateful. I am thankful to many whom Maarten’s generous hospitality allowed me to meet: Rolf, Ana, Maartje, Mostafa, Hosein, Harrie, Nikos, Yifeng, Dan, Petra, Marzieh & Hamid, and again, many more.

I wish to thank friends outside of (this part of) academia, geographically near and far, for their support during these years: Anders B., Martin S., Anders D., Jerome, Magnus H., Tommaso, Sashko, Pål, Christopher, Frode, Diana, and many, *many* more.

Finally but most importantly, I wish to give great thanks to my family for their love and support throughout this journey: my parents Arne and Arnhild, my sister Wivek, and my nieces and nephew Sigrid, Vinjar, and Tiril.

Contents

Chapter 1

Introduction

The field of information retrieval (IR) is concerned with systems that "make a given stored collection of information items available to a user population" [111]. The way in which information is made available to the user depends on the formulation of this broad concern of IR into specific *tasks* by which a system should address a user's information need [85]. The specific IR task also dictates how the user may express their information need.

The classic IR task is *ad hoc retrieval*, where the user issues a query to the system and gets in return a list of documents ranked by estimated relevance of each document to the query [85]. However, it has long been acknowledged that users are often looking for answers to questions, rather than an entire document or ranked list of documents [17, 141]. *Question answering* (QA) is thus another IR task; it comes in many flavors, but overall consists of taking in a user's natural language (NL) question and returning an answer.

This thesis describes work done within the scope of the QA task. The flavor of QA called *knowledge graph question answering* (KGQA) is taken as the primary focus, which enables QA with factual questions against structured data in the form of a *knowledge graph* (KG). This means the KGQA system addresses a structured representation of knowledge rather than—as in other QA flavors—an unstructured prose context. KGs have the benefit that given some identified entities or predicates, all associated properties are available and relationships can be utilized. KGQA then enables users to access structured data using only NL questions and without requiring formal query language expertise.

Even so, the construction of satisfactory KGQA systems remains a challenge. Machine learning with deep neural networks (DNNs) is a far more promising approach than manually engineering retrieval models [29, 56, 130]. The current era dominated by DNNs began with seminal work on computer vision, where the deep learning paradigm demonstrated its first cases of "superhuman" performance [32, 71]. Subsequent work in other applications has also demonstrated "superhuman" performance with DNNs [58, 87]. As a result of its early position and hence longer history as a leading application of deep learning, computer vision with DNNs has been bolstered with much work on different approaches towards augmenting [120] or synthesizing [94] additional training data. The difficulty with machine learning approaches to KGQA appears to rest in large part with the limited volume, quality, and variety of available datasets for this task. Compared to labeled image data for computer vision, the problems of data collection, augmentation, and synthesis are only to a limited extent solved for QA, and especially for KGQA. There are few datasets for KGQA overall, and little previous work that has found unsupervised or semi-supervised learning approaches to address the sparsity of data. Instead, neural network approaches to KGQA rely on either fully or weakly supervised learning [29].

We are thus concerned with neural models trained in a supervised setting to perform QA tasks, especially of the KGQA flavor. Given a clear task to delegate to a computational system, it seems clear that we want the task performed as well as possible. However, what methodological elements are important to ensure good system performance within the chosen scope? How should the quality of system performance be assessed? This thesis describes work done to address these overarching questions through a number of more specific research questions. Altogether, we designate the topic of this thesis as *KGQA evaluation*, which we address in a broad sense, encompassing four subtopics from (1) the impact on performance due to *volume of training data* provided and (2) the information leakage between training and test splits due to unhygienic *data partitioning*, through (3) the naturalness of NL questions resulting from a common approach for *generating KGQA datasets*, to (4) the axiomatic analysis and development of *evaluation measures* for a specific flavor of the KGQA task. Each of the four subtopics is informed by previous work, but we aim in this thesis to critically examine the assumptions of previous work to uncover, verify, or address weaknesses in current practices surrounding KGQA evaluation.

1.1 Research Questions

We formulate four main research questions, each of which is further articulated with specific subquestions. First of all, in the supervised learning setting models are often trained using as large a training split as possible, but it is not always clear how much this benefits the final performance. This is important since the construction of datasets of high quality and volume is a considerable challenge in QA. The first main research question is then:

RQ 1 How does training data volume impact QA performance?

This research question is relevant to other methods of question answering using neural architectures, and even to machine learning in general. We argue that considering the capability of machine learning methods to learn from variable volumes of training data is an important way to evaluate not just the actual performance of a machine learning method for a given training dataset, but also to estimate the potential of the method to learn its task if trained with a different training dataset. In other words, the evaluation methodology should seek to generalize about the particular machine learning method, and not just for the machine learning method to generalize from training split to test split. If the models' performance does not respond to increased training data volume, then the model is not learning. Working with QA in the flavor of *answer sentence selection* (given a question, selecting an answer sentence from a candidate set; see Definition 2.3.1.), we investigate the following specific research questions:

RQ 1.1 To what extent do QA models improve when trained on a larger volume of data?

RQ 1.2 How sensitive are the models to fractional changes in training data volume?

Based on findings from initially addressing the above research questions with respect to the answer sentence selection flavor of QA, we subsequently proceed with a focus on KGQA due to the intrinsically richer structure in the data for that task. A particularly motivating meta-research question in this regard is "How can we know if any meritorious model performance is merely a question of vocabulary similarity or corresponds to an actual 'understanding' of the natural language?"

Second, we look at how datasets are constructed and partitioned into training, validation, and test splits to follow the common pattern of supervised learning. Large scale datasets for KGQA may be constructed without (additional) crowdsourcing by synthetically generating instances from templates which have been extracted from pre-existing datasets [57]. A *template* consists of text and placeholders which can be given values; setting values to the placeholders creates an instance from the template. Each template is used as the basis for many instances, sometimes hundreds. If the model is trained on instances generated from a given template, then we can say the model has *seen* the underlying template. The second main research question is then

RQ 2 How does it affect model performance or behavior if the test split includes instances that were generated from templates *seen* during training?

To address this we investigate the following specific research questions:

RQ 2.1 How is the performance of trained neural KGQA models affected by whether testing templates are *seen* or *unseen*?

RQ 2.2 How is the ability to generalize to instances based on *unseen* templates affected by the volume of training data used?

RQ 2.3 How does the proportion of *unseen* templates to *seen* templates affect the trained models' ability to generalize?

Third, as noted, the supervised learning setting makes the assumption that instances in the dataset represent the task which the model should learn. The quality of data used in KGQA is therefore important, as supervised neural models must be trained and tested on large datasets specific to the KGQA task. Consequently, we look more closely at the common approaches of KGQA data generation involving crowdsourcing. A crucial characteristic of KGQA (and QA generally) is that the task involves taking as input a *natural language* question. Typically, the ground truth label is handcrafted by experts or generated automatically as a correct formal query from the KG, along with some corresponding template-based pseudo-natural language question. Crowd workers are then tasked with paraphrasing the generated pseudo-natural question into a more genuinely natural NL question. However, we hypothesize that this division of labor in dataset construction introduces a mismatch between the formal query intent and the NL question intent. The third main research question is then:

RQ 3 How can we ensure or enhance the genuine naturalness of NL questions in KGQA datasets?

In order to answer this larger research question, the following more specific research questions are defined:

RQ 3.1 Can the crowdsourced NL questions in existing KGQA datasets be considered as genuinely natural?

RQ 3.2 What are the properties of a high quality NL question?

RQ 3.3 What happens to the performance of KGQA models when testing against genuinely natural questions?

Fourth, we consider whether the evaluation measures used to quantify the performance of KGQA models are appropriate. Considering the KGQA task, and the established *information retrieval* (IR)-based and *semantic parsing* (SP)-based approaches (see Sect. 2.4), we focus on the SP-based paradigm, because from an evaluation perspective, SP-KGQA methods not only can be used to retrieve answers like IR-based KGQA, they also yield formal queries that are in principle interpretable and can be used to check if the model performance is due to an appropriate understanding of the NL question or merely a happy accident due to associations of terms. Thus, the "externally visible" behavior of SP-KGQA systems encompasses that of IR-KGQA systems, while the converse is not guaranteed. We therefore look critically at established measures for SP-KGQA and conduct an axiomatic analysis leading to contributions towards a formally grounded evaluation framework for SP-KGQA. The fourth main research question is then:

RQ 4 What are appropriate evaluation measures for SP-KGQA performance evaluation?

To address this, we investigate the following specific research questions:

RQ 4.1 What, if any, are the shortcomings of commonly used measures in SP-KGQA evaluation?

RQ 4.2 Can we reason axiomatically about appropriate SP-KGQA evaluation measures?

RQ 4.3 If so, can we construct measures for SP-KGQA evaluation that are formally grounded?

In summary, in this thesis we concern ourselves with: the capacity of a neural model to learn from increasing volumes of training data; the hygienic use of synthetically generated data for training and testing neural models; the quality of the data with respect to representing the intended task; and, the evaluation measures used to quantify performance in evaluation.

1.2 Main Contributions

In seeking to address these research questions, we bring a critical perspective to how KGQA systems are to be evaluated. The research findings show the potential for meaningful improvements over the currently common practices in KGQA evaluation. The main contributions presented in this thesis are listed below. Each contribution is categorized as being either a *theoretical and methodological*, *resource*, or *insight* contribution.

Theoretical and Methodological Contributions

- In Chap. 4 we present a novel dataset partitioning scheme that provides a facility to quantify the generalized learning achieved by models trained on template-generated synthetic data.
- In Chap. 5 we present a novel coding scheme to characterize to what extent nominal natural language questions in KGQA datasets actually constitute natural language.
- In Chap. 6 we present a novel SP-KGQA evaluation framework and novel measures, as well as a first formulation of axioms for SP-KGQA measures, which we apply in an axiomatic analysis of established and novel measures.

Resource Contributions

- In Chap. 5 we present a novel test collection, IQN-KGQA, consisting of 3x250 questions sampled from 3 prominent KGQA datasets, made publicly available at `https://github.com/iai-group/IQN-KGQA`. The sampled questions are rated on unnaturalness along 5 dimensions by at least 3 crowd workers each, and rewritten for greater naturalness where possible.

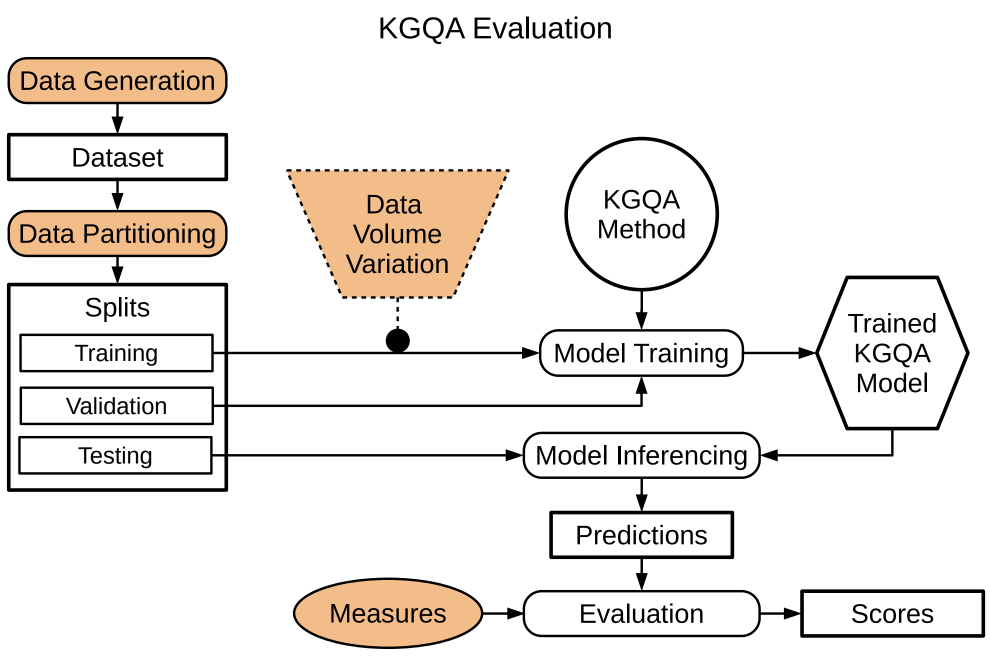


Figure 1.1: Overview of KGQA evaluation as a system of components, as well as the experimental intervention of *data volume variation*. The colored items represent the subtopics of KGQA evaluation investigated in this thesis.

Insight Contributions

- In Chap. 3 we find that most state-of-the-art answer sentence selection models do not exhibit the expected behavior in terms of performance improvement in response to increased training dataset size.
- In Chap. 4 we identify the problem of information leakage in template-based synthetic generation approaches. The significance of our finding, however, extends beyond KGQA, as it applies to any template-base data generation approach, and raises a set of interesting questions around training models with synthetic data using fair conditions.

1.3 Organization of the Thesis

The abstract architecture for KGQA evaluation is illustrated in Fig. 1.1, including the subtopics around which this thesis is organized. *Data generation* is the process of creating the *dataset* where every instance represents a pair of values, input $\vec{X}$ and output (or label) y. By *data partitioning* the dataset is divided into *splits* to be used exclusively for *training*, *validation*, or *testing*, respectively. The training and validation splits are used in the *model training* process, with model architecture, hyperparameters, and other settings defined by the chosen *KGQA method*. For some of our experiments, we perform *data volume variation* and restrict the volume of training data used in model training. Regardless of any applied data volume variation, the resulting *trained KGQA model* is then used to perform *model inferencing* to make *predictions* ($\hat{y}$), based on the input ($\vec{X}$) in the test split. Given the test split ground truth labels (y) and corresponding predictions ($\hat{y}$), as well as the definitions of some *measures*, it is then possible to perform the *evaluation* and quantify the performance of the system with some evaluation *scores*.

The background and previous literature relating to the work presented in this thesis are summarized in Chap. 2. Next, in Chap. 3 we consider the impact of varying the volume of data used in training in Chap. 3. After that, we look into the impact of a more circumspect data partitioning scheme in Chap. 4. In Chap. 5 we investigate the naturalness (or lack thereof) of NL questions produced by the common method of data generation for KGQA. This is followed in Chap. 6 by our axiomatic analysis of the formal basis for task-specific measures for SP-KGQA, and our derivation of novel measures. Finally, we review and discuss the implications of our findings in Chap. 7.

1.4 Origins of the Material

Chronologically, the thesis is based on the following articles, all accepted after peer-review:

- Linjordet and Balog [77]: "Impact of Training Dataset Size on Neural Answer Selection Models" (ECIR 2019, Short paper)
- Linjordet and Balog [78]: "Sanitizing Synthetic Training Data Generation for Question Answering over Knowledge Graphs" (ICTIR 2020, Full paper)
- Linjordet [76]: "Neural (Knowledge Graph) Question Answering Using Synthetic Training Data" (CIKM 2020, Doctoral consortium paper)

- Linjordet and Balog [79]: "Would You Ask it that Way? Measuring and Improving Question Naturalness for Knowledge Graph Question Answering" (SIGIR 2022, Resource paper)
- Linjordet et al. [80]: "Towards Formally Grounded Evaluation Measures for Semantic Parsing-based Knowledge Graph Question Answering" (ICTIR 2022, Full paper)

Chapter 2

Background

In this chapter, we describe the technical background for our research which we subsequently present in Chaps. 3—6. In Sect. 2.1, we review the concepts of machine learning and neural networks as they pertain to our machine learning experiments. In Sect. 2.2, we provide background on knowledge graphs (KGs) and structured data. In Sect. 2.3, we discuss the broad task of question answering (QA), along with an overview of its specific flavors. On this basis, we go into detail in Sect. 2.4 about the task definition of knowledge graph question answering (KGQA). In Sect. 2.5, we discuss datasets for KGQA and how they are created. In Sect. 2.6, evaluation measures are discussed in the context of KGQA, as well as axiomatic approaches to analyzing existing evaluation measures and developing novel evaluation measures. Finally, in Sect. 2.7, we give an overview of the methods—e.g., neural network architectures—used to construct KGQA systems.

2.1 Machine Learning and Neural Networks

Machine learning refers to a broad family of algorithms that instantiate models which learn from data how to perform tasks [52]. Neural networks are a type of machine learning algorithm where the parameters learned from data are the weights on directed edges in a graph where the nodes are non-linear activation units. Usually neural network architectures include several layers of parallel components which combine input from a previous layer, apply a non-linear transformation—e.g., the logistic sigmoid function—and then pass the result forward on the weighted edges connecting an activation unit on one layer to some or all activation units in the next layer. Deep neural networks include many layers (more than six is the heuristic according to Goodfellow et al. [52]) and are able to capture complex relationships between raw input variables. Prior to the current era of deep learning, complex manual feature engineering was often applied to raw data, yielding engineered feature vectors, which were then used in training models. With sufficient volume and quality of data, deep neural networks are effectively able to learn features from raw data, eliminating or reducing the need for feature engineering.

In the present work we use the term *method* to refer to a specific set of steps to create a task-performing system, such as a KGQA model. In principle, a method may also consist of manually and formulaically constructing a system to perform the intended task. In contrast, in the present work we focus on machine learning methods that train models on data to perform a task. Specifically, we use deep neural network (DNN) methods because DNN models have shown higher performance on complex tasks in many domains compared to models created by other methods [29, 32, 56, 58, 71, 74, 87, 130, 155].

Next, we provide a brief overview of machine learning paradigms, their relationships to datasets, and how research is looking at the impact on machine learning by different methods of partitioning datasets. We explain the supervised learning paradigm in Sect. 2.1.1 and its relationship to training data. We contrast this with related paradigms of machine learning in Sect. 2.1.2. In Sect. 2.1.3, we summarize work on learning capacity in neural networks. Finally, in Sect. 2.1.4 we describe investigations into schemes for partitioning datasets into different splits for training and different stages of evaluation. These topics inform the common background for our work using primarily neural methods in the supervised learning setting. For example, we explain briefly what self-supervision is with reference to the BERT architecture in Sect. 2.1.2. This provides context to understand the role of BERT in Sect. 2.7, as well as the challenge the KGQA task poses for dataset construction as described in Sect. 2.5.2.

2.1.1 Supervised Machine Learning and Datasets

In *supervised* machine learning, the model is trained using data consisting of instances $(\vec{X}, y)$ to perform model inferencing, i.e., the prediction task $\vec{X} \to \hat{y} \approx y$. Here, $\vec{X}$ represents the input or independent variables of the instance, y is the ground truth output or label, and the model must learn to make a prediction or estimate $\hat{y}$ that approximates the ground truth y. Every instance is a point sampled from a distribution, and the model should learn to generalize, i.e., to correctly infer from novel $\vec{X}$ to appropriate corresponding $\hat{y}$. The datasets used to train, validate, and test a machine learning model are critically important to achieve an effective model that can perform the task correctly for the desired range of possible inputs $\vec{X}$.

The datasets must be large and varied enough so that the machine learning model can capture the relationships between different variables in $\vec{X}$. In addition, the quality of the dataset is important, meaning that the instances in the dataset must correctly represent the task to be learned.

2.1.2 Un-, Semi-, and Self-Supervised Machine Learning

In contrast to supervised machine learning, where data consists of instances $(\vec{X}, y)$ and y is a verified and possibly manually added (i.e., annotated) ground truth label, there are some other machine learning categories. Here it is worth pointing out *unsupervised*, *semi-supervised*, and *self-supervised* machine learning.

Unsupervised learning assumes there are no labels y for a given data point $\vec{X}$, and the learning algorithm itself should find the structure in the dataset. For example, k-means clustering algorithms may be trained to classify instances into a predefined number $k \in \mathbb{N}$ different classes, without seeing any labeled instances during training [52].

Semi-supervised learning combines supervised and unsupervised learning, to make the most of available data. For example, if only a small non-empty subset of instances $\vec{X}$ have an associated ground truth label y, then the labeled instances may be used to train, in a supervised manner, a model which in turn can be used to pseudo-label all the remaining unlabeled instances. The complete dataset can then be pseudo-labeled and used to train a final model [139].

Finally, self-supervised learning refers to the case where a substitute for a label is found in the context of the raw, unlabeled dataset [103, 108]. For example, large language models, such as BERT [35], are trained on raw text data collected on the Internet.[1] Instead of manually adding or automatically generating labels,

[1]Specifically, Devlin et al. [35] first trained (pre-trained) BERT in a self-supervised manner, then further trained (fine-tuned) the model in a supervised manner on specific language tasks.

the prose text itself is used to construct both $\vec{X}$ and y, either by masking a random word in a sentence and training the model to predict that word, or else masking the next sentence in a paragraph and training the model to predict the whole subsequent sentence.

2.1.3 Memorization and Generalization in Neural Networks

As mentioned in Sect. 2.1.1, supervised machine learning models learn to perform tasks from labeled training data. In this work, we ask how training data volume impacts question answering (QA) models (RQ 1). Deep neural network methods currently comprise the most promising family of machine learning approaches. Therefore, we look at previous research into the impact of training data volume on machine learned neural models.

The impact of the size of training datasets has been investigated for convolutional neural networks (CNNs) trained on image data [30, 127]. In the latter work, it was observed that model performance improves roughly logarithmically as a function of increased training data. The idea of a logarithmic relationship between performance and dataset size was further corroborated empirically by Hestness et al. [59].

An investigation of the generalization problem in deep neural networks, i.e., the discrepancy between the performance of a trained model on training data and test data, shows that the deep neural models have a representational capacity that enables “memorization” of training data: Zhang et al. [153] show the order-of-magnitude relationship between training dataset size (sample size), input data dimensionality, and the depth of a network with sufficient parameters to fully memorize the training dataset. They report a theorem with proof such that for any finite n-sized sample of d-dimensional inputs, there exists a two-layer ReLU neural network with $2n + d$ weights that can represent any function on the sample. As a corollary, this finding extends from this hypothetical shallow and wide network to a narrow and deep network where the relationship between sample size and number of parameters is conserved. This may not be how deep neural networks learn in practice [12], but the theorem indicates the challenge that finite datasets may present to generalization in deep learning models.

More recently Nguyen et al. [92] have worked with natural language data while relying on pre-trained language models such as BERT. Their findings indicate that given a pre-trained model in a similar domain, the need for task-specific training data is much reduced. Zhang et al. [154] have updated their previous work [153] with a survey of recent related developments, and note that a model fitting and hence memorizing all training instances is "not necessarily at odds with generalization." Ongoing empirical research may further elucidate generalization and memorization in the context of machine learning with deep neural networks.

2.1.4 Dataset Partitioning

The commonly accepted practice in machine learning is to address datasets partitioned into disjoint splits for training, validation, and testing. During training, the machine learning model's parameters are adjusted, e.g., by the gradient descent algorithm, to optimize model performance on a loss function with respect to instances in the training split. The validation split is reserved for adjusting hyperparameters, while the test split is reserved to estimate the final model's ability to generalize to unseen instances. The difference in error rates of a trained model inferencing on the training split versus on the test split is called the *generalization gap* [52].

Dataset partitioning thus affects both learning optimization and the ability to estimate how well the trained model has learned to generalize. For this reason, methods of dataset partitioning are the subject of research within the field of machine learning [49, 66, 68]. Gu et al. [53] explore the use of multiple test splits to characterize discrete levels of generalization in the KGQA task: generalizing to (1) *independent and identically distributed (IID)* generalization, where test instances come from the same distribution as training instances, (2) *compositional generalization*, where test instances represent novel compositions of schema items seen during training, and (3) *zero-shot generalization*, where test instances comprise previously unseen schema items "or even domains."

2.2 Structured Data and Knowledge Graphs

Data are considered to exist on a spectrum from *unstructured* through *semi-structured* to *structured* [16]. Where a particular dataset is placed on this spectrum depends on the degree to which the data has features that support systematized (i.e., machine) processing. Plain text written in natural language is a typical example of unstructured data, which can be treated simply as a sequence of words and symbols [16].

Structured data, on the other hand, is data that can typically exist in a relational database, populating a system of interrelated tables that express an explicit data model, highly organized and following a strict schema [16]. A relational database can be queried with Structured Query Language (SQL), and the (formal) query then represents a relational algebra operation on the database. Formal queries are also called *logical forms*, a term derived from formal logic in philosophy [109].

The relational database lends itself well to a system where a large number of similar types of records are to be kept, and for each table the variables to be populated for those records, as well as the permitted values, are known ahead of time [16].

Standing between the two farthest points of the spectrum, semi-structured data refers to data that is not rigidly structured, but contains typically a combination of unstructured data such as raw text, as well as structural elements that are annotations in the document markup according to some schema for resources such as entities, classes, relationships, and properties.[2] Two common document formats that exemplify semi-structured data are HTML and XML.

The annotations in the semi-structured data are optional for each document, and the schema of annotations may be developed in an ad hoc manner. Therefore, semi-structured data is said to have a *self-describing* schema [16]. Such semi-structured data may also be organized and annotated using the Resource Description Framework (RDF) data model, and the specific schema adopted provides or informs which "uniquely and globally identifiable" Uniform Resource Identifiers (URIs) are assigned to resources [16].

This way of annotating documents is helpful to search engines or other services that can process the structured elements in semi-structured data [15]. Semi-structured data as a machine readable annotation embedded in a document itself is a technology that undergirds the larger mission of projects variously called the Semantic Web, Linked Open Data, semantic technology, and so on.[3]

[2]One common set of schemata is provided by `https://www.schema.org`.

[3]*Semantic* here means that there is structured data in a document that refers to a common, explictly defined schema. This is a slightly different meaning from the linguistic or colloquial use,

The RDF approach to structured data, of annotating documents with facts encoded as *subject, predicate, object-triples* (SPO-triples), also implies that the structured data elements from all sources, e.g., relational databases or the annotated documents in a collection, together can be organized as a set of facts in a database, called a knowledge base. Taking the perspective that all subjects and objects constitute vertices, and that all predicates form edges, we can say the database of facts is a *knowledge graph* (KG). This is sometimes called a knowledge base (KB) in parts of the literature, but to emphasize the graph nature of KGs, we refer to them as such in the present work. Knowledge graphs can be subjected to formal queries much like relational databases, but use KG-specific query languages. Formal queries typically include *triple patterns*, which consist of three components, S, P, O, like the SPO-triples mentioned above, but where each element may be treated as a variable rather than a determined value. The most common formal query language for KGs is SPARQL.[4]

A major distinction from formally querying relational databases or tabular data is that KGs have explicit relational semantics via the predicates, which represent relationships between subject entities and object entities or properties where subject entities have object literal values.

The three largest and most widely used open-domain KGs are DBpedia[5], Freebase[6], and Wikidata.[7] While Freebase is discontinued as a service, the knowledge graph can be copied and self-hosted.[8]

---

where the term *semantic* refers to the meaning of language. Often this sense is intended in contrast with *syntax*, the structure of language. However, in this sense, the semantics of a term or statement are not always unambiguously tied to a single denotational definition. However, "semantic" in the semantic technology sense refers to a document's markup containing structured data, i.e., a machine readable annotation about or some piece of text in the document having an unambiguous correspondence with some item in the schema used for structured data in the document.

[4]https://www.w3.org/TR/sparql11-query/

[5]https://www.dbpedia.org/

[6]http://rdf.freebase.com/, was discontinued and migrated into Wikidata.

[7]https://www.wikidata.com

[8]Freebase discontinued: https://web.archive.org/web/20120516075431/http://blog.freebase.com/2008/10/30/introducing_the_rdf_service/
Latest dump available for download: https://developers.google.com/freebase

2.3 Question Answering

Since the early days of computer science and the field of artificial intelligence [86, 132, 144], the concept of a machine being able to appropriately answer natural language (NL) questions posed by a user has been much studied. These early works argued that if this concept—the question answering (QA) task understood broadly—could be realized, the system's ability to "understand" natural language would have reached a threshold required to merit the title of artificial intelligence. The term "understand" here is taken pragmatically, as the system's responses would be an imitation of a human response to natural language behavior.

Question answering (QA) over unstructured text has been the focus of research for decades within the fields of information retrieval (IR) and natural language processing (NLP) [17, 28, 107, 141]. The field of NLP has approached QA as a problem to solve using linguistics, common sense and "highly structured data bases [*sic*]" [1]. Until machine learning methods became applicable, NLP used formal theories of linguistics to inform the design of handcrafted computer programs to parse and process NL text.

In contrast, IR has initially adapted methods from ad hoc retrieval to the QA task. Ad hoc retrieval in IR seeks to satisfy a user's information need by presenting a ranked list of the most relevant documents, where the information need is expressed by a query which may be a collection of keywords, or an NL question, or some intermediate form [85]. The IR approach to QA further seeks to retrieve the correct short text (e.g., paragraph, sentence, or text span) that directly answers the NL question posed by the user. These fields of research represent different paradigms and may interpret the QA task differently in light of their respective perspectives and preferred methods [28].

Whether for paradigmatic or practical reasons, various flavors of the QA task have emerged. We name and briefly summarize the task descriptions for prominent elementary flavors of QA. For conciseness, the task description is expressed in imperative form.

***Definition 2.3.1 - Answer Sentence Selection**:*
(or simply *answer selection*:) Given an NL question and a set of question-specific candidate answers, select the correct answer from a predefined set of candidate answers.

***Definition 2.3.2 - Answer Span Prediction**:*

Given an NL question and a short piece of prose as *context*, select the span of text from the context which is the direct answer to the question.

***Definition 2.3.3 - Answer Generation**:*

Given an NL question and a short prose context, generate the correct answer as a fluent NL statement.

***Definition 2.3.4 - Knowledge Graph Question Answering (KGQA)**:*

(or *Knowledge Base QA*:) Given an NL question and a KG, return the answer(s) to the question based on the factual support in the KG.

These task descriptions are accurate in the abstract, but we also note that research typically addresses datasets (or *benchmarks*) that may concretize these tasks in idiosyncratic ways. For example, datasets WikiQA[9] [148] and SQuAD[10] [104] represent the tasks of answer sentence selection and answer span prediction, respectively, exactly as described above. In contrast, the dataset MS MARCO[11] [91] consists of several different tasks and flavors thereof, but ties an IR subtask to its QA tasks, e.g., QnA v1.1: *"Given a query and 10 candidate passages select the most relvant [*sic*] one and use it to answer the question."* In other words, the QA task is here cast as a combination of ranking candidate short prose contexts, and then using the highest ranking one as a basis to generate an answer as a fluent NL statement to answer the NL question. This example of the MS MARCO flavors of QA illustrates the interplay between task definition, the realization of the task as a dataset, and the technological approach to the task anticipated in the design of the dataset. Flavors of the QA task which involve inferring an answer to an NL question from a context are classified under the *machine reading comprehension* (MRC) [106, 107] class of QA flavors. This includes both answer span prediction and answer generation under the definitions listed above.

Research in QA is largely focused on answering fact-based or *factoid* NL questions [107], likely because system evaluation is more easily operationalized in this setting. In other words, if each NL question has a single "objective" answer [107],

[9] `https://www.microsoft.com/en-us/research/publication/wikiqa-a-challenge-dataset-for-open-domain-question-answering/`

[10] `https://rajpurkar.github.io/SQuAD-explorer/`

[11] `https://microsoft.github.io/msmarco/`

then it is easier to decide whether or not the correct answer was returned. In the case of answer generation, and especially if the NL question is not factual in nature, it is clear that in general, innumerable valid answers or formulations thereof could exist.

For fact-based QA, it is assumed some “evidence input” [107] is provided along with the NL question for the system to be able to infer the correct answer. In the case of answer span prediction and answer generation as described above, the short prose context constitutes the evidential basis for answering the NL question. As implied by the MS MARCO inclusion of a context selection step in the QA task, it is possible to consider whole documents or even text collections (corpora) as the evidential basis for QA [91, 107]. Any of these QA task flavors can also be subsumed in the *Conversational QA* setting [102],where the previous turns of dialogue between the user and the system constitute part of the context or evidential basis to answer an NL question correctly.

None of these task flavors necessarily consider structured data as the evidential basis for fact-based QA. Some systems have been constructed to perform QA on semi-structured tabular data [99] as well as on relational databases via text-to-SQL semantic parsing [54, 152]. However, the KGQA flavor takes advantage of a far richer structured data by way of the more complex and formalized schema underlying the KG as evidential basis for QA [38, 54, 107, 134].

2.4 Knowledge Graph Question Answering

By harnessing structured data in the form of KGs, knowledge graph question answering (KGQA) can facilitate information access that would otherwise require expertise in formal query languages. Here we consider the relationship between the KGQA task, defined abstractly, and its interpretation in light of practical approaches and assumptions.

2.4.1 The KGQA Task

The KGQA task is a type of fact-based question answering (QA), using a knowledge graph (KG) as the background knowledge for obtaining answers to questions. Reiterating our definition above, the KGQA task is:

Definition 2.4.1 - KGQA:
Given an NL question and a KG, return the answer(s) to the question based on the factual support in the KG.

2.4.2 Approaches to KGQA

The task of KGQA is typically approached as either an *information retrieval* (IR) or a *semantic parsing* (SP) problem [10, 29, 72, 90].[12]

IR-KGQA

IR-based KGQA casts the task as the problem of generating, scoring, and then selecting candidate solutions for a given NL question. Lan et al. [72] call this a "retrieval-and-rank" paradigm. For a particular KGQA method, the candidate solutions will be all of one type, e.g., formal queries [62], query graphs [84, 149] (which can be deterministically translated to a formal query), or answers (entities and literals in the KG, or operations upon these) [138]. Because of this variety of approaches, we can formally define IR-KGQA with only limited specificity.

Definition 2.4.2 - IR-KGQA:
Given an NL question q and a knowledge graph $\mathcal{K}$, construct and return a (ranked) set of (candidate) answers or answer elements a.

SP-KGQA

SP-KGQA can be cast as semantically parsing an NL question to produce a formal query, which is executed on the KG to produce the predicted answer. Lan et al. [72] call this a "parse-then-execute" paradigm. SP-KGQA is characterized by a single NL question being parsed into a single formal query. Formally, the SP-KGQA task is defined [29] as:

Definition 2.4.3 - SP-KGQA:
Given an NL question q and a knowledge graph $\mathcal{K}$, predict a formal query f that executes on $\mathcal{K}$ to return the correct answer a, such that f also correctly represents the meaning of q.

[12] In fact, Chakraborty et al. [29] claim semantic parsing is the most common approach to KGQA.

Here, the execution of the formal query against the KG yields the one predicted answer to the NL question. This one answer may itself be a set of multiple elements, such as a list of entities with attributes. One approach to solving the SP-KGQA task has been using machine translation (MT) methods, such as various neural machine translation (NMT) architectures [29, 42]. Under this approach, the KGQA training data, consisting of question-formal query pairs, are treated as source and target language instances.

2.4.3 Perspectives and Assumptions

IR-KGQA has the advantage of being able to robustly leverage entity descriptions as well as graph structure. However, it is difficult to interpret how the system arrived at the retrieved answers, and to verify whether the system's "understanding" of the question was correct. Conversely, SP-KGQA predicts an explicit formal query (e.g., SPARQL) that represents the NL question posed by a human user, and, in turn, executes the formal query to retrieve answers [29, 72]. This provides greater interpretability by showing explicitly how the system "understood" the NL question. This means the reasoning represented by the formal query can be reconstructed in natural language by a human who is an expert in the formal query language.[13] In other words, the interpretation may lie beyond the ability of non-expert users in the case of complex formal queries, but the fact that a single formal query is expressed allows interpretation in principle.

Vakulenko et al. [138] illustrate the challenge of categorizing KGQA approaches, as their method involves separate stages called "parsing" and "matching," in a phase called "question interpretation," which indicates a staggered combination of semantic parsing and information retrieval. In the "parsing" stage, references to the categories of entity, predicate, and class are extracted, and the type of question is determined, whereupon the extracted references are matched with a ranked list of candidate elements from the KG. Ultimately, their approach can be categorized as IR-KGQA, not only because typical IR techniques, such as ranking and matching, are used throughout their KGQA system, but because crucially, no single formal query is produced to represent the system's "understanding" of the NL question.

Surveys on KGQA assert that the answer returned by KGQA must be based

[13]We make extensive use of the term "expert" in this sense in Sect. 2.5.

on a set of entities from the KG and acknowledge that possible answers in KGQA include sets of entities or literals from the KG, numerical results from aggregation operations on the KG, or boolean values `True/False` [29, 54, 72]. Chakraborty et al. [29] also acknowledge that "answers could take on complex forms, such as ordered lists ... or even a table." We follow the perspective of Chakraborty et al. [29] as the most coherent and least restrictive interpretation of the KGQA answer space.

2.5 Datasets and Dataset Generation for KGQA

As described in Sect. 2.4, the KGQA task can be approached in different ways, and the datasets used to train and evaluate KGQA systems have a defining role in concretizing the task. The following section therefore discusses existing datasets and the way they have been created. In Sect. 2.5.1 we discuss the distinction between simple and complex KGQA. In Sect. 2.5.2 we look at historically important KGQA datasets from previous work in the field with a focus on how each dataset was created. The reported approaches to creating or augmenting KGQA datasets are then summarized in Sect. 2.5.3, with an emphasis on datasets for complex KGQA where manual labeling, especially using crowdsourced labor, played a role in dataset creation.

2.5.1 Simple and Complex KGQA Datasets

There exist several KGQA benchmarks, which can be broadly classified as *simple* or *complex*. Simple KGQA means finding the answer is only about completing a single SPO triple (fact or relation) that exists in the KG: to retrieve the object o, given a subject s and predicate p as input. In contrast, complex KGQA means each formal query addresses more than one SPO triple in the KG. Thus, valid complex KGQA formal queries can be mapped to KG subgraphs with more than one edge. Examples of simple KGQA benchmarks include WebQuestions [20], SimpleQuestions [22], and Free917 [27] over Freebase, and SimpleQuestions [39] over Wikidata.

In recent years, the focus has been shifted to complex KGQA benchmarks [100, 129], examples of which include LC-QuAD v1.0 [131] and DBNQA [57] over DBpedia, LC-QuAD 2.0 [43] over DBpedia and Wikidata, ComplexWebQuestions [129],

Table 2.1: Chronological overview of KGQA datasets.

Dataset	Year	KG	Size	Crowdsourcing
Free917 [27]	2013	Freebase	917	Unclear
WebQuestions [20]	2013	Freebase	5,810	Yes
SimpleQuestions [22]	2015	Freebase	108,442	Unclear
ComplexQuestions [18]	2016	Freebase	2,100	No
GraphQuestions [126]	2016	Freebase	5,166	Yes
WebQuestionsSP [150]	2016	Freebase	4,737	No
LC-QuAD v1.0 [131]$^{\diamond}$	2017	DBpedia	5,000	No
QALD series (1–9) [81, 135, 136]	2013–2018	DBpedia	~50-500 each	No
ComplexWebQuestions [129]	2018	Freebase	34,689	Yes
DBNQA [57]$^{\diamond,\dagger,\ddagger}$	2018	DBpedia	894,499	No
LC-QuAD v2.0 [43]†	2019	DBpedia, Wikidata	30,000	Yes
CFQ [69]	2020	Freebase	239,357	No
KQA Pro [119]	2020	Wikidata	117,970	Yes
GrailQA [53]†	2021	Freebase	64,331	Yes

$\diamond$: in Chap. 4. †: in Chap. 5. ‡: in Chap. 6.

ComplexQuestions [18], and GraphQuestions [126] over Freebase. The work presented in Chaps. 4—6 focuses on complex KGQA, as opposed to simple KGQA.

2.5.2 Chronology of Salient KGQA Datasets

The field of KGQA is in many ways defined by the datasets used to train and test systems. At the same time, unlike tasks such as the self-supervision tasks used to pre-train BERT [35], appropriate datasets for the KGQA task cannot be found freely in existing, incidental data, i.e., "in the wild." The processes used to create KGQA datasets are therefore of interest. In the following, we describe some salient milestone KGQA datasets and their manner of construction. Previous work [29, 72, 107, 145] has surveyed the field of KGQA, which we draw on in our present summary. The KGQA datasets are grounded in one or more of the three most common open-domain knowledge graphs (KGs): Freebase, DBpedia, and Wikidata. We note that the overall trend in KGQA dataset construction has been towards more complex formal queries as well as larger datasets. For each dataset, we defer to the respective papers' stance as to whether the dataset should be considered to contain complex formal queries. The datasets are listed chronologically in Table 2.1 in order of year of publication, and the creation procedure is summarized for each dataset below. While not exhaustive of all KGQA datasets, this chronology reflects the major trends in KGQA dataset construction approaches over the last decade.

Free917: Cai and Yates [27] create the dataset Free917 by asking two native English speakers to pose questions in multiple domains, and then annotating these questions with formal queries.

WebQuestions: Berant et al. [20] construct the dataset WebQuestions, consisting of 5810 instances with only NL questions and answers, but no formal queries. The dataset is constructed by generating single-entity questions with the Google Suggest API, and then crowdsourcing answers based only on the Freebase page of the entity in a given NL question. Question-answer pairs are kept as instances when at least two crowd workers agree on an answer.

SimpleQuestions: Bordes et al. [22] create the large dataset SimpleQuestions, consisting only of NL questions that can be answered by a single fact (SPO-triple) in the KG, and the corresponding fact. The dataset is created by short-listing a set of facts, and then having English-speaking annotators generate NL questions mentioning the subject and predicate of the fact, such that the answer would be the object.

ComplexQuestions: Bao et al. [18] construct the dataset ComplexQuestions consisting of question-answer pairs by mining a search query log for queries with overlapping terms as in WebQuestions and SimpleQuestions, and then categorizing the search queries according to some rules to identify multi-constraint questions. The questions are manually annotated with answers. Additional question-answer pairs are taken directly from pre-existing datasets.

GraphQuestions: Su et al. [126] construct the dataset GraphQuestions—where each instance includes NL question, formal query, and ground truth answer—by first generating query graphs, and then converting these to NL questions via crowdsourcing. The ground truth answer is retrieved by converting the query graph to a formal query and executing it. This approach to crowdsourcing for KGQA datasets has been referred to as the Overnight method [126].

WebQuestionsSP: Yih et al. [150] construct the dataset WebQuestionsSP by having experts annotate instances in WebQuestions [20] with SPARQL queries where feasible.

LC-QuAD v1.0: Trivedi et al. [131] construct the dataset LC-QuAD v1.0, which consists of NL questions and formal queries. The dataset is created from a set of 38 hand-made abstract query graphs extending at most two hops from a seed entity.[14] First query graph templates are combined with whitelisted (non-metadata) entities and predicates, instantiating formal queries. Pseudo-NL question templates are used in parallel to instantiate pseudo-NL questions. These tentative template-based pseudo-NL questions are paraphrased by non-expert annotators to improve the grammar of the question. The resulting paraphrased NL questions are then reviewed and revised by experts.

QALD series (1–9) [81, 135, 136]: A series of datasets, from the Question Answering over Linked Data (QALD) challenges,[15] are almost exclusively created manually at small scale. Each dataset consists of questions generated by students and formal queries hand-crafted by experts. Within this initiative, the re-use and revision of data from previous years has been common.

ComplexWebQuestions: Talmor and Berant [129] construct the dataset ComplexWebQuestions by programmatically generating more complex formal queries from WebQuestionsSP [149] by adding constraints to each instance. The pseudo-NL question templates are extended with manually constructed predicate-specific templates. The generated pseudo-NL questions are paraphrased by non-expert crowd workers into NL questions.

DBNQA: Hartmann et al. [57] construct the dataset DBNQA, consisting of NL questions and formal queries, from the LC-QuAD v1.0 [131] and QALD-7-train [136] datasets. The pre-existing datasets are taken as the basis (seeds) to extract templates for both formal queries and NL questions, and those templates are then instantiated with different entity and predicate bindings.

Templates are extracted manually from QALD-7-train, but semi-automatically from LC-QuAD v1.0. In the latter case, the resulting templates are reviewed by SPARQL experts. For each entity URI or surface form in the seed data, corresponding placeholders are inserted in the templates. The templates are then instantiated using the results of the executable SPARQL templates applied to a DBpedia endpoint to find suitable entities for the placeholders.

LC-QuAD v2.0: Dubey et al. [43] construct the dataset LC-QuAD v2.0, extending the workflow established by Trivedi et al. [131] by crowdsourcing the paraphrasing of generated pseudo-NL questions into improved NL questions. This also includes several rounds of crowd workers generating further paraphrasing of NL questions and performing quality control on others' annotations. An-

[14]The published file only contains 35 templates.
[15]https://project-hobbit.eu/, https://github.com/ag-sc/QALD

other difference is that the initial set of 22 query subgraph templates is constructed not from geometric constraints but from consideration of pre-existing QA datasets.

CFQ: Keysers et al. [69] construct the dataset CFQ, with instances comprising formal queries and NL questions, in a completely rule-based manner. With 239,357 instances, this is to our knowledge the largest KGQA dataset generated entirely with rule-based automation.

KQA Pro: Shi et al. [119] construct the dataset KQA Pro broadly following the Overnight [126] approach of generating formal queries and pseudo-NL questions, and then using crowdsourcing to paraphrase pseudo-NL questions into NL questions, and finally using crowd workers to cross-validate the paraphrases of their colleagues.

GrailQA: Gu et al. [53] also construct the dataset GrailQA in a similar manner as Shi et al. [119], broadly following the Overnight [126] approach, including the use of crowdsourced labor for paraphrasing pseudo-NL questions and cross-validation.

Usage in the Present Work

Of the complex KGQA datasets mentioned, in Chap. 4 we choose to focus on DBNQA [57] because it is the largest complex dataset that has been used for evaluating several NMT-based models [151]. Because of the approach taken in the construction of DBNQA [57], Chap. 4 also uses, but does not directly train or test on LC-QuAD v1.0 [131]. Through our work discussed in Chap. 4, we create and introduce two partitionings of DBNQA called Sanitized-1 DBNQA and Sanitized-2 DBNQA. In Chaps. 5 and 6, we refer to Sanitized-1 DBNQA simply as DBNQA*. DBNQA* [78] is a partitioning of DBNQA [57] into training, validation, and testing splits based on the underlying templates, avoiding leakage of information between training and test splits. The instances are identical to DBNQA, and so we use DBNQA* in our experiments. Chap. 5 additionally works with the datasets LC-QuAD v2.0 [43] and GrailQA [53].

2.5.3 Dataset Construction and Augmentation Approaches

In the previous section we have briefly described the reported construction process of specific KGQA datasets. In this section, we generalize the approaches we have seen. To create a KGQA dataset, first some initial design decisions must be made before instances can be created. Design decisions include:

- Which KG(s) will be addressed?
- Which entities, predicates, and formal query language aggregation operations will be included?
- How many triple patterns may be included in the formal query?
- Which items will each instance comprise? NL question, formal query, answer, or other items?

The naive and effortful approach to construct a KGQA dataset is for one or more experts to make all the design decisions as well as manually performing the work of constructing NL questions and their corresponding ground truth formal queries (or answers). This *fully manual* approach is taken in the construction of the QALD [135] series, as well as the Free917 [27] dataset.

In order to dedicate labor to expertise-appropriate tasks in a more cost-efficient manner, various approaches emerged involving crowdsourced labor and automation (i.e., programs) for steps in the instance generation requiring less narrow expertise. For example, researchers can (create programs to automatically) generate valid formal queries and corresponding template-based pseudo-NL questions, while crowd workers can paraphrase the template-based pseudo-NL questions into more fluent NL questions.

Two further labor-saving steps are (1) using crowd workers to quality control the work of other crowd workers, as in the construction of LC-QuAD v2.0 [43], and (2) using previously generated instances as a *seed* from which to extract templates which in turn are used to generate novel instances, as in the construction of DBNQA [57].

Table 2.2 shows how these different approaches to instance generation apply to a selection of complex KGQA datasets. The table also shows what type of labor was involved in the manual post-processing of the initially generated instances. Out of the various QALD datasets, QALD-7-train [136] is highlighted due to its use among the seed data from which Hartmann et al. [57] extracted templates.

[16]https://project-hobbit.eu/; https://github.com/ag-sc/QALD
[17]http://nlp.cs.tau.ac.il/compwebq; https://github.com/alontalmor/WebAsKB
[18]https://github.com/AKSW/DBNQA

Table 2.2: Selected datasets for complex KGQA.

Dataset	Generation	Manual post-processing
QALD-{1-9}[16]	Manual	N/A
QALD-7-train [136]	Manual	Programmatically filtered
LC-QuAD [131]	Hand-made templates	Paraphrasing† and reviews‡ of NLQs
LC-QuAD 2.0 [43]	Hand-made templates	Paraphrasing† and reviews† of NLQs
ComplexWebQuestions [129][17]	Hand-made templates	Paraphrasing† of NLQs
DBNQA [57][18]	Extracted templates	Reviews‡ of generated templates

†: Non-experts. ‡: Experts. NLQs: NL questions.

The datasets listed in Table 2.2 indicate a trend towards scalable instance generation to economically generate larger volumes of question-query pair data for complex KGQA. The progression started with fully manual dataset generation, moving to using hand-made templates for automated instance generation, and with DBNQA the automated template generation from pre-existing seed datasets. While these changes in automation have increased the scale of available datasets, the need for manual post-processing also increases. Consequently, it becomes economically desirable to divide the post-processing work into (i) work that requires expert knowledge of the formal query language, and (ii) work that can be adequately performed by crowdsourced non-experts. The non-experts only need adequate natural language skills to improve the grammar of template-based NL question generation. These are prevailing assumptions regarding KGQA dataset construction, which we question in Chap. 5.

Synthetic Data Generation and Data Augmentation

To satisfy the need for large volume datasets to train deep learning models, various approaches have been explored to enhance the collection of real data points, such as data augmentation [34, 120] and synthetic data generation [94].[19] Gen-

[19]The distinction between synthetic and augmented data can become ambiguous in some cases, as augmented data means taking data from real measurements and changing the data in some way that preserves key qualities of the data point while challenging the model to learn the preserved relationships. On the one hand "synthetic" data could be considered to include all data that are not the result of direct measurement, which would subsume data augmentation. On the other, "synthetic data" implies that the data is constructed to represent an underlying distribution beyond simply transforming original data with certain invariances. Following the above distinction, DBNQA [57] may represent an ambiguous case, as semantically, the generated instances are all novel as in synthetic data, but syntactically, they are variations that serve to reinforce the shared pattern, as in data augmentation. Our work hopefully elucidates this further.

erating synthetic data to train machine learning models has been done for a large number of tasks. Within the field of information retrieval (IR), synthetic data generation has been explored to train models for various tasks, including ad hoc document retrieval [13], suggesting NL questions to clarify search intent from query terms [40], and query auto-completion [70].

Synthetic data generation has also been used for question answering (QA) tasks [4, 51, 147, 156]. Much effort has focused on the machine reading comprehension (MRC) flavors of QA, where questions should be answered in the context of a prose paragraph. For example, Golub et al. [51] looked at how to improve transfer learning, fine-tuning a model (pre-trained on one source domain MRC dataset) with synthetic MRC data generated from the target domain corpus of context paragraphs. The common approach, also taken by Alberti et al. [4], is to use neural language models to select answer spans from paragraphs, and to generate questions conditioned on the answer and paragraph.

Completely synthetic data generation for KGQA has not yet been investigated in depth. The KGQA datasets in Sect. 2.5.2 are almost all the product of (1) handcrafting instances entirely, (2) automatically generating formal queries and pseudo-NL questions followed by manual paraphrasing to get NL questions, (3) augmenting existing instances in a rule-based manner, or a combination of these. CFQ [69] is a notable exception, being a completely synthetically generated KGQA dataset. Keysers et al. [69] explain their generation algorithm in their paper, but the code implementation is not shared openly. We note that large fully synthetic KGQA datasets that are entirely generated by neural models have not yet been published.

2.6 Evaluation Measures

In the following section, we review work on established evaluation measures for KGQA, as well as work on axiomatic analysis of evaluation measures. KGQA systems have been evaluated using typical IR measures [85], either considering the correctness of answers [19, 82, 117], ranked candidate formal queries generated [41, 84], or in terms of sub-tasks [123], such as entity linking [149] and answer type prediction [93]. In evaluating answers with typical IR measures, set-based measures (e.g., accuracy, precision, recall, and F_1) [19, 37, 41, 101, 117, 149] and ranked-list-based measures (e.g., H@1, MRR) [26] have been used. We note that evaluating answers using a ranked-list approach may not be suitable, since the user relies on the KGQA system to provide a single (definitive) answer as opposed to a list of candidate answers. This may be a consequence of the IR-based approach commonly taken with the task, in order to award partial reward in the evaluation (cf. Sect. 2.4.2).

In our work we focus mainly on the SP-KGQA flavor of KGQA, and in Chap. 6 we devise ways to measure partial success on a single prediction rather than a list of candidate predictions. Some existing SP-KGQA systems and benchmarks also report machine translation-based evaluation measures [125, 151] with respect to the predicted formal query. Those measures, like BLEU [97], focus on n-gram overlap, which is insufficient to capture the complexities of formal queries. Regardless, the appropriateness of these measures has not been addressed to date. Chapter 6 aims to fill that gap.

2.6.1 Established Measures for SP-KGQA

A number of established measures have been used in SP-KGQA: Exact Match, set-based measures (Accuracy, Precision, Recall, and F_1), Perplexity, and BLEU. In addition, the ROUGE family of measures comprise reasonable alternatives to BLEU for SP-KGQA, being similarly based on n-gram overlaps.

Exact Match (EM) [104] is a binary measure, the proportion of instances predicted exactly as given by any single associated ground truth reference. It is used for QA [104] and in some cases for KGQA [42, 151]. For SP-KGQA, it can be considered an overly strict measure, unless each NL question is exhaustively labeled with all possible satisfactory ground truth formal queries.

Set-based measures like Accuracy, Recall (R), Precision (P), and F_1 are well known in an IR context [85], and for KGQA are used to evaluate retrieved answers as overlapping sets of elements [36, 84, 138]. In the case of SP-KGQA or other formal query prediction tasks [42, 151], such set-based measures can consider the formal queries as sets of syntactic elements.

Perplexity (PPL) is a measure of how close a model is to the underlying probability distribution being modeled. An exponentiation of the loss function cross-entropy on text [23], PPL has been reported for KGQA [151]. However, this measure represents the intrinsic quality of the model with respect to the dataset, rather than the performance of the model with respect to an extrinsic task [67].

Bilingual evaluation understudy (BLEU) [97] is an n-gram overlap measure developed for MT and used for SP-KGQA [125, 151]. Technically a family of measures based on a *modified precision*, BLEU most commonly refers to the variant reported as best in [97]. BLEU originally assumed corpus-level evaluation, aggregating over corpus sentences in a manner proportional to their lengths, and not by the arithmetic mean of BLEU for each sentence.

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [75] is another family of n-gram overlap measures, devised for summarization but also used in MT and other text generation tasks [61]. In contrast to BLEU, ROUGE measures are based on R and F_1, and n-grams matched in multiple references are given more weight. We focus on ROUGE-L, defined as an F-measure based on the longest common subsequence of two sequences, i.e., a candidate and a reference. To the best of our knowledge, ROUGE has not been used for SP-KGQA evaluation.

2.6.2 Deriving Measures Axiomatically

There is a solid body of existing research on axiomatically analyzing existing evaluation measures and deriving novel evaluation measures for various information access tasks. In this approach, formal constraints are defined and used to theoretically show which performance measures satisfy each constraint, and hence possess the corresponding quality. Several tasks have been studied in this way, including clustering [6], classification [113], filtering [7], quantification [114], diversification [3, 9, 110], and recommender systems [98]. In addition, the axiomatic methodology itself has been investigated in the context of IR [5] and the properties of IR effectiveness measures have been axiomatically analyzed [25, 46], such as whether they are interval scales [47] and what statistical properties different measures have as a consequence [45]. Finally, work has been done towards constructing general theories of IR effectiveness measurements [8, 48]. In Chap. 6, we make the first attempt to axiomatically derive evaluation measures for the SP-KGQA systems.

2.7 Methods for KGQA

Various methods have been developed to build systems for performing complex KGQA. In Sect. 2.4.2 we described the distinction between the two approaches to the KGQA task, IR-KGQA and SP-KGQA, with their different task definitions. As the definition of a task changes, the method required to accomplish it may also need to change. Conversely, we also argue that the different approaches to the KGQA task are themselves informed by what techniques can be combined in a method to solve the task.

Recent technological developments have brought to the forefront techniques based on neural networks. Therefore, an important distinction among KGQA methods is the one between neural and non-neural methods. The former focuses on the development of suitable neural architectures tending towards end-to-end learning, while the latter tends to decompose the KGQA task into a sequence of discrete subtasks and create purpose-built solutions for each. In this section, a brief background is provided for non-neural methods of KGQA in Sect. 2.7.1. After this, the neural methods used experimentally in the present work are described in greater detail in Sect. 2.7.2.

2.7.1 Non-Neural Methods

As if to indicate the pervasive shift towards neural approaches, Chakraborty et al. [29] refer to non-neural KGQA methods as "traditional" [20, 105, 133]. Diefenbach et al. [38] consider all KGQA tasks to consist of separate stages, all of which must be solved by the KGQA system. Whether approached as an IR-based or SP-based task, KGQA is still being studied as a problem with distinct sub-tasks to be solved in modular manner [72, 82]. For example, the message-passing architecture QAmp [138] can be considered a hybrid of neural and non-neural approaches, but is not an end-to-end neural system. QAmp uses neural components (RNN classifiers, word embeddings) in its question interpretation stage, in addition to BM25 and embeddings in its candidate answer ranking stage. In the answer inference stage, probabilistic graphical model approaches are used. Another hybrid approach was devised by Abujabal et al. [2], who separated the semantic parsing for KGQA into various components: an offline neural component that learned from KGQA data to generate the syntactic template components that were used compositionally to parse the semantics of an NL question into a formal query; another component that generates candidate queries; and a component to rank the candidate queries to output the inferred formal query.

2.7.2 Neural Architectures

In the present work, the methods used experimentally for KGQA are exclusively neural architectures. In particular, the methods have primarily been neural architectures following the encoder-decoder pattern introduced for sequence-to-sequence tasks [31, 128]. These neural architectures were previously used for neural machine translation (NMT) between two different natural languages. SP-KGQA can then be approached by analogy as a translation from an NL question to a formal query. At the time of our experiments, the respective methods used represent the state of the art in terms of neural methods for KGQA.

Given that KGQA can be cast as a semantic parsing task, this makes sense, although the strict syntax of the formal query language differs from the more varied and flexible syntax in natural languages. As shown by Yin et al. [151], a number of NMT KGQA architectures can be successfully trained on SP-KGQA data, i.e., NL question and SPARQL query pairs. In the present work, the following neural architectures are used as methods for SP-KGQA:

Neural SPARQL Machine (NSpM) Soru et al. [124, 125] introduce an NMT-inspired approach to SP-KGQA, training their RNN-based sequence-to-sequence model end-to-end to translate NL questions to SPARQL queries. The architecture is a 2-layer Long Short Term Memory (LSTM) RNN without any attention mechanism [151].

NSpM+Att1 This architecture differs from NSpM only in that a global Bahdanau attention mechanism is added [14].

NSpM+Att2 This architecture differs from NSpM only in that a local Luong attention mechanism is added [83].

ConvS2S With this architecture, Gehring et al. [50] introduced the use of CNNs in the encoder-decoder pattern. This architecture also includes a multi-step attention mechanism where the aforementioned Bahdanau and Luong attention mechanisms are classified as single-step attention mechanisms [50].

Transformer Vaswani et al. [140] introduced the revolutionary self-attention mechanism which improved many text-to-text tasks and lead to a proliferation of variants, including the famous pre-trained BERT [35] and its even larger, but still Transformer-based successors such as GPT-3 [24].

GrailQA Transduction+BERT Gu et al. [53] use an LSTM-based encoder-decoder pattern, but use BERT [35] to encode the NL question together with schema items from the KG.

In addition to the above SP-KGQA methods used in the present work, a variant of the GrailQA model was used experimentally in Chap. 5 and represents an example of IR-KGQA.

GrailQA Ranking+BERT Gu et al. [53] use the same LSTM-based encoder-decoder architecture, along with encoding using BERT. However, in this configuration, the sequence-to-sequence model is used to score candidate formal queries, which are then ranked.

Neural KGQA is surveyed in greater detail by Chakraborty et al. [29], including classification and ranking methods, especially for simple KGQA, as well as machine translation methods, which are considered more suitable for complex KGQA.

Chapter 3

Variable Volumes of Training Data

It is held as a truism that deep neural networks require large datasets to train effective models. However, large datasets, especially with high-quality labels, can be expensive to obtain. Because such costs are a challenge in making effective QA models, our main research question is how training data volume impacts QA performance (RQ 1). To answer this main research question, we investigate the subquestions: to what extent QA models improve when trained on a larger volume of data (RQ 1.1), and how sensitive the models are to fractional changes in training data volume (RQ 1.2).

This chapter considers a practical approach to investigating the impact of training dataset size on the performance that can be achieved with various deep neural architectures for the task of answer sentence selection. We investigate a pre-existing implementation of neural architectures for answer sentence selection by truncating the training data to fractions of the original training dataset, to quantify the differences in performance by trained models given different amounts of training data from the same distribution.

Unlike the later chapters, this chapter addresses a simpler flavor of QA and sets the stage for the subsequent chapters focused on the more complex task of KGQA. Here the focus is on simple matching of short texts without complicating factors such as language modeling with sequential dependence between tokens, and without structured data or the use of a formal query language. The data is simpler for answer sentence selection than for KGQA, but so are the evaluation measures (further explored in Chap. 6) and the neural methods used to train models: simple matching between two short NL texts versus taking an NL question as input to generate a formal query on a specific knowledge graph. For practical reasons, this initial simplicity is also helpful. Still, even in this simple flavor of QA we find challenges with machine learning the task.

3.1 Motivation

The impressive performance improvements brought by deep learning applied to certain domains—computer vision, audio speech-to-text, and natural language processing (NLP) [73, 112]—has motivated a great deal of interest to apply deep learning to other domains as well, including information retrieval (IR). However, the performance improvements from deep learning relative to conventional machine learning approaches have depended on increased computational power, larger datasets to learn from, and some developments on the algorithm and architecture level. Of these three factors, large datasets may represent the least tractable challenge faced by those who would apply deep learning to new domains. Quality training data, especially for supervised learning, requires intensive effort to prepare for the actual learning process.

A category of tasks at the intersection of the fields of IR and NLP, question answering (QA) means returning a correct answer sentence in response to a grammatically well-formed, natural language question. In the present chapter, a specific variant of the QA task is considered, namely answer (sentence) selection, the task of matching single-sentence questions with single-sentence answers. The answer selection task is simply: given a question and a predefined set of candidate answers, select the correct answer. Furthermore, when the candidate answers for a given question are all full sentences, the task is called answer *sentence* selection. This task has recently been investigated as a neural IR problem [88, 95, 148].

3.2 Approach

The approach presented in this chapter is practical in that dataset size was manipulated and the effects were evaluated using a pre-existing implementation of multiple neural IR models with a single original dataset. Specifically, this chapter

Table 3.1: Summary of datasets.

	Training					Validation	Testing
	10%	25%	50%	75%	100%		
#Questions	78	209	414	639	857	122	237
#QA pairs	823	2256	4321	6537	8651	1126	2341

presents work on the MatchZoo project [44], where a number of deep neural architectures for text matching have been implemented.[1] Here, answer selection is considered as a form of question answering, where the question text is matched with the text of the correct answer. The original dataset used for training, validation, and testing, was the canonical WikiQA dataset [148]. The performance of the implemented models on a given dataset was characterized in terms of Mean Average Precision (MAP) over the candidate answer rankings for each question in that dataset.

3.2.1 Data Preparation

The training dataset was filtered to provide the models being trained with meaningfully labelled training data. The filtering rule was simply to omit any question and its associated set of candidate answer sentences if the set of candidate answer sentences did not include both true and false candidates.

Table 3.1 summarizes the datasets used in the training of the various models. Note that the same validation and test sets were used throughout, while the training dataset used was systematically varied between the original (filtered) training set (100%), and various partial training sets truncated to 10%, 25%, 50%, and 75% of the original (filtered) training set. These partial training sets were made by randomly sampling (without replacement) on the questions in the original (filtered) training set. Each selected question was then included in the respective partial training set along with all corresponding candidate answers and their labels. The percentages thus represent the probability for each question to be included in each partial dataset. However, once the random sub-sampling was accomplished, these partial training sets were fixed. Each of the models was then trained five times independently on each dataset size.

[1] https://github.com/faneshion/matchzoo

3.2.2 Models

Ten models were able to train and perform nominally with the code provided by the MatchZoo project [44]. The ten models investigated in the present chapter comprised:

- **Deep Structured Semantic Model (DSSM)** [63], which extends latent semantic analysis with deep architectures; a seminal work on neural IR.

- **Convolutional Deep Structured Semantic Model (CDSSM)** [118], which uses a convolutional neural network (CNN) to extend DSSM with contextual information at the word n-gram level.

- **Architecture-I (ARC-I)** [60], an extension of CDSSM whereby siamese CNNs learn to represent two sentences, deferring matching of sentence pairs to a final multi-layer perceptron (MLP).

- **Architecture-II (ARC-II)** [60], an alternative to ARC-I where sentences interact by 1D convolution before proceeding through a 2D CNN component which is purported to learn both the representation of the indvidual sentences, as well as the structure of their relationship. Again, matching of the representations is determined by a final MLP.

- **Multiple Positional Sentence Representations (MV-LSTM)** [142], follows the aforementioned models by capturing local information on multiple levels of granularity within a sentence, using bidirectional long short-term memory networks (bi-LSTMs) to represent input sentences, modeling interactions with a similarity function (tensor layer), and aggregating interactions with k-Max Pooling before a final MLP to match the obtained representations.

- **Deep Relevance Matching Model (DRMM)** [55], distinguishes relevance matching from semantic matching, using pre-trained neural embeddings of terms and building up fixed-length matching histograms from variable-length local interactions between each query term and document. Each query term matching histogram is passed through a matching MLP, and the overall score is aggregated with a query term gate—a softmax function over all terms in that query.

- **Attention-based Neural Matching Model (aNMM)** [146], which follows a similar structure as ARC-II, except instead of position-shared weighting, aNMM has adopted a value-shared weighting scheme "to learn the importance of different levels of matching signals," and incorporated a query term gate similar to that used in DRMM.

- **Combined Local and Distributed Representations (DUET)** [89], which aims to combine local exact matching with embeddings of query-document pairs in semantic space. This relevance matching is enabled by both the local and distributed models, hence a "duet" of two parallel neural models. The final matching score is simply the sum of the two outputs.

- **MatchPyramid** [96], which uses a matching matrix layer to evaluate pairwise term similarity between two texts, followed by 2D convolutional and pooling layers, with a final matching MLP.

- **DRMM_TKS** [44], which is a variant of DRMM provided by the MatchZoo project, for matching short texts. The architecture is simply described by "Specifically, the matching histogram is replaced by a top-k max pooling layer and the remaining parts are fixed."

Some of these models are motivated more by ad hoc search and document retrieval, whereas others were developed specifically for answer selection and the similar task of sentence completion. However, the commonality is that all the models are designed for text matching.

3.3 Experiments and Results

The following experimental results show the effect of varying training set size.

3.3.1 Final Performance of Trained Models

Figure 3.1 presents the performance on the test dataset of the different models after training for 400 iterations on datasets of various sizes. These figures show that aside from the DSSM, CDSSM, and possibly MatchPyramid models, some improvement does appear to happen with greater training dataset sizes. However, by having an order of magnitude more training data (10% to 100%), only

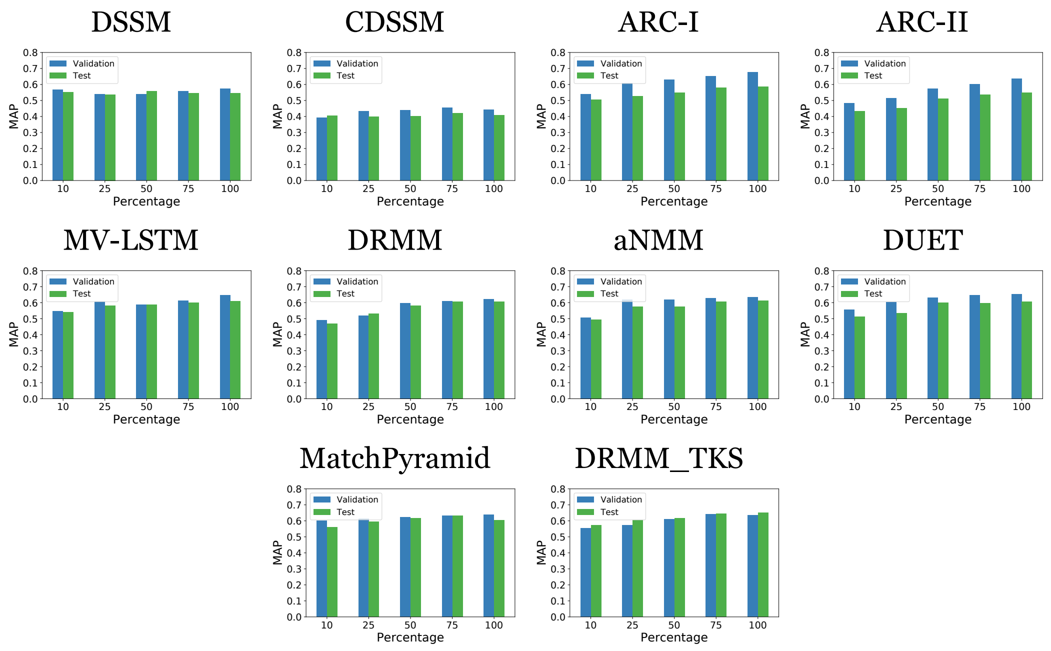


Figure 3.1: Performance (as measured by mean average precision) on the validation (blue) and test (green) datasets with different training dataset sizes.

three models, CDSSM, ARC-II, and DRMM_TKS, achieve a relative improvement above 20%. Four more models, DSSM, MV-LSTM, aNMM, and DUET manage to achieve a relative improvement above 10%. For DRMM, performance even slightly decreases (by 1%). The relative improvements after having doubled (25% to 50%), tripled (25% to 75%), or quadrupled (25% to 100%) the training data size are similarly moderate for most models. Specifically, after doubling, only CDSSM and aNMM showed relative improvement above 10%, and with tripling and quadrupling, only DSSM, CDSSM, ARC-II, and aNMM showed relative improvement above 10%.

3.3.2 Model Training Histories

Figure 3.2 illustrates the relationship, for each model, between the size of the training dataset and performance improvements over the course of training. We can see that most models either reach a plateau or approximately monotonically increase on the training set (shown in blue curves in Fig. 3.2) within the recorded training history. There are, however, a few exceptions, namely DRMM and aNMM,

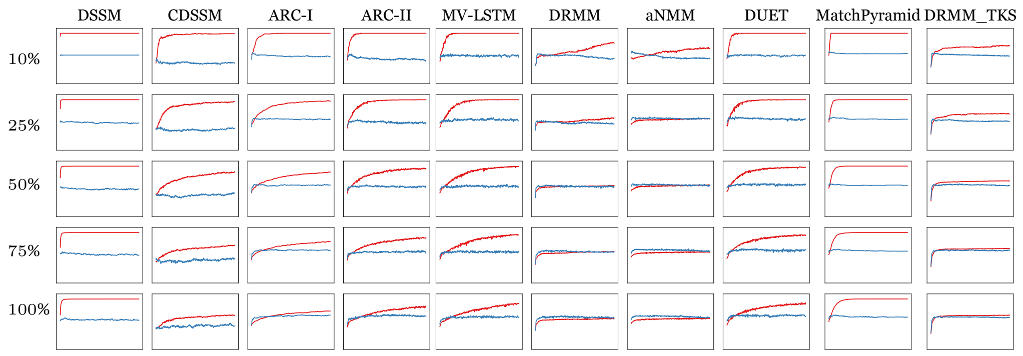


Figure 3.2: Training histories for various models (columns) with varying training dataset size (rows). The red and blue lines correspond to performance for training and validation datasets, respectively. Performance is measured in terms of MAP, and indicated with respect to the y-axes, which range from 0 to 1. The x-axes indicate the number of training iterations (epochs), and range from 0 to 399. The x- and y-axes are identically scaled in each of the sub-plots.

which do not exhibit this desired behavior. Another outlier is DRMM_TKS, which improves at a drastically slow rate. It is also worth pointing out that the models DSSM and MatchPyramid overfit very quickly. This may suggest a memorization effect.

Looking at the MAP scores on the validation set (shown in blue curves in Fig. 3.2), we see a discrepancy from expected behavior. The desired behavior would be that these follow the same monotonically increasing trend as the red lines, with the gap between the two lines decreasing as the amount of training data increases. Most of the models, however, do not behave like that. The validation lines plateau out quickly for most models, or even degrade (DRMM, aNMM).

3.4 Summary

We have briefly looked at the effects of dataset size on the neural IR task of answer sentence selection for a number of deep architectures. The consequences of reducing the available training data logarithmically (10% versus 100%) are discernible, and indicate primarily a failure to generalize. This can be seen from the discrepancy between performance improvement on training data, compared to the modest improvements on validation data. Note that these findings are based on one particular implementation, and the inner workings of the implementation were not rigorously analyzed and verified, but were assumed to correctly enact the cited algorithms.

These findings show that when choosing algorithms and strategies in regard to data volume, there are factors which must be considered beyond the reported benchmarks of fully trained models. The actual performance of the models during different stages of training, relative to different scales of training data, must be considered to discover any unexpected trends.

Furthermore, performance on validation sets is clearly a very important basis for comparison, to gain an intuition about how fast models generalize from different volumes of training data, and with different numbers of training epochs.

Future work may consist of a deeper investigation into the reproducibility of answer selection state-of-the-art results, as well as into quantifying the relationship between training dataset size and the impact of diverse neural models on generalizability.

Despite the fact that answer sentence selection is a simpler flavor of QA, we see that state-of-the-art models are challenged to learn their task from data. The theme of data volume variation (i.e., RQ 1) is carried forward later in this thesis, where experiments involve training KGQA models (Chap. 4 and Chap. 6). We investigate the impact on model performance by training only on different fractions of a training split, where training occurs over a fixed number of epochs or training steps. This offers a way to consider the impact of prospective additional volumes of training data. In turn, this gives an indication of how worthwhile additional data collection or data generation would be for a given method. We speculate that this is also a useful tool to go beyond optimizing model performance for a task and to begin considering the fitness of a method for that task.

Chapter 4

Information Leakage Across Train and Test Splits

Synthetic data generation is important to training and evaluating neural models for question answering over knowledge graphs. The quality of the data and the partitioning of the datasets into training, validation, and test splits impact the performance of the models trained on this data. If the synthetic data generation depends on templates, as is the predominant approach for this task, there may hypothetically be a leakage of information via a shared basis of templates across data splits if the partitioning is not performed hygienically.

This leads to our main research question which we address in this chapter, whether it is appropriate to include in the test split instances that were generated from templates that are also used to generate training split (RQ 2). The following subquestions therefore arise: whether KGQA model performance is affected by whether testing templates are *seen* or *unseen* during training (RQ 2.1), whether generalization to unseen templates is affected by the volume of training data (RQ 2.2), and whether the proportion of unseen test templates to seen training templates affects the performance (RQ 2.3).

This chapter investigates the extent of such information leakage across data splits, and the ability of trained models to generalize to test data when the leakage is controlled. We find that information leakage indeed occurs and that it affects performance. At the same time, the trained models do generalize to test data under the sanitized partitioning presented here. Importantly, these findings extend beyond the particular flavor of question answering task we studied and raise a series of difficult questions around template-based synthetic data generation that will necessitate additional research.

4.1 Motivation

Synthetic data generation can benefit neural models by producing adequate volumes of training data. Knowledge graph question answering (KGQA)—the problem of mapping natural language questions to SPARQL queries—is a task where deep neural models have recently been introduced. Neural models for KGQA by their high-volume data requirements bring about the need for synthetic data generation. However, unlike for other tasks like ad hoc document retrieval [13], query clarification terms [40], or query auto-completion [70], synthetic data generation for KGQA has so far been developed in a template-based manner. This raises a number of interesting methodological questions.

In particular, we consider a hypothesis that training models on template-based synthetic data instances may result in *information leakage* if the partitioning of synthetic data into training, validation, and test splits is not done carefully. If the training and test splits are randomly partitioned without regard for the underlying templates, it is possible that a significant portion of the performance seen in trained models is not coming from correct generalizations. Instead, some portion of the observed performance may come from memorizing the underlying patterns of the finite set of templates, which are common across the training, validation, and testing splits. This *leaky* partitioning condition is illustrated in the top part of Fig. 4.1. To explore the hypothesis, we devised an alternative, *sanitized* partitioning scheme, illustrated in the bottom part of Fig. 4.1.

As a guide to intuition, we can imagine that the KGQA models trained on template-based instances will "see" through the instance to the underlying template, which is therefore considered *seen* with respect to the trained model. The question of the trained models' ability to generalize can then be cast as a question of how the trained models perform on instances generated from *unseen* templates.

To address our research questions, then, we specifically look at complex KGQA, which is a variant of KGQA where the formal query represents a multi-relation subgraph on the knowledge graph (KG). We investigate the properties of synthetic

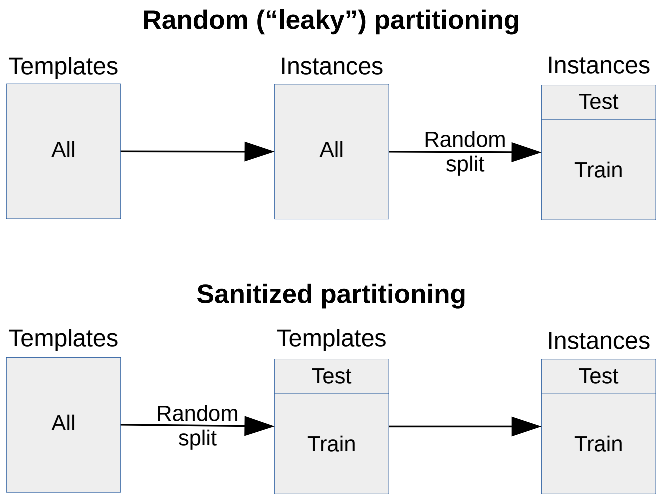


Figure 4.1: Illustration of leaky and sanitized partitioning for template-based synthetic data generation.

data in the context of neural network models, using the largest KGQA dataset that exists to date, DBpedia Neural Question Answering (DBNQA) [57]. We empirically compare three neural machine translation (NMT) architectures that represent a specific family of neural network architectures, recurrent neural networks (RNNs), which were shown to be effective on this task [29, 124, 125].

In the *leaky* partitioning, instances are randomly assigned to splits without regard for underlying templates. This *leaky* partitioning is both convenient and provides the models with the maximum volume and variety of training instances. In the *sanitized* partitioning, templates are partitioned into train and test splits, and instances are then partitioned so that test instances will only be those generated from *unseen* templates. If the synthetic data is not generated in a *sanitized* manner initially, the *sanitized* partitioning requires additional processing to achieve: first templates must be partitioned into test and training splits; then the generated instances must be matched with the templates they were generated from; and finally, the instances must be allocated to test and training splits, accordingly. Nevertheless, this approach helps minimize information leakage when testing model performance.

4.2 Approach

In this section, we describe the approach taken in preparing the shared basis for the various individual experiments which then are described in Sect. 4.3. In Sect. 4.2.1, we define the scope of KGQA methods to be used, and in Sect. 4.2.2 the structure and generation process for the datasets are described. In Sect. 4.2.3 the original partitioning of the dataset is discussed, while in Sect. 4.2.4 we describe the two sanitized partitionings of the dataset.

4.2.1 Scope

Out of the available approaches to KGQA, we consider only neural machine translation (NMT) architectures, which are interpreted as performing *semantic parsing* on the NL question q to produce a semantically equivalent formal (SPARQL) query f that also executes on the target knowledge graph $\mathcal{K}$, retrieving the correct answer a. In the present work, we limit ourself to a particular baseline architecture and its variants, to ensure the comparability of the obtained results. We note that the same experiments can be performed with additional architectures in the future. Specifically, the baseline architecture is taken from [151] (originally from [124, 125]), as well as two attention-based variations of this architecture, which in [151] performed well on DBNQA. The work of Yin et al. [151] was taken as a starting point because it was the only work that considered a variety of NMT architectures applied to the largest available complex KGQA dataset, DBNQA [57]. The selection was made both due to the high performance on the randomly partitioned DBNQA, as well as the fact that these models were implemented in the same framework, Tensorflow. The three models used are:

NSpM The baseline architecture is here referred to as **NSpM** following [124, 125, 151]. However, it is a basic Tensorflow NMT architecture, with 2 layers, 128 units per layer, a dropout rate of 20%, and optimizing on the BLEU metric.

NSpM+Att1 The second architecture is called **NSpM+Att1**, again following [151], and it differs from NSpM only in that a global Bahdanau attention mechanism is added [14]. Since the type of global Bahdanau attention mechanism utilized in [151] was not further specified, the present work selected a "normed" variant.

NSpM+Att2 The third architecture is called **NSpM+Att2**, again following [151], and it differs from NSpM only in that a local Luong attention mechanism is added [83]. Since the type of local Luong attention mechanism utilized in [151] was not further specified, the present work selected a "scaled" variant. This architecture had the second-best performance of the 8 architectures evaluated in [151], second only to the Convolutional sequence-to-sequence architecture **ConvS2S**, and even that difference was relatively slight.

We are looking at a dataset [57] where instances were generated with templates extracted from seeds [131, 136]. The evaluation of this synthetic dataset was done with randomly partitioned training, validation, and test splits [151]. This random partitioning did not avoid allocating instances generated from the same template to different splits. Thus, models trained and evaluated on this random partitioning would see "familiar" instances in the validation and test splits, i.e., instances generated from the same template as instances used for training that model. Could this have created an information leakage, whereby the trained models have memorized a finite set of underlying templates—those *seen* during training—rather than learning to generalize from training instances to previously *unseen* patterns?

To answer this question, we have designed a method to *sanitize*[1] the existing dataset, and ensure that a held-out test split contains only instances generated from a held-out split of templates. Since we are working with a pre-existing dataset with no labelling or index of which template generated each instance, the sanitation process is inevitably somewhat uncertain and depends on constructing a reasonable set of rules to identify which template generated an instance. We make a best effort to recover this information, that is, the originating template for each instance.

We can consider that the two approaches taken, shown in Fig. 4.2, the fully random partitioning by Yin et al. [151], and the sanitized partitioning in the present work, may represent two extremes in how template-based synthetic data should be treated. This perspective is further developed in Sect. 4.4.

[1]This term might be value-laden and undescriptive of the mechanics employed, but reflects the intention to remove or minimize any contamination due to information leakage.

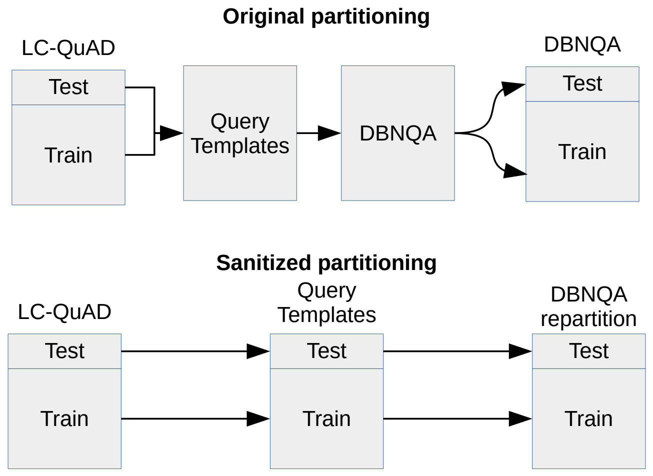


Figure 4.2: Illustration of original and sanitized DBNQA partitioning.

4.2.2 Preliminaries

The pipeline utilized by Hartmann et al. [57] to generate a large volume of KGQA training data can be considered as three discrete stages, which we refer to as *seeds*, *templates*, and *instances*. First, a small high-quality KGQA dataset consisting of question-query pairs is taken as the seed dataset $s \in \mathcal{S}$ from which are extracted templates $t \in \mathcal{T}$, capturing the underlying pattern of the seed data points, but replacing certain parts of the seed data points with placeholder tokens or URIs, for NL question and SPARQL forms, respectively. The templates are then instantiated into concrete data points, replacing the placeholders in a template with appropriate terms (entity labels) or entity URIs. Each template can be used to generate an arbitrary number of such new instances $i \in \mathcal{I}$, bounded only by the availability of unique paths (subgraphs) on the knowledge graph that fit the path(s) of the template. The different stages are illustrated in Table 4.1, with a pair of NL question and SPARQL forms for each stage. The examples are chosen such that the template is derived from the seed, and the instances are both generated from the same template. Two example instances are shown to illustrate the similarities of instances generated from the same template.

Table 4.1: Examples of seed, template, and instance. (NLQ stands for natural language question.)

Seed	NLQ	*Is Peter Piper Pizza in the pizza industry?*
	SPARQL	`ASK WHERE { <http://dbpedia.org/resource/Peter_Piper_Pizza>`
		`<http://dbpedia.org/ontology/industry>`
		`<http://dbpedia.org/resource/Pizza> }`
Template	NLQ	*Is <B> in the <A> industry?*
	SPARQL	`SELECT DISTINCT ?a, ?b WHERE {`
		`?b <http://dbpedia.org/ontology/industry> ?a }`
Instance 1	NLQ	*Is robot comics in the publishing industry?*
	SPARQL	`ASK WHERE { <http://dbpedia.org/resource/Robot_Comics>`
		`<http://dbpedia.org/ontology/industry>`
		`<http://dbpedia.org/resource/Publishing> }`
Instance 2	NLQ	*Is tiger aircraft in the aerospace industry?*
	SPARQL	`ASK WHERE { <http://dbpedia.org/resource/Tiger_Aircraft>`
		`<http://dbpedia.org/ontology/industry>`
		`<http://dbpedia.org/resource/Aerospace> }`

4.2.3 Original DBNQA

The DBNQA dataset is provided without any canonical partitions [57]. Researchers are free to randomly partition the dataset into training, validation, and testing splits. This was done by Yin et al. [151], who reported allocating 80%-$10\% - 10\%$ to the respective splits. However, their unique partitioning is not recoverable from their paper or code repository. The present work randomly partitioned DBNQA in the same proportions, but used specific random seeds. In order to ensure that any differences were not due to random chance, this random partitioning was done five times with different random seeds each time, and the resulting partitions were used in Experiment 1 (see Sect.4.3.1) to separately train models of each of the three NSpM architectures discussed in Sect. 2.7.2.

4.2.4 Sanitized DBNQA

In order to investigate the question of information leakage via templates, the train-and-test-splits partitioning of the major part of the seed dataset, LC-QuAD, was used to coordinate a partitioning of the LC-QuAD-based templates dataset previously used in generating DBNQA. Subsequently, the partitioned templates were used to partition the instances dataset, DBNQA. This repartitioning is illustrated in the bottom half of Fig. 4.2. Templates were assigned to the template test split $\mathcal{T}_{\text{test}} \subset \mathcal{T}$ if the NL question forms of both the seed and template were

Table 4.2: Overview of dataset splits used in our experiments. Five different random splits of original DBNQA were used with these proportions. Sanitized-1 DBNQA and Sanitized-2 DBNQA were based on 20% and 10% test splits in the LC-QuAD seed set, respectively.

Dataset	**Train**	**Validation**	**Test**
Original DBNQA	715 600 (80.0%)	89 449 (10.0%)	89 450 (10.0%)
Sanitized-1 DBNQA	659 313 (74.8%)	73 257 (8.3%)	148 397 (16.8%)
Sanitized-2 DBNQA	726 355 (82.4%)	80 706 (9.2%)	73 906 (8.4%)

identical except where the template placeholders allow a contiguous sequence of tokens in the seed, and if the predicates in the seed SPARQL are the same as those in the template SPARQL. Similarly, instances were assigned to the instance test split $\mathcal{I}_{\text{test}}$ if the NL question forms match as above, and if all the predicates in the template SPARQL are also in the instance SPARQL, in the same order.

The datasets were also de-duplicated at each stage. The original and sanitized instance datasets are summarized in Table 4.2. Sanitized-1 DBNQA was based on the canonical split of LC-QuAD into an 80% training split and a 20% test split. Sanitized-2 DBNQA was based on a 90%-10% split of LC-QuAD. Only after repartitioning in this systematic manner based on template matching is the instance training split itself randomly partitioned into a 90% training split and a 10% validation split. Thus, the test split is sanitized with respect to the training split, while the validation split is not. By evaluating trained models on both the test split and validation split, we illustrate the information leakage via templates caused by a purely random partitioning of an instance dataset like DBNQA. Having thus repartitioned the instances, we trained KGQA NMT models as done by Yin et al. [151], both reproducing the approach taken by Yin, et al. with the original DBNQA partitionings described in Sect. 4.2.3, as well as on the repartitioned datasets, to compare the results of training with and without information leakage across the dataset splits.

4.3 Experiments

This section presents a series of experiments we performed on the original and sanitized DBNQA collections, to answer our research questions. Model perfor-

Table 4.3: Results of comparing original DBNQA and Sanitized-1 DBNQA to train and evaluate models.

Architecture	**Original DBNQA**				**Sanitized-1 DBNQA**			
	BLEU		**Perplexity**		**BLEU**		**Perplexity**	
	Valid.	Test	Valid.	Test	Valid.	Test	Valid.	Test
NSpM	62.56 ± 0.10	62.52 ± 0.10	2.37 ± 0.01	2.37 ± 0.01	64.96	41.12	2.33	11.86
NSpM+Att1	79.27 ± 1.82	79.22 ± 1.77	1.58 ± 0.06	1.58 ± 0.06	85.08	54.39	1.60	7.73
NSpM+Att2	80.58 ± 0.95	80.53 ± 0.89	1.54 ± 0.03	1.54 ± 0.02	84.67	54.55	1.61	9.01

mance is evaluated in terms of the metrics BLEU [97] and perplexity [23], for comparison with results reported by Yin et al. [151]. These measures are commonly used in machine translation evaluation, especially BLEU, which reflects how well the predicted output matches with the ground truth. Perplexity reflects the degree of surprise caused by model predictions compared to the ground truth. For perplexity, unlike BLEU, a higher value represents worse model performance.

4.3.1 Sanitized Data Partitioning

Is the performance of trained models affected by the sanitized data partitioning? We addressed this question with our first experiment, which compared models trained on either the original DBNQA, or on Sanitized-1 DBNQA, which was partitioned based on the canonical partitioning of LC-QuAD as described in Sect. 4.2.4. The original DBNQA was randomly partitioned five times with unique random seeds but identical proportions between the splits ($80\% - 10\% - 10\%$), and the models were independently trained on each random partition. The performance results of original DBNQA are therefore shown as the mean and $\pm$ standard deviation of the models across the five random partitionings. These results, listed in Table 4.3, show that the performance of the trained models was very similar across the test and validation splits of original DBNQA, as well as the unsanitized validation split of Sanitized-1 DBNQA. However, the sanitized test split showed a significant reduction in performance across all the trained models, in terms of both BLEU and perplexity.

Is the performance of the trained models with respect to the sanitized test split dependent on the amount of training data? This would indicate whether there is some degree of generalization from instances based on templates that have been *seen* before during training, or if there is no discernible generalization at all. We addressed this question with our second experiment, which compared the effects

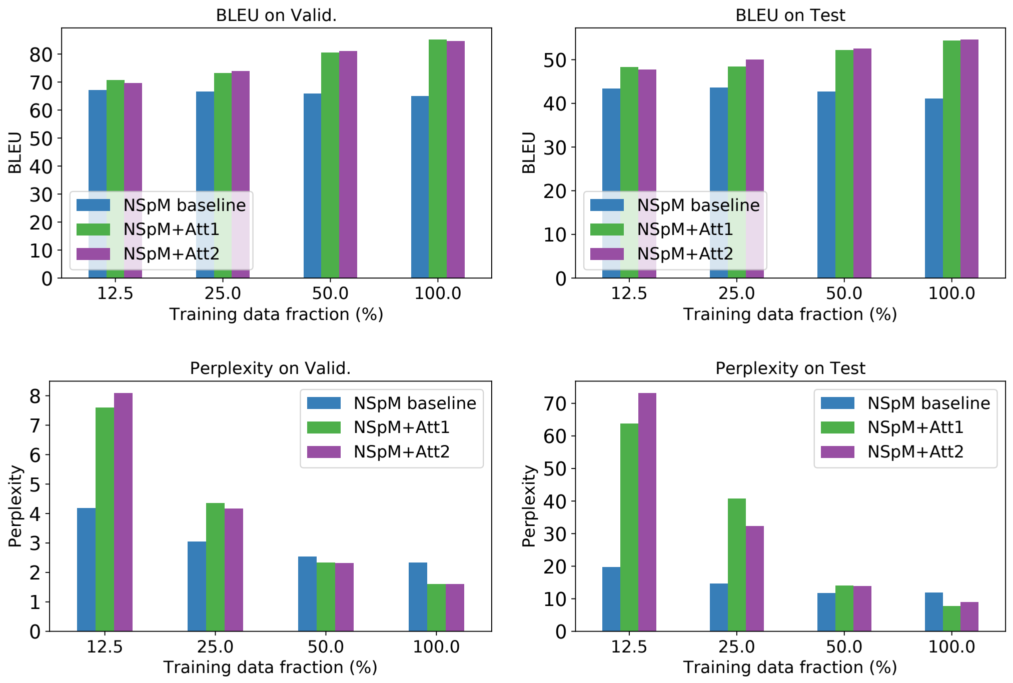


Figure 4.3: Results of comparing models trained on different fractions of the Sanitized-1 DBNQA training split.

of different amounts of training data on the trained models' performance with respect to the Sanitized-1 DBNQA validation and test splits. Although performance increase as a function of increased training data volume is the expected behavior, it has been shown to not always be the case in QA, such as in Chap. 3 [77]. Thus, it is not *a priori* certain that the sanitized test split is similar enough to the sanitized training split that the expected behavior occurs. This experiment verifies whether or not what the models learn generalizes to the sanitized test split *proportionally* to the volume of training data. Thus, the experiment also elucidates whether the models are only learning to memorize the *seen* template patterns, or is learning to generalize to instances from *unseen* templates.

4.3.2 Varying Volumes of Training Data

We trained with fractions of the training data used in our first experiment: 12.5%, 25%, 50%, and 100%. From the results shown in Fig. 4.3, there is a clear trend for

Table 4.4: Results of evaluating models trained on Sanitized-2 DBNQA.

Architecture	**BLEU**		**Perplexity**	
	Validation	Testing	Validation	Testing
NSpM	64.35	42.58	2.36	10.01
NSpM+Att1	82.87	53.09	1.59	7.33
NSpM+Att2	86.05	56.94	1.52	6.57

the models NSpM+Att1 and NSpM+Att2, where increased amounts of training data yield improved model performance on both the sanitized test split and the unsanitized validation split, in terms of both BLEU and perplexity. The NSpM model, on the other hand, does not benefit as much from increased training data, improving in terms of perplexity, but even deteriorating slightly in terms of BLEU. This also holds for the unsanitized validation split, and so reflects on the architecture's ability to improve from training data, not on the sanitized partitioning.

For all models, performance is reduced by the challenge of the sanitized test split, but where performance improves with increased training data, it does so even on the sanitized test split.

4.3.3 Varying Size of Seed Partitions Before Sanitation

Is it possible that different partitions of the seed set can affect the degree of generalization? To address this question, in our third experiment we investigated whether increasing the proportion of templates *seen* via the training instances would translate into improved performance on the sanitized test split. We divided the canonical test split of LC-QuAD in half, and added one half back into the seed training split, before doing the sanitizing procedure, yielding Sanitized-2 DBNQA, as described in Sect. 4.2.4.

As can been seen from results shown in Table 4.4, here as in our second experiment the performance of all models on the unsanitized validation split was generally better than the performance of models trained on the original DBNQA. Models trained on Sanitized-2 DBNQA performed similarly to models trained on Sanitized-1 DBNQA, with some variations on the order of the standard deviations seen for original DBNQA in Table 4.3. We note, however, that all models trained on Sanitized-2 DBNQA performed better in terms of perplexity with respect to the sanitized test split.

4.4 Summary

As we have seen from the datasets presented in Sect. 2.5, there is a trend towards satisfying an important desideratum of machine learning generally, and deep learning in particular: (i) a large-scale training dataset of such quality and variety that it allows the model to observe and learn to predict patterns when presented new data from the same distribution. However, there is an important second desideratum: (ii) a test set that comprises data from the same distribution but which is novel enough to the model so that model performance is due to the model learning the underlying dynamics of the data, rather than memorizing a finite set of patterns.

In the present work, we have questioned whether DBNQA as used in previous work [151] satisfies (ii). Our hypothesis is that there is a leakage of information between the DBNQA training split on the one hand and the validation and test splits on the other, as used by Yin et al. [151]. We argue that Yin et al. [151] have sacrificed (ii) in favor of (i), while in this chapter we considered the other extreme, where (i) is sacrificed in favor of (ii). For future work, we speculate, is there a middle ground that can be reliably found?

In our experiments, we first showed in Sect. 4.3.1 that there is indeed a large difference in performance on the test split of our sanitized DBNQA partitioning, compared to the validation split, which is randomly partitioned in a template-naive manner. From our second experiment in Sect. 4.3.2, we showed that for models that improve with increased volumes of training data, that improvement also generalizes to the sanitized test split. Finally, in our third experiment in Sect. 4.3.3, the models trained on Sanitized-2 DBNQA showed some tendency to improve performance on both validation and test split, indicating generalization from *seen* to *unseen* templates.

Our results raise a set of interesting questions around training models with synthetic data using fair conditions. These are questions raised by the present study that may be the subject for future work: How well do these findings generalize to other model families than those tested here? Of particular interest are the architectures of ConvS2S [50], Transformer [140], and BERT [35]. Can the distinction between memorization and generalized learning be precisely characterized? How can synthetically generated training data be structured to promote learning dynamics (e.g., of a formal syntax) rather a finite set of fixed patterns?

For template-based synthetic data generation, what should be the relationship between training and test splits to fairly evaluate the performance of trained models?

In summary, we have shown that several NMT-based neural KGQA systems have reduced performance on instances generated from templates where the models saw no instances generated from those templates during training. At the same time, the performance on instances from such *unseen* templates did show improvement from increased training data, indicating that some models were able to generalize better with more training data.

We have shown that a significant part of performance in these models as reported by Yin et al. [151] may largely be attributed to the models learning to recognize the underlying patterns of specific templates from which were generated the instances seen during training. In contrast, the ideal NMT KGQA system would learn the underlying syntaxes of the source and target languages and handle unseen patterns according to implicit principles.

Chapter 5

Data Quality and Question Naturalness

Datasets used to train KGQA models that would provide such a service are expensive to construct, both in terms of expert and crowdsourced labor. Typically, crowdsourced labor is used to improve template-based pseudo-natural questions generated from formal queries. However, the resulting datasets often fall short of representing genuinely natural and fluent language.

In this chapter, we therefore investigate our main research question of how we can ensure or enhance NL question naturalness (RQ 3). To address this in turn we consider subquestions regarding: whether NL questions in existing KGQA datasets can be considered as natural (RQ 3.1), what characterizes a high quality NL question (RQ 3.2), and how KGQA model performance fares against genuinely natural questions (RQ 3.3).

To characterize and remedy these shortcomings of the common KGQA dataset construction approach, we create the IQN-KGQA test collection by first sampling questions from existing KGQA datasets and evaluating them with regards to five different aspects of naturalness. Then, the questions are rewritten to improve their fluency. Finally, the performance of existing KGQA models is compared on the original and rewritten versions of the NL questions.

5.1 Motivation

Knowledge Graph Question Answering (KGQA) is an approach to answering users' questions that both harnesses structured data in the form of knowledge graphs (KGs) and also allows the user to articulate their information need in natural language (NL). Training machine learning models for KGQA requires large-scale datasets specific to the KGQA task. Most commonly, such datasets consist of instances that each comprises a formal query (also known as logic form) and a corresponding NL question [72].

In order to construct large KGQA datasets, the work is typically divided into expert and non-expert subtasks which are then assigned to different people. This makes sense economically, but the resulting dataset may have qualitative shortcomings as a result. The formal query is typically constructed by experts or generated synthetically, while the NL questions are typically added by crowdsourced labor tasked with paraphrasing some generated pseudo-natural form of the corresponding formal query [143]. The NL question is thus typically not formulated by the same person who devised the formal query. Critically, this decouples the intent of the formal query from the NL question meant to express that intent. In addition, in large-scale dataset construction, the data is often back-generated from formal queries, that is, the formal query is generated based on available data, and the corresponding NL question is created afterwards. An individual working with a KG for practical reasons would first develop an information need, which may or may not be first expressed as an NL question, and only then construct a formal query to represent that information need. The crowd worker is also not guaranteed to be completely fluent in the specific language that is used in the dataset being constructed. Furthermore, even so-called open-domain KGQA typically consists of questions in a variety of specific domains. If the crowd worker is unfamiliar with this domain, they may not be able to apply the appropriate wording for the underlying domain and categories. We therefore hypothesize that this approach to KGQA dataset construction does not ensure genuinely natural NL questions.

In Table 5.1 we have listed five example questions sampled from existing KGQA datasets, each in their original form and in a rewritten form, generated by additional rounds of crowdsourced paraphrasing and quality control. These are all examples where a KGQA model trained on the original dataset performed per-

Table 5.1: Example questions, each rewritten by crowd workers as a more natural way to express the original question.

Original question From DBNQA [57].	**Rewritten question** From IQN-KGQA [79] (our work).
List the territory of romanian war of independence ?	What territory was involved in the Romanian War of Independence?
Name the nearest city to la laguna lake ?	What is the nearest city to La Laguna Lake?
What is the government type of wallis and futuna ?	What type of government does Wallis and Futuna have?
What is the origin of faberrebe?	What is the origin of the faberrebe grape?
What is the total number of writers whose singles are recorded in ferndale?	How many writers had singles recorded in Ferndale?

fectly on the original question, but completely failed on the rewritten question. This illustrates that some KGQA systems trained on less natural NL questions are not able to address a more naturally phrased version of the same question.

From a machine learning perspective, it is unsurprising that test data from a different distribution than training data may be challenging. However, as the rewritten questions in Table 5.1 illustrate, the KGQA models are failing on more naturally articulated questions. This calls into question whether KGQA models are really learning to perform their nominal task. We investigate how NL questions in KGQA datasets can be considered unnatural, and develop a coding scheme for dimensions of unnaturalness. We determine five dimensions of unnaturalness in NL questions: grammar, form, meaning, answerability, and factuality.

Next, we use our coding scheme in a crowdsourcing context to characterize original NL questions and collect rewritten forms of these NL questions, which are included in our test collection, IQN-KGQA. We sample 250 NL questions from each of three benchmark KGQA datasets: DBNQA [57], LC-QuAD v2.0 [43], and GrailQA [53].

To develop truly effective KGQA systems requires an appreciation of how well these systems fare against realistic questions formulated in genuinely natural language. We apply KGQA models to the original and rewritten questions and see how improved naturalness challenges existing systems. We find that performance drops up to 78% when KGQA models are challenged with the set of rewritten questions.

5.2 Preliminary Analysis

Larger KGQA datasets typically rely on crowdsourcing for generating NL questions from synthetically generated formal queries. We hypothesize that scaling up a KGQA dataset by relying heavily on these two distinct modes comes at the expense of NL question quality, and that the original NL questions may not always be genuinely natural NL questions. To investigate unnaturalness in the NL questions of existing KGQA datasets, we begin by testing that hypothesis on a small sample of instances using expert annotators.

We select three KGQA datasets to sample NL questions from. We choose KGQA datasets that are recent, large, have complex questions and formal queries. We also choose the datasets so that all of the most common KGs are represented in the formal query bindings. Specifically, we consider the datasets DBNQA* [78], LC-QuAD v2.0 [43], and GrailQA [53]. We then randomly sample 25 NL questions from each of these datasets. Specifically, the 25 NL questions are respectively sampled from the entire DBNQA* dataset, and from the train splits of LC-QuAD v2.0 and GrailQA.

Following the approaches of Arguello et al. [11] and Jørgensen and Bogers [65], we perform an *open coding* pass to collect impressions on how the NL questions fall short of being "natural." Three academic researchers are presented each NL question and asked to (i) judge whether or not the question is natural, (ii) produce a (more) natural paraphrase of the question, and (iii) comment on the NL question and suggest any tags or categories regarding "why and how the question is or is not natural." The first author of [79] then collates the responses, and the comments and categories are harmonized into a consistent coding scheme of tags by the first author. Both authors of [79] review the extracted tags and discuss common themes across tags. The tags are then organized into the five dimensions of unnaturalness illustrated along with NL question examples in Table 5.2. We note that the examples in the table may exhibit more than one of the properties that exemplify a given tag or dimension of unnaturalness.

5.3 Data Annotation

Having defined codes to characterize question unnaturalness in KGQA datasets, we next design a protocol for larger-scale data labeling using a two-step crowd-

Table 5.2: Dimensions of question unnaturalness

Dimension	Tag	Example
Grammar	Grammatical errors	Which is {godmother} of {Camillo Benso di Cavour}, whose {craft} is {politician} ?
	Poor flow/word ordering	Who lives in Anita Bryant whose arrondissement is Pittsburg County?
	Non-idiomatic	What is character role of Turandot ?
Form	Quizlike	astronaut gerhard thiele is associated with which space agency?
	Imperative	find beaufort wind force whose wave height is 0.1
	Inconcise	Which is the regression analysis that is used by the logistic regression analysis and contains the word logistic in it's name?
Meaning	Inconsistent domains/categories	Was 6063 jason invented in eugene merle shoemaker
	Overly specific	Which university attended by arturo macapagal was also the alma mater of hector tarrazona ?
	Redundant constraint	What is the death place of the étienne pélabon and is the birthplace of the abeille de perrin?
Answerability	Under-constrained	which organism was born on 1926-06?
	Nonsense/ Unintelligible	what routed drug that a marketed formulation that has a reference form of neurontin 250 solution?
Factuality	Two questions	Who was married to Faye Dunaway and when did it end?
	Descriptive answer expected	What is a crescent?

sourcing pipeline. Crowd workers are first asked to annotate and paraphrase the sampled NL questions. Then, in a separate task, a different set of workers is employed to select the best version of a question from a set, including the original formulations as well as rewritten questions from the first task.

We sample a new set of NL questions, this time 250 NL questions from each of the three datasets. Specifically, we randomly sample the NL questions from the test split of DBNQA* and LC-QuAD v2.0, but from the validation split of GrailQA, since for the latter, the public test split does not include ground truth answers. The resulting test collection is termed IQN-KGQA and is summarized in Table 5.3.

5.3.1 Crowdsourcing: Platform and Workers

Our data annotation was conducted on the Amazon Mechanical Turk platform. For both tasks, workers were required to have a HIT approval rate of 98% with more than 1000 approvals. The payments were set to USD $0.30 and $0.15, respectively, based on the estimated effort demanded for each task.

Crowd workers were not required to have domain knowledge, based on the findings of Dubey et al. [43]. Since only open-domain KGQA datasets are used in

Table 5.3: Summary of the IQN-KGQA collection.

Subset	Split	#Questions	#Rewritten
DBNQA*	test	250	180
LC-QuAD v2.0	test	250	150
GrailQA	validation	250	211
Total		750	541

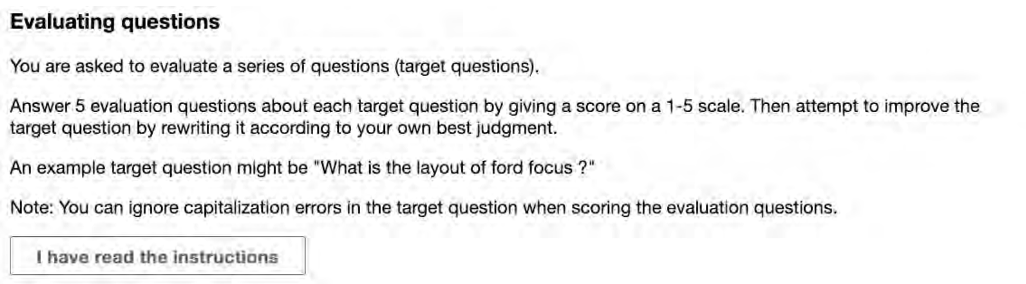
this chapter, the annotation tasks are designed to rely on common sense and English language knowledge primarily. For example, the prompt for the Likert scale "answerablility" is the question "Would you be able to answer this question with the help of a search engine or Wikipedia?" In other words, the data annotation relies on metacognition with respect to an NL question rather than actually finding some answer. Identifying crowd workers with comparable levels of expertise in specific domains prior to data annotation would present a major additional cost. Also, those workers would not necessarily be representative of the general user population whose information needs KGQA datasets aim to capture.

5.3.2 Task 1: Annotate and Rewrite

In the first task, the crowd workers are given one of the sampled NL questions (the target question) and are asked to rate the question in terms of the five dimensions of unnaturalness. For each dimension, the question is rated on a Likert scale. Next, the crowd workers are asked to rewrite the question, to "write a better, more natural and correct version" of the question. Finally, the crowd workers are asked to indicate if they rewrote the question, and if not what the reason was, including a free text field to elaborate on any "other" reason for not rewriting.

The instruction for this task is shown in Fig. 5.1, with examples of the Likert scales in Figs. 5.2 and 5.3. Examples for the question rewriting part of the task are shown in Fig. 5.4, along with examples of cases where for the given reasons a target question could not be rewritten. The form used by crowd workers in Task 1 is shown in Fig. 5.5 and Fig. 5.6.

The responses from crowd workers are then quality controlled, and responses which overtly demonstrate a lack of genuine effort are entirely removed. Criteria for this exclusion include indicating that a question was rewritten but providing



Evaluating questions

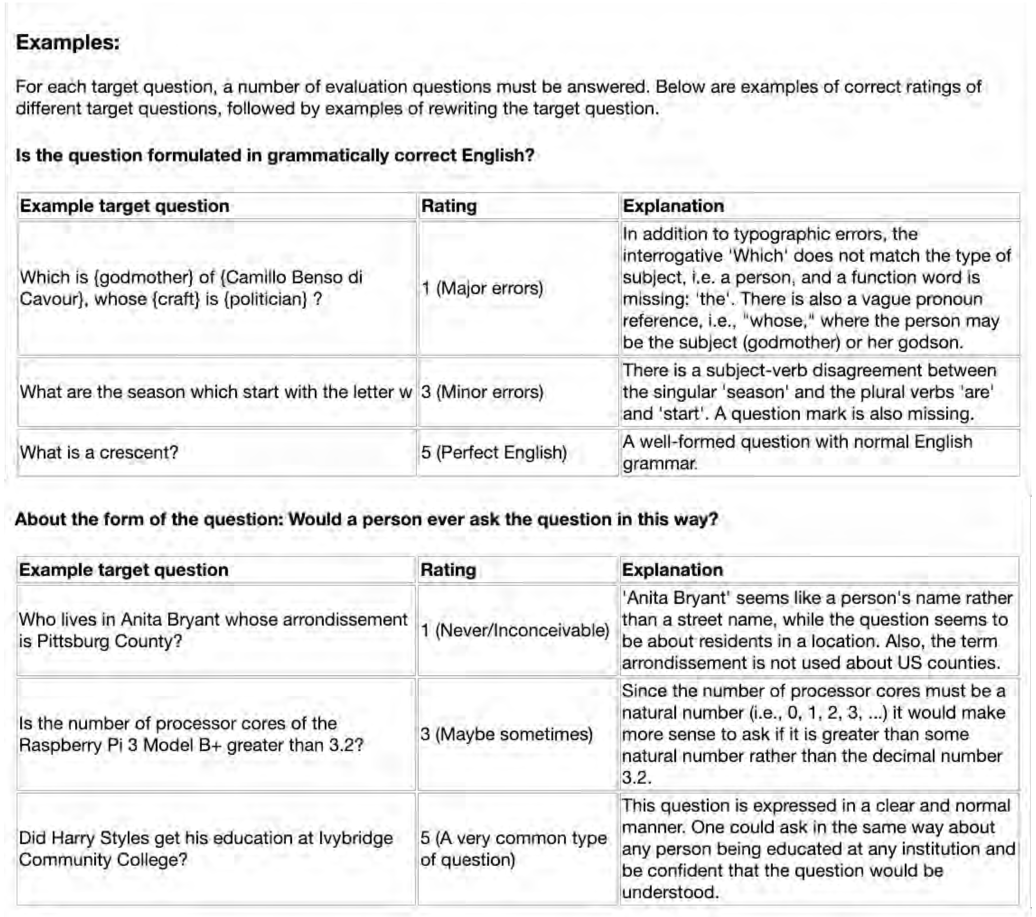
You are asked to evaluate a series of questions (target questions).

Answer 5 evaluation questions about each target question by giving a score on a 1-5 scale. Then attempt to improve the target question by rewriting it according to your own best judgment.

An example target question might be "What is the layout of ford focus ?"

Note: You can ignore capitalization errors in the target question when scoring the evaluation questions.

Figure 5.1: Instructions for crowd workers performing data annotation Task 1.



Examples:

For each target question, a number of evaluation questions must be answered. Below are examples of correct ratings of different target questions, followed by examples of rewriting the target question.

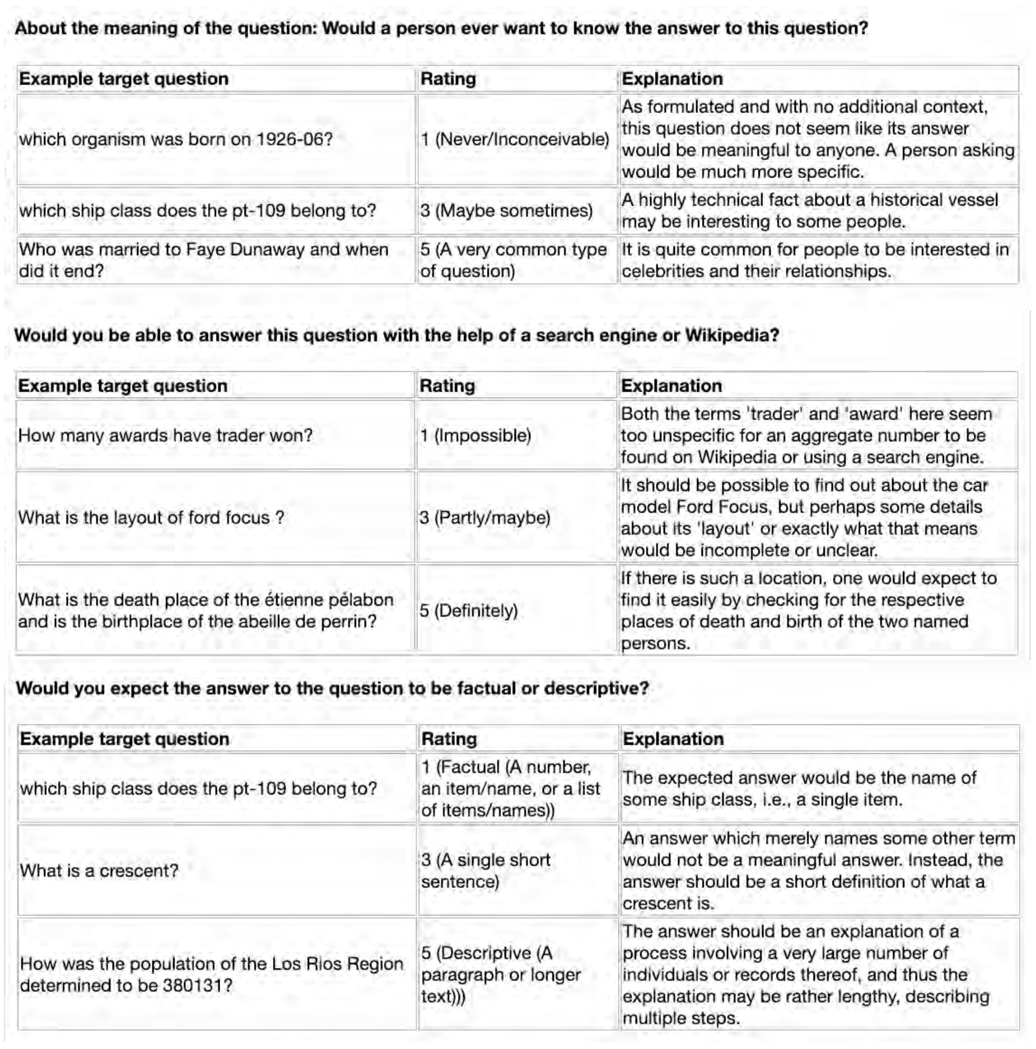
Is the question formulated in grammatically correct English?

Example target question	Rating	Explanation
Which is {godmother} of {Camillo Benso di Cavour}, whose {craft} is {politician} ?	1 (Major errors)	In addition to typographic errors, the interrogative 'Which' does not match the type of subject, i.e. a person, and a function word is missing: 'the'. There is also a vague pronoun reference, i.e., "whose," where the person may be the subject (godmother) or her godson.
What are the season which start with the letter w	3 (Minor errors)	There is a subject-verb disagreement between the singular 'season' and the plural verbs 'are' and 'start'. A question mark is also missing.
What is a crescent?	5 (Perfect English)	A well-formed question with normal English grammar.

About the form of the question: Would a person ever ask the question in this way?

Example target question	Rating	Explanation
Who lives in Anita Bryant whose arrondissement is Pittsburg County?	1 (Never/Inconceivable)	'Anita Bryant' seems like a person's name rather than a street name, while the question seems to be about residents in a location. Also, the term arrondissement is not used about US counties.
Is the number of processor cores of the Raspberry Pi 3 Model B+ greater than 3.2?	3 (Maybe sometimes)	Since the number of processor cores must be a natural number (i.e., 0, 1, 2, 3, ...) it would make more sense to ask if it is greater than some natural number rather than the decimal number 3.2.
Did Harry Styles get his education at Ivybridge Community College?	5 (A very common type of question)	This question is expressed in a clear and normal manner. One could ask in the same way about any person being educated at any institution and be confident that the question would be understood.

Figure 5.2: Likert scale examples presented to crowd workers performing data annotation Task 1. Continues in Fig. 5.3.



About the meaning of the question: Would a person ever want to know the answer to this question?

Example target question	Rating	Explanation
which organism was born on 1926-06?	1 (Never/Inconceivable)	As formulated and with no additional context, this question does not seem like its answer would be meaningful to anyone. A person asking would be much more specific.
which ship class does the pt-109 belong to?	3 (Maybe sometimes)	A highly technical fact about a historical vessel may be interesting to some people.
Who was married to Faye Dunaway and when did it end?	5 (A very common type of question)	It is quite common for people to be interested in celebrities and their relationships.

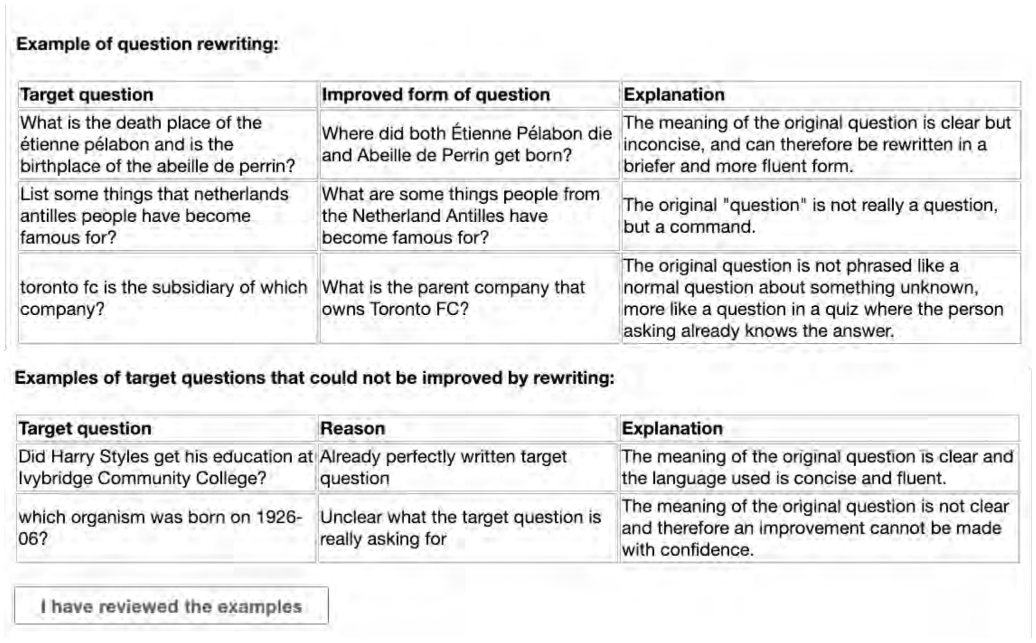
Would you be able to answer this question with the help of a search engine or Wikipedia?

Example target question	Rating	Explanation
How many awards have trader won?	1 (Impossible)	Both the terms 'trader' and 'award' here seem too unspecific for an aggregate number to be found on Wikipedia or using a search engine.
What is the layout of ford focus ?	3 (Partly/maybe)	It should be possible to find out about the car model Ford Focus, but perhaps some details about its 'layout' or exactly what that means would be incomplete or unclear.
What is the death place of the étienne pélabon and is the birthplace of the abeille de perrin?	5 (Definitely)	If there is such a location, one would expect to find it easily by checking for the respective places of death and birth of the two named persons.

Would you expect the answer to the question to be factual or descriptive?

Example target question	Rating	Explanation
which ship class does the pt-109 belong to?	1 (Factual (A number, an item/name, or a list of items/names))	The expected answer would be the name of some ship class, i.e., a single item.
What is a crescent?	3 (A single short sentence)	An answer which merely names some other term would not be a meaningful answer. Instead, the answer should be a short definition of what a crescent is.
How was the population of the Los Rios Region determined to be 380131?	5 (Descriptive (A paragraph or longer text)))	The answer should be an explanation of a process involving a very large number of individuals or records thereof, and thus the explanation may be rather lengthy, describing multiple steps.

Figure 5.3: Likert scale examples presented to crowd workers performing data annotation Task 1. Continued from Fig. 5.2.


Example of question rewriting:

Target question	Improved form of question	Explanation
What is the death place of the étienne pélabon and is the birthplace of the abeille de perrin?	Where did both Étienne Pélabon die and Abeille de Perrin get born?	The meaning of the original question is clear but inconcise, and can therefore be rewritten in a briefer and more fluent form.
List some things that netherlands antilles people have become famous for?	What are some things people from the Netherland Antilles have become famous for?	The original "question" is not really a question, but a command.
toronto fc is the subsidiary of which company?	What is the parent company that owns Toronto FC?	The original question is not phrased like a normal question about something unknown, more like a question in a quiz where the person asking already knows the answer.

Examples of target questions that could not be improved by rewriting:

Target question	Reason	Explanation
Did Harry Styles get his education at Ivybridge Community College?	Already perfectly written target question	The meaning of the original question is clear and the language used is concise and fluent.
which organism was born on 1926-06?	Unclear what the target question is really asking for	The meaning of the original question is not clear and therefore an improvement cannot be made with confidence.

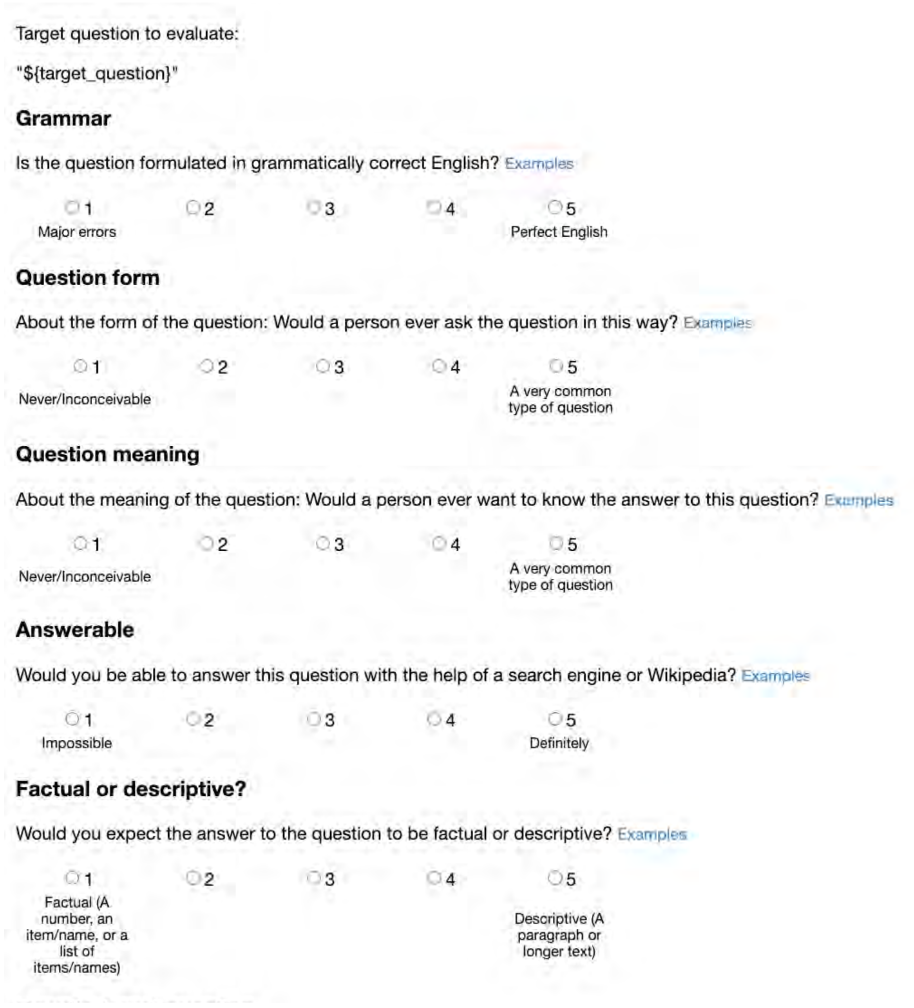
Figure 5.4: Question rewriting examples presented to crowd workers performing data annotation Task 1. Continued from Fig. 5.3.

no paraphrase, writing a short comment like “good” instead of a question, or else copy-pasting parts of the instructions into the rewrite field.

Whenever crowdsourced responses are excluded, additional responses are requested, so that every sampled NL question is annotated (and potentially rewritten) with acceptable responses by at least three different crowd workers.

The results of the Likert scale ratings are shown in Fig. 5.7. The scales are oriented so that the farther to the right the scale lies, the more natural the questions are considered by the crowd workers. The differences between the rows are intuitive since the rows group the responses in terms of the reason given for whether the original question has been rewritten or not. Specifically, the middle row reflects responses where the crowd worker deems the original question to be “already perfect” and hence abstains from rewriting the question. This is also the row with the highest ratings over all five Likert scales.

The bottom two rows also show a consistency between the Likert scale ratings and the reason given why the original question was not rewritten. However, here the ratings are mostly negative compared to the distribution over all responses.



Target question to evaluate:

"${target_question}"

Grammar

Is the question formulated in grammatically correct English? Examples

○ 1	○ 2	○ 3	○ 4	○ 5
Major errors				Perfect English

Question form

About the form of the question: Would a person ever ask the question in this way? Examples

○ 1	○ 2	○ 3	○ 4	○ 5
Never/Inconceivable				A very common type of question

Question meaning

About the meaning of the question: Would a person ever want to know the answer to this question? Examples

○ 1	○ 2	○ 3	○ 4	○ 5
Never/Inconceivable				A very common type of question

Answerable

Would you be able to answer this question with the help of a search engine or Wikipedia? Examples

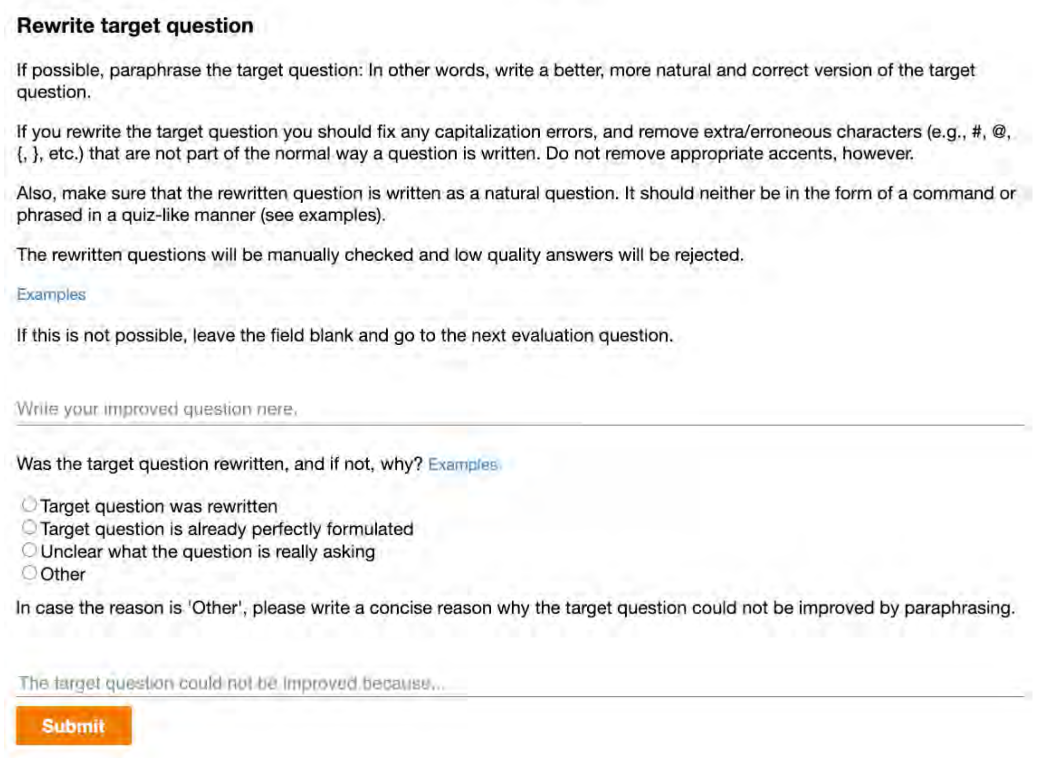
○ 1	○ 2	○ 3	○ 4	○ 5
Impossible				Definitely

Factual or descriptive?

Would you expect the answer to the question to be factual or descriptive? Examples

○ 1	○ 2	○ 3	○ 4	○ 5
Factual (A number, an item/name, or a list of items/names)				Descriptive (A paragraph or longer text)

Figure 5.5: Data annotation form presented to crowd workers performing data annotation Task 1. Continues in Fig. 5.6.



Rewrite target question

If possible, paraphrase the target question: In other words, write a better, more natural and correct version of the target question.

If you rewrite the target question you should fix any capitalization errors, and remove extra/erroneous characters (e.g., #, @, {, }, etc.) that are not part of the normal way a question is written. Do not remove appropriate accents, however.

Also, make sure that the rewritten question is written as a natural question. It should neither be in the form of a command or phrased in a quiz-like manner (see examples).

The rewritten questions will be manually checked and low quality answers will be rejected.

Examples

If this is not possible, leave the field blank and go to the next evaluation question.

Write your improved question here.

Was the target question rewritten, and if not, why? Examples

○ Target question was rewritten
○ Target question is already perfectly formulated
○ Unclear what the question is really asking
○ Other

In case the reason is 'Other', please write a concise reason why the target question could not be improved by paraphrasing.

The target question could not be improved because...

Figure 5.6: Data annotation form presented to crowd workers performing data annotation Task 1. Continued from Fig. 5.5.

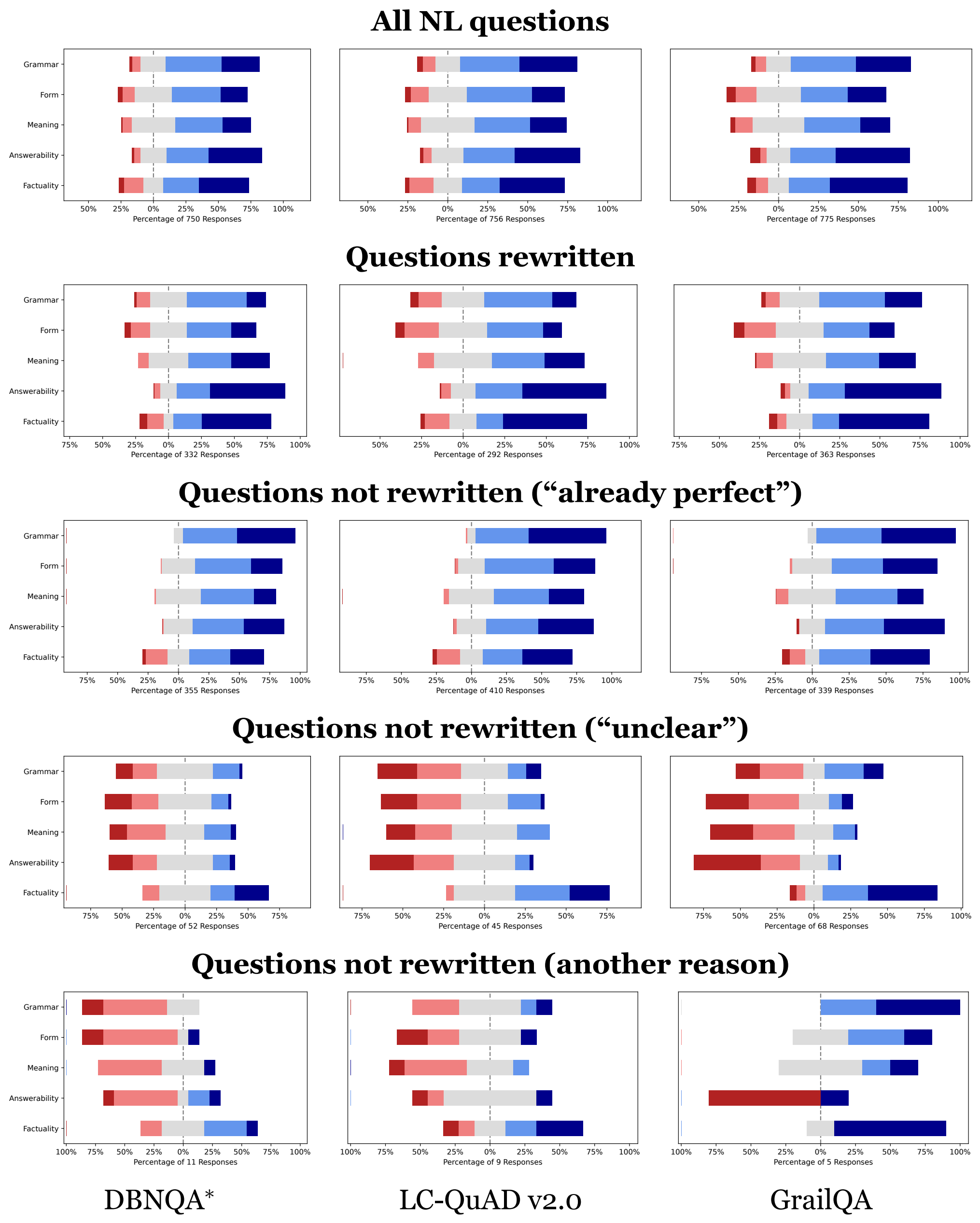


Figure 5.7: Likert scale rating results on all NL questions (Top), on questions which were rewritten (Second row), on questions which were not rewritten because the original question was “already perfect” (Middle), on questions which were “unclear” (Fourth row), and on questions which were not rewritten for another reason (Bottom), broken down per dataset (columns).

The bottom two rows' ratings are also negative compared to the second and middle rows from the top, which reflect the original question having been rewritten or being "already perfect." One interesting exception is that the Likert scale "factuality" is rated highly even in the bottom two rows of Fig. 5.7. It is possible that the distinction was not made clear to the crowd workers between whether a question indicates a very terse and factual answer or a longer, more descriptive one. Alternatively, it may be possible for a question to clearly indicate that its proper answer is factual, but that what is asked is so unclear that the question cannot be improved.

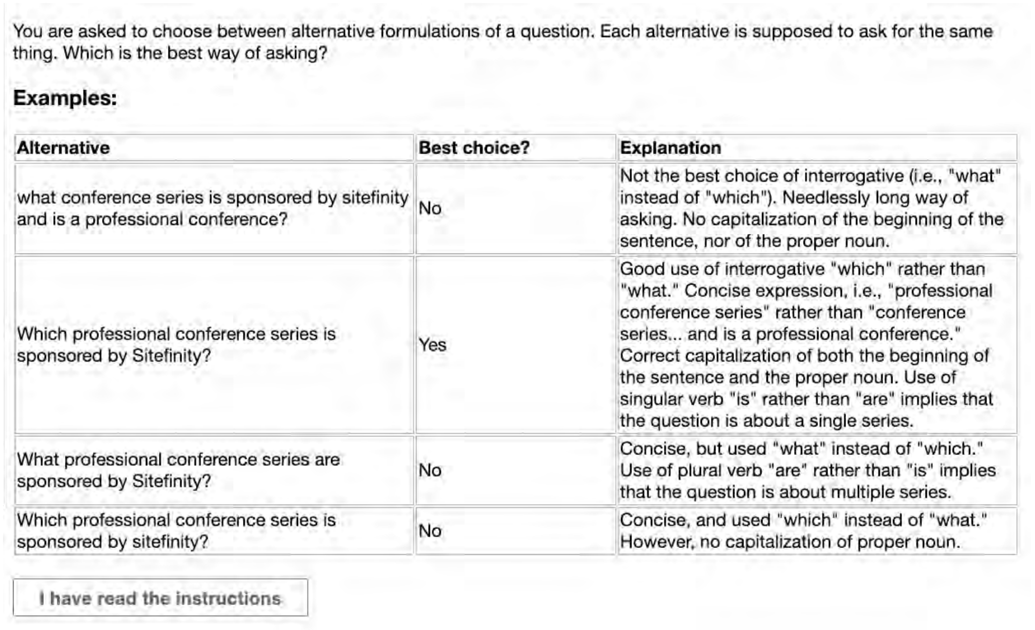
Overall, the consistency of ratings across the five Likert scales over the 750 sampled NL questions as rated in the approved crowdsourced responses is calculated as Cronbach's α = 0.707, which is designated as "acceptable."

5.3.3 Task 2: Validate and Vote

We then use the rewritten questions from the previous task to establish which version of an NL question is the better formulation. For every original question where at least one rewrite was provided by crowd workers in Task 1, we take the original question and up to three rewrites, shuffle the order, and ask a different set of crowd workers to choose which version of the question is the best way of asking. Both the instruction and examples are illustrated in Fig. 5.8, while Fig. 5.9 shows the simple form for this task.

For this task, since the response type is very simple and if less than three rewritten questions were generated there is always at least one non-option which the crowd worker technically can choose, quality control consists of removing responses from crowd workers who repeatedly choose non-options.

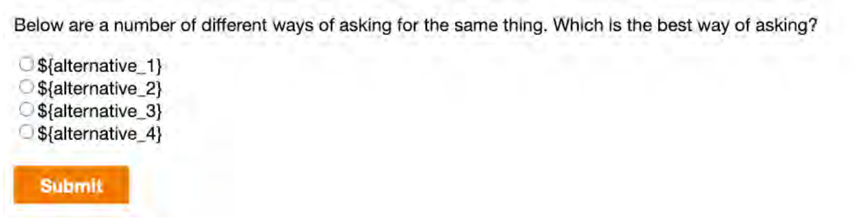
Each question and its rewrites are validated by at least three crowd workers. If the Task 2 result is a clear majority for any specific version of the question, then that is the question carried forward into the rewritten questions test collection. If there is not a clear majority given a set of original question and its rewrites, two more crowd worker validations are requested until a majority vote emerges. The distribution of responses is displayed in Fig. 5.10, while the number of crowd worker responses required to reach a majority is shown in Fig. 5.11. In total, 541 of the 750 questions are rewritten in the new collection; see Table 5.3 for a breakdown on specific subsets.


You are asked to choose between alternative formulations of a question. Each alternative is supposed to ask for the same thing. Which is the best way of asking?

Examples:

Alternative	Best choice?	Explanation
what conference series is sponsored by sitefinity and is a professional conference?	No	Not the best choice of interrogative (i.e., "what" instead of "which"). Needlessly long way of asking. No capitalization of the beginning of the sentence, nor of the proper noun.
Which professional conference series is sponsored by Sitefinity?	Yes	Good use of interrogative "which" rather than "what." Concise expression, i.e., "professional conference series" rather than "conference series... and is a professional conference." Correct capitalization of both the beginning of the sentence and the proper noun. Use of singular verb "is" rather than "are" implies that the question is about a single series.
What professional conference series are sponsored by Sitefinity?	No	Concise, but used "what" instead of "which." Use of plural verb "are" rather than "is" implies that the question is about multiple series.
Which professional conference series is sponsored by sitefinity?	No	Concise, and used "which" instead of "what." However, no capitalization of proper noun.

Figure 5.8: Instructions and examples presented to crowd workers performing data annotation Task 2.


Below are a number of different ways of asking for the same thing. Which is the best way of asking?

- ${alternative_1}
- ${alternative_2}
- ${alternative_3}
- ${alternative_4}

Figure 5.9: The form presented to crowd workers performing data annotation Task 2.

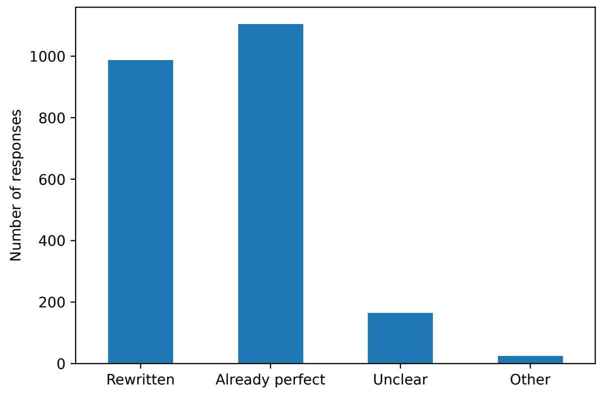


Figure 5.10: Histogram of frequencies over whether a rewritten question was provided (“rewritten”) or otherwise reasons given for not rewriting (the original question was “already perfect” or “unclear,” or else some “other” reason).

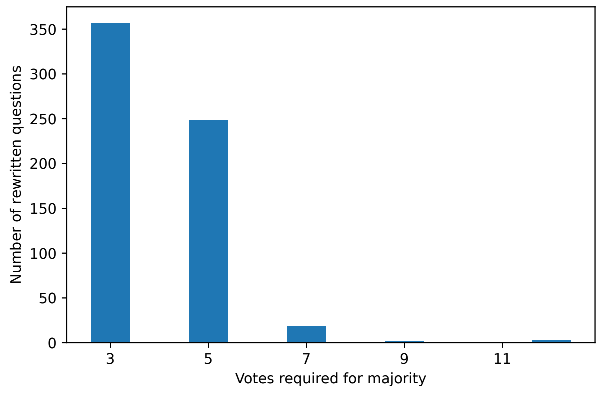


Figure 5.11: Histogram of frequencies over how many votes were required to determine a majority in favour of one version of the original or rewritten question.

5.4 Experiments

We compare model performance on the original versus rewritten NL questions in our samples. Specifically, we use neural KGQA models trained on the DBNQA* [78] and GrailQA [53] datasets.[1] The question we seek to answer is how quality improvements on the input NL questions impact the answer prediction effectiveness

[1]Since we could not find papers with open source code addressing LC-QuAD v2.0 and there are still no models on the corresponding leaderboard, this dataset is not included in our experiments.

of the models (RQ 3.3).

5.4.1 Experimental Setup

For each of the KGQA datasets, the underlying KG is provided by a Virtuoso triple-store instance. For DBNQA*, DBpedia 2016 is the KG used to execute formal queries to retrieve answers. For GrailQA, Freebase is served as the KG, following the instructions provided by Gu et al. [53].[2] This includes using their processed version of Freebase to make it fully compatible with the relevant Resource Description Framework (RDF) standard.

The GrailQA models rely on entity linking, which is provided for the full GrailQA validation split. In order to compare the original and rewritten question samples under equivalent conditions, the rewritten questions are identified with the original questions' query ID to apply the same entity linking to the rewritten NL question.

5.4.2 Methods

We use seven different neural KGQA methods to test the effect of rewritten NL questions on KGQA performance. These are sequence-to-sequence neural models with an encoder-decoder motif, where all but one are used to generate the formal query as a sequence of tokens. The exception is Ranking+BERT [53], where a neural model is used as a ranker to rank generated candidate formal queries.

Three methods are variations of the Neural Sparql Machine (NSpM) [124, 125, 151] architecture, including the NSpM, NSpM+Att1, and NSpM+Att2 models. They are all based on Tensorflow NMT. NSpM+Att1 features a normed Bahdanau [14] attention mechanism, while NSpM+Att2 uses a scaled Luong [83] attention mechanism. All three NSpM models are specified with 2 layers and a dropout coefficient of 0.2. They are also all trained for 50,000 training steps.

Two methods, ConvS2S [50] and Transformer [140], are adapted from machine translation between natural languages to semantic parsing for KGQA. We rely on the sequence-to-sequence model implementations in Pytorch.[3] To better support the models, the both NL and formal query data is pre-processed with

[2]https://github.com/dki-lab/Freebase-Setup
[3]https://github.com/bentrevett/pytorch-seq2seq/

sub-word tokenization, specifically Byte Pair Encoding (BPE) [116] using Sentencepiece.[4] The other hyperparameters for ConvS2S and Transformer are kept as default, except notably the training data is not shuffled between epochs during training, and the models are trained in a case-sensitive manner.

The five models mentioned thus far are all trained on the training split of DBNQA*, which has been shuffled. The models predict formal queries from the NL questions in the full test split of DBNQA*, as well as the original sample of 250 NL questions from the test split, and the rewritten NL questions of the same sample.

Next, we use the methods using a pre-trained model based on BERT [35] provided by Gu et al. [53].[5] Specifically, the model is an LSTM-based sequence-to-sequence which uses uncased base-BERT for encoding, and is fine-tuned on GrailQA train split. The Transduction+BERT method uses this model for generating a formal query in an auto-regressive manner. In contrast, Ranking+BERT uses this model to rank candidate formal queries.

5.5 Results and Analysis

With the methods described above, we achieve the KGQA performance results listed in Table 5.4. We use two performance measures, Exact Match (EM) and F_1, to quantify effectiveness. Exact match compares the predicted formal query to the ground truth formal query. For DBNQA*, EM is 1.0 for an instance if and only if the two strings are identical. Meanwhile, using the provided evaluation script with GrailQA [53], the predicted and ground truth formal queries are both converted to query graphs and are considered as exactly matching if the graphs are isomorphic. The F_1 measure is based on the precision and recall of comparing answer sets. For the KGQA models evaluated on DBNQA*, if both the ground truth answer and the predicted answer are empty sets, the score for an instance is 1.0. This follows the example of Usbeck et al. [137].

In Table 5.4, we observe that the original questions in our sample (IQN-KGQA) may differ in terms of mean performance when compared to the full subset from which the sample was taken. The difference can be either either lower (e.g., Transformer) or higher (e.g., ConvS2S) on the sample than on the full subset. However,

[4] https://github.com/google/sentencepiece
[5] https://github.com/dki-lab/GrailQA/

Table 5.4: Results on the DBNQA* (testing split) and GrailQA (validation split) datasets. The full subset refers to the original benchmarks and is included for reference. The IQN-KGQA dataset contains a sample of 250 questions per dataset. Performance is reported on the original questions in the sample as well as on their rewritten variant with improved naturalness. Significance is tested between the rewritten and original questions.

Dataset	**Method**	**Full subset**		**IQN-KGQA**			
		Original questions		**Original questions**		**Rewritten questions**	
		EM	**F_1**	**EM**	**F_1**	**EM**	**F_1**
DBNQA*	NSpM	0.000	0.013	0.000	0.020	0.004	0.016
	NSpM+Att1	0.081	0.119	0.085	0.105	0.033†	0.050†
	NSpM+Att2	0.089	0.132	0.081	0.117	0.028†	0.050‡
	ConvS2S	0.091	0.138	0.121	0.152	**0.036**‡	0.048‡
	Transformer	**0.177**	**0.260**	**0.166**	**0.254**	**0.036**‡	**0.067**‡
GrailQA	Ranking+BERT	**0.510**	**0.583**	**0.452**	**0.540**	**0.372**	**0.452**†
	Transduction+BERT	0.337	0.364	0.296	0.339	0.208†	0.251†

performance on original questions are of the same magnitude for both the full subset and sample across all methods. This holds true for both DBNQA*-test and GrailQA-dev.

In contrast, performance is reduced drastically when predicting on the rewritten questions. The three methods with the highest performance overall, Transformer, Ranking+BERT, and Transduction+BERT, all show that performance in both EM and F_1 is reduced by a large fraction (up to 78%) when predicting on the rewritten questions compared to predicting on the original questions. This trend is also followed by the NSpM+Att2 results, while the remaining models, NSpM, NSpM+Att1, and ConvS2S all show some deviations. These models achieve a higher performance in one or both measures on the sampled original questions compared to the full DBNQA*-test split. Excepting the NSpM, however, performance in both measures is less when predicting on the rewritten questions than the original questions.

We indicate statistical significance in Table 5.4 on the performance of the rewritten questions sample compared to the original questions sample for each KGQA model. A single dagger (†) indicates that the p-value was below $\alpha = 0.05$, while a double dagger (‡) indicates that the p-value was less than the Bonferroni-corrected threshold of $\frac{\alpha}{7}$ based on the seven comparisons made for each dependent variable.

5.6 Discussion

This chapter addresses data quality and collects improved formulations of NL questions, to yield the IQN-KGQA test collection. We reflect on the data collection process, discuss possible uses of our test collection, and identify limitations.

5.6.1 Data Collection

We have followed a similar procedure as the crowdsourced paraphrasing and cross-validation used in the construction of several large-scale KGQA datasets described in Sect. 2.5. Unlike the reported crowdsourcing of prior datasets, we have involved the crowd workers in a consideration of language quality and question naturalness, by soliciting ratings on the five unnaturalness dimensions, immediately prior to paraphrasing an original NL question. The Likert ratings themselves provide a perspective into how crowd workers see NL questions that should be rewritten compared to those that should not or cannot be rewritten.

A majority of sampled NL questions were marked by some of the crowd workers as needing rewriting. Furthermore, during the second crowdsourced task, we see that the for most rewritten questions, the preferred version emerges quickly—in most cases with 3–5 votes. The resulting test collection has a majority (541 of 750) of its NL questions rewritten from their original form. This indicates that crowd workers have found room for improvement even after the initial paraphrasing and cross-validation undertaken in the original KGQA datasets' construction. This proportion of question rewrites also indicates that all three KGQA datasets can benefit in terms of question naturalness from extensive NL question rewriting.

5.6.2 Utilization

This chapter describes a process of improving NL questions for KGQA datasets. This shows the value of additional rounds of rewriting and quality control when creating NL questions via crowdsourcing. However, the reduced performance of KGQA models on the sample with rewritten NL questions also calls into question the overall approach of relying heavily on crowdsourcing for large scale KGQA dataset construction.

We encourage other researchers to report their performance on our IQN-KGQA test collection as well as the test splits of the KGQA datasets on which they train their models. This will serve to keep the true KGQA performance in perspective. The reduced performance caused by rewritten NL questions illustrates that KGQA models are effectively overfitting on their datasets and do not generalize to natural question formulations. Our IQN-KGQA collection can be used to guard against this.

Our crowdsourcing designs can be utilized in future large-scale KGQA dataset construction efforts. The numerical ratings on the various dimensions of unnaturalness (in Task 1) may be used as quality control. Our collection could also be utilized for automatic question rewriting using, e.g., for fine-tuning large language models, to generate question paraphrases to contribute to the pool of options that crowd workers can vote on (as in Task 2).

5.6.3 Limitations

We tried to simplify quality control of crowdsourcing by having some heuristics about what constituted a reasonable effort of rewrites, but these filters were perhaps not sufficient. There are examples where the crowdsourced rewriting and validation seem to fail to improve the NL question that is rewritten. For example, the original question "Which past members of the labelle also sang somebody loves you baby (Blackstreet & Ma song)?" was voted down in favor of the rewritten question "What song is Patti LaBelle famous for ?"

Our test collection is of small scale, yet has been relatively expensive to produce, on the order of US$1000. Although crowdsourcing labor may be an economic way to scale up data annotation, there remains a question of how involved the manual quality control should be from the researchers' side.

5.7 Summary

We have investigated the dimensions of unnaturalness in the nominally natural language questions found in several modern large-scale knowledge graph question answering (KGQA) datasets. Specifically, we have developed a coding scheme to evaluate the naturalness of NL questions. We have also used crowdsourcing to rewrite such NL questions in KGQA datasets to be more genuinely natural. By combining language quality evaluation with NL question rewriting, we have attempted to prime crowd workers with attention towards language quality. From these rewritten NL questions, we have created the IQN-KGQA test collection with grounding in each of the three major knowledge graphs (KGs) addressed in previous KGQA research: DBpedia, Freebase, and Wikidata. This test collection can put KGQA performance in a more realistic perspective compared to testing KGQA systems on validation and test splits created with the exact same procedure as the training split. We have experimentally shown the impact of our test collection on the performance of KGQA models compared to performance on the corresponding sample of original NL questions and found that model performance deteriorated substantially when a more natural formulation of the same questions was provided. This suggests that existing models do not generalize well to genuinely natural questions. This chapter represents an initial effort to better understand ways to improve the naturalness of NL questions for KGQA and to ensure that KGQA performance is evaluated with genuinely natural questions.

Chapter 6

Grounded Evaluation Measures

This chapter focuses on the SP-KGQA task, which consists of mapping a natural language (NL) question to a formal query that is machine executable, such as SPARQL. The SP-KGQA task is currently evaluated by adopting evaluation measures from other tasks, such as information retrieval and machine translation. However, this adoption typically occurs without fully considering the desired behavior of SP-KGQA systems. This leads to our main research question about what constitutes appropriate SP-KGQA evaluation measures (RQ 4), which boils down to a series of subquestions: the shortcomings of established measures (RQ 4.1), the axioms that SP-KGQA evaluation measures should satisfy (RQ 4.2), and the formally grounded evaluation measures that might be constructed to satisfy those axioms (RQ 4.3). To address these research questions, in this chapter we articulate task-specific desiderata, then develop novel SP-KGQA measures based on a probabilistic framework. We use the desiderata to formulate a set of axioms for SP-KGQA measures and conduct an axiomatic analysis that reveals insufficiencies of established measures previously used to report SP-KGQA performance. We also perform experimental evaluations, using synthetic and state-of-the-art neural machine translation approaches. The results highlight the importance of grounded alternative SP-KGQA measures.

6.1 Motivation

The reported evaluation measures in KGQA research are often repurposed from other tasks, like machine translation and ad hoc retrieval [29, 72], as detailed in Sect. 2.6.1. When the interpretability of predictions is of interest, i.e., SP-KGQA, it is not clear that straightforward adoption of measures established for other tasks is appropriate. In fact, SP-KGQA evaluation has been an undeservedly neglected field of research.

In order to capture the performance quality of SP-KGQA systems there is a need for a theoretically grounded analysis which is currently missing in the field. Axiomatic analysis has been a productive methodology to investigate and develop evaluation measures [3, 5, 6, 8, 9, 25, 45–48, 113, 114], as detailed in Sect. 2.6.2. In the present work, we therefore make an initial effort to apply the axiomatic approach to developing formally grounded evaluation measures for SP-KGQA.

Research in SP-KGQA is increasingly oriented towards neural machine translation (NMT) architectures [42, 125, 151], and the reported results are promising. We therefore further limit our scope to focus on state-of-the-art NMT methods in our experimental evaluation, noting that the proposed measures are nevertheless applicable to all SP-KGQA systems. The scope of NMT systems is chosen as a tractable and sufficient experimental scope where state-of-the-art performance from previous work with open-source codebases can be reproduced on a complex KGQA dataset.

We begin by describing the desiderata of the SP-KGQA task. Using these desiderata, we derive a probabilistic framework for novel evaluation measures that combines desideratum-specific *component* measures into *compound* measures that by construction address all task desiderata. We also describe a number of specific component measures, and specific instantiations of the framework, i.e., novel compound measures.

We next postulate axioms for SP-KGQA measures on the basis of the task desiderata. With this theoretical basis, we perform an axiomatic analysis on established measures, as well as novel proposed measures. This analysis reveals that all the established measures used to evaluate SP-KGQA in previous work have critical shortcomings with respect to properly evaluating this task, as established measures can only partially satisfy the axioms.

To validate the framework itself and the novel measures, we perform an experimental evaluation using both synthetic (ground truth degraded in a controlled manner) and state-of-the-art NMT SP-KGQA models. We find that important differences between NMT architectures for SP-KGQA are obscured by evaluating with individual established measures. Our proposed measures provide the necessary instrumentation to conduct a balanced re-evaluation.

6.2 Measurement Framework

We begin by defining the SP-KGQA task and associated desiderata in Sect. 6.2.1. On this basis, we derive a probabilistic framework to express an ideal SP-KGQA measure in Sect. 6.2.2. We next develop component measures in Sect. 6.2.3 and present some possible compound measures in Sect. 6.2.4.

6.2.1 Desiderata of the Semantic Parsing KGQA Task

The task of SP-KGQA is, given some natural language (NL) question q, to produce a corresponding formal query f that when executed on knowledge graph $\mathcal{K}$ will return the correct answer a to the question. We denote the result of this query execution as $r_{\mathcal{K}}(f) = a$. If the formal query is syntactically incorrect, attempted execution returns an error, $r_{\mathcal{K}}(f) = \epsilon$. Conversely, if the formal query is well-formed but returns no entities or values, we denote this as $r_{\mathcal{K}}(f) = \emptyset$. In the present work we only address NL questions that can be correctly represented as a KG formal query, e.g., factual questions with semantic support in the KG. In the context of a specific $\mathcal{K}$ then, for an NL question q there exists a corresponding ground truth formal query f^* that represents the NL question in a structured manner (i.e., as a logical form), which when executed returns the ground truth answer, $r_{\mathcal{K}}(f^*) = a^*$.

According to Chakraborty et al. [29], a correct formal query f produced by SP-KGQA must both correctly represent the meaning of q, and return the correct answer a corresponding to q when f is executed on the KG. We write $f \overset{\text{sem}}{\equiv} f^*$ to denote semantic equivalence between a predicted formal query f and the ground truth formal query f^* (acknowledging that f may be semantically equivalent to f^* without being verbatim identical). As a simple example, a fact `x parentOf y` is semantically equivalent to the fact `y childOf x`. Another example of semantic equivalence is that the ordering of triple patterns in the `WHERE`-clause of a SPARQL query can be reordered without changing the meaning or effect of the formal query:

```
SELECT ?person WHERE { ?person childOf ?parent
                     . ?parent childOf Albert_Einstein }
```

is semantically equivalent to

```
SELECT ?person WHERE { ?parent childOf Albert_Einstein
                     . ?person childOf ?parent }
```

and both these formal queries represent the NL question “Who are the grandchildren of Albert Einstein?”

Note that retrieving the correct answer requires the formal query to be syntactically correct. We therefore also make the *executability* requirement explicit in postulating the following desiderata:

D1 *Semantic representation* (or semantic structure in [29]): f correctly represents the meaning of q. Formally: $f \overset{\text{sem}}{\equiv} f^*$.

D2 *Syntax correctness:* f is well-formed under the formal query language and does not return an error ϵ when executed on $\mathcal{K}$. Formally: $r_{\mathcal{K}}(f) \neq \epsilon$.

D3 *Answer correctness:* f when executed on $\mathcal{K}$ retrieves the correct answer a. Formally: $r_{\mathcal{K}}(f) = r_{\mathcal{K}}(f^*) = a^*$.

An imperfect system could satisfy some but not all these desiderata. For example, if the NL question is “What country has the highest GDP in the world?”, then a good system would retrieve the answer “USA.” If a system retrieves the same answer by a formal query that represents the meaning of a different NL question, such as “What country has a flag known as the Stars and Stripes?”, this would be better than getting the wrong answer, but not as good as getting the correct answer with the correct formal query.

We want correct answers, but a reliable and trustworthy system must get the correct answers for the correct reasons. Thus, syntax correctness reflects whether the formal query is well-formed, and hence executable, while semantic representation reflects how well the meaning of the NL question is reflected in the formal query.

As a more explicit example from our dataset and experiments, we can compare a ground truth formal query with a spurious predicted formal query that illustrates the same point: The NL question is "Did justin madden study at the australian catholic university university? [*sic*]" and the ground truth formal query is

```
ASK WHERE { dbr:Justin_Madden dbp:university
                               dbr:Australian_Catholic_University }
```

but the model predicts the formal query

```
ASK WHERE { dbr:Justin_Madden dbo:almaMater ?uri . }
```

which represents a different NL question, "Did Justin Madden have an alma mater?" However, the answers to the ground truth and predicted formal queries are identical: `True`. This illustrates how measuring task performance only with respect to one desideratum could give a misleading impression, either overestimating the quality of predictions or neglecting the merits of imperfect performance.

6.2.2 Probabilistic Framework

Given that the SP-KGQA task has several desiderata which might not all be fulfilled simultaneously, an evaluation measure for SP-KGQA should ideally take each of the three desiderata into account. Furthermore, we would like to be able to quantify partial success with respect to each desideratum. Therefore, the measure of each desideratum may be expressed as a probability of how plausible [64] the prediction is, given the evidence. Here, the prediction comprises both the predicted formal query and answer, while the evidence referenced is the ground truth formal query and answer. We denote the overall correctness of the SP-KGQA prediction as the probabilistic expression $P(C = 1|q, f, f^*)$.

Here C is a binary random variable denoting correctness with respect to all desiderata, $C = D_1 \wedge D_2 \wedge D_3$, where D_i is the binary random variable corresponding to desideratum Di. For binary random variable $X \in \{0, 1\}$, we simplify the probability notation as $P(X = 1) = P(X)$. We then express $P(C|q, f, f^*)$ as:

$$\begin{aligned} P(C|q,f,f^*) &= P(D_1 \wedge D_2 \wedge D_3 \,|\, q, f, f^*) && (6.1)\\ &= P(D_1|q,f,f^*)\, P(D_2 \wedge D_3|q,f,f^*) && (6.2)\\ &= P(D_1|q,f,f^*)\, P(D_2|q,f,f^*)\, P(D_3|D_2,q,f,f^*) && (6.3)\\ &= P(D_1|f,f^*)\, P\big(D_2|r_{\mathcal{K}}(f)\big)\, P\big(D_3|D_2, r_{\mathcal{K}}(f), r_{\mathcal{K}}(f^*)\big) && (6.4) \end{aligned}$$

Starting from Eq. (6.1) we make the assumption that D1 is independent both from D2 and from D3, and get Eq. (6.2). This assumption is made because predicted formal queries can represent relevant semantic and structural aspects of the NL question, even if other desiderata are not well satisfied. As stated, if the predicted formal query is not executable, there can be no answers, much less correct answers. Therefore, D3 is conditionally dependent on D2. In contrast, it is perfectly possible for a predicted formal query to have correct syntax without returning the ground truth answer. Therefore D2 is conditionally independent from D3. This yields Eq. (6.3) from Eq. (6.2). Finally, in Eq. (6.4) we express each *component* in terms of the input variables immediately relevant to them.

Equation (6.4) presents a general framework, based on which one can instantiate specific *compound* measures, by setting the different components. Note that the instantiated compound measure does not need to be strictly probabilistic; since component measures are multiplied, any real-valued measure in a fixed range may be used. The framework is probabilistic to make explicit the dependencies between the desiderata. Nevertheless, the component measures do not need to be probabilistic—this allows us to instantiate specific measures based on the framework using existing measures as component measures (as we do with BLEU in Sect. 6.2.4), with the overall result being rank-equivalent.

In addition, the framework can be relaxed to ignore a desideratum by substituting the value 1 for the respective component. Also note that because of the multiplicative formulation, the compound expression in Eq. (6.4) yields zero if any of the components are zero. In order to preserve the evaluation of partial successes on each component, we need to make sure that $P(D_i|.) > 0$ for $i \in 1, 2, 3$. This could be ensured by enforcing a minimum non-zero value γ for each of the components: $P(D_i|.) = \gamma + (1 - \gamma)\hat{P}(D_i|.)$. Finally, this formulation assumes that each of the desiderata are weighted equally. However, the framework could be extended to allow for non-uniform weighting; this is left to future work.

6.2.3 Component Measures

Next, we consider in more detail each component and discuss suitable measures, using either established measures, where appropriate, or developing new ones.

Measuring Semantic Representation: We express D1 as $f \overset{\text{sem}}{\equiv} f^*$ to emphasize that semantic equivalence is more nuanced than just an Exact Match [42, 104, 151]. From machine translation, n-gram-based measures like BLEU [97] or ROUGE [75] may be used to measure semantic representation, capturing both partial and complete success of a predicted formal query. However, such n-gram-based approaches do not distinguish between terms; specifically, they do not recognize the key semantic elements in the formal query, like entities and predicates. We therefore define novel component measures to quantify semantic representation, $P(D_1|q, f, f^*)$, that can reflect both partial and complete success in terms of semantic elements in the formal query. First, considering the formal queries as sets of individual semantic elements, i.e., entities and predicates:
$f_{\text{Sem}} = \{e \in f\} \cup \{p \in f\}$.

This gives us recall and precision on the level of individual semantic elements,

$$R_{\text{Sem}}(f, f^*) = \frac{|f_{\text{Sem}} \cap f^*_{\text{Sem}}|}{|f^*_{\text{Sem}}|}; \quad P_{\text{Sem}}(f, f^*) = \frac{|f_{\text{Sem}} \cap f^*_{\text{Sem}}|}{|f_{\text{Sem}}|} \tag{6.5}$$

and consequently semantic representation F-measure,

$$F_{\beta,\,\text{Sem}}(f, f^*) = \frac{1 + \beta^2 R_{\text{Sem}}(f, f^*) P_{\text{Sem}}(f, f^*)}{R_{\text{Sem}}(f, f^*) + \beta^2 P_{\text{Sem}}(f, f^*)}, \tag{6.6}$$

where we simply take $\beta = 1$ to have $\text{F}_{1,\,\text{Sem}}(f, f^*)$.

Second, following the same rationale as above, but considering formal queries as sets of (s, p, o) triple patterns: $f_{\text{Tri}} = \{(s, p, o) \in f\}$, we define the triple patterns-based semantic representation F-measure where we simply take $\beta = 1$ to get $\text{F}_{1,\text{Tri}}(f, f^*)$, following the same steps as Eq. (6.6). Following Usbeck et al. [137], for each set-based measure, if both the ground truth answer and the predicted answer are empty sets, the score for an instance is 1.0. Note that we here take advantage of the permutation invariance of the triple patterns as illustrated in Sect. 6.2.1. For both $\mathrm{F}_{1,\,\mathrm{Sem}}(f, f^*)$ and $\mathrm{F}_{1,\mathrm{Tri}}(f, f^*)$, the rationale is to focus on the distinct meaningful parts of the formal query, i.e., the URIs, while disregarding the syntactical elements and triple pattern ordering. In the case of $\mathrm{F}_{1,\mathrm{Tri}}(f, f^*)$, the evaluation is simplified by ignoring the placeholder variables in the triple patterns, which would otherwise require more involved coordination reflecting the graph structure of the formal query. Extending $\mathrm{F}_{1,\mathrm{Tri}}(f, f^*)$ in this regard may be warranted in future work.

Measuring Syntax Correctness: Simply and strictly, syntax correctness of the predicted formal query can be evaluated by execution. We distinguish this specific measure *Executability* (Exec), from D2 (*syntax correctness*) as it may be possible to measure degrees of syntax correctness. Hence, we express D2 simply as $r_{\mathcal{K}}(f) \neq \epsilon$. We then have $\mathrm{Exec}(f) = 1$ iff $r_{\mathcal{K}}(f) \neq \epsilon$, otherwise $\mathrm{Exec}(f) = 0$. A continuous measure of syntax correctness with values between 0.0 and 1.0 could be implemented in various ways, but this is left as future work.

Measuring Answer Correctness: For answer correctness, Exact Match is an applicable established measure. However, to capture partial success, we can consider the retrieved answers as sets of result tuples, $\mathcal{T}_a = \{\tau_a \in a\}$. We can then obtain an answer F-measure, following the same steps as Eq. (6.6), to obtain $\mathrm{F}_{1,\mathrm{Ans}}$. As D3 depends on D2, if $r_{\mathcal{K}}(f) = \epsilon$ we simply set $P(D_2|.) = P(D_3|.) = \gamma$.

There is a variety of types of answers that can be retrieved by formal queries from $\mathcal{K}$. For a given formal query (considering formal queries in SPARQL, specifically), the answer type may be, for example, a boolean, an entity or predicate URI, a literal value, a tuple, or a set of tuples. Generally, we treat all answers as sets of tuples, even if they contain a single item. That way we can use set overlap-based measures for answer correctness.

6.2.4 Novel Compound Measures

Using the probabilistic evaluation framework introduced in Sect. 6.2, we now instantiate three novel compound measures GEK-1..3, where GEK is an acronym for "Grounded Evaluation of SP-KGQA." Specifically, we vary the semantic representation component, but measure syntax correctness and answer correctness in a fixed way, yielding the following novel compound measures:

$$\text{GEK-1} = \text{BLEU} \cdot \text{Exec} \cdot \text{F}_{1,\text{Ans}}$$

$$\text{GEK-2} = \text{F}_{1,\text{Sem}} \cdot \text{Exec} \cdot \text{F}_{1,\text{Ans}}$$

$$\text{GEK-3} = \text{F}_{1,\text{Tri}} \cdot \text{Exec} \cdot \text{F}_{1,\text{Ans}}$$

While other combinations would also be possible, we have selected a tractable number of compound measures that are expected to address the shortcomings of established measures.

6.3 Axiomatic Analysis

We have proposed a framework for SP-KGQA measures based on task desiderata in Sect. 6.2.1. In light of these, we develop axioms that formally express the constraints that SP-KGQA measures should satisfy. We then analyze established and proposed measures in terms of these axioms.

6.3.1 Axioms

The SP-KGQA task consists of instances with elements (q, f, f^*), where for a given KG (f, f^*) determine the answers (a, a^*). For each axiom, we consider an abstract measure of the form $m(f, f^*)$, the evaluation of a predicted formal query f with respect to the ground truth f^*. The axioms discuss comparisons of the form $m(f_1, f^*) \geq m(f_2, f^*)$ when comparing the evaluation properties of two hypothetical predictions f_1 and f_2. Note that answer-level measures are analogously expressed in the form $m(a, a^*)$.

Our first axiom (A1) corresponds to desideratum D1. Practically, we must evaluate the logical equivalence of f to q by comparing f to f^*.

***Axiom A1 - Semantic Representation**:*

A formal query f may be considered as a set $\mathcal{U}_f$ of elements u_f that are semantic properties extracted from the formal query f, such as entities in f, predicates in f, (s, p, o) triple patterns in f, or formal query language keywords used in f. If we have the following:

a. A set comparison function $g(f, f^*) \in [0, 1]$ that compares f and f^* as sets of elements $u_f \in \mathcal{U}_f$ and $u_{f^*} \in \mathcal{U}_{f^*}$, such that a correctly predicted formal query $f \overset{\text{sem}}{\equiv} f^* \implies g(f, f^*) = 1$.

b. Two predicted formal queries f_1 and f_2 where f_1 is a better prediction than f_2, i.e., where $1 > g(f_1, f^*) > g(f_2, f^*) \geq 0$.

Then we must also have that $m(f_1, f^*) > m(f_2, f^*)$.

Our second axiom (A2) corresponds to D2, concerning syntax correctness.

***Axiom A2 - Executability**:*
A predicted formal query must be syntactically well-formed and executable to be correct. If we have the following:

a. A ground truth formal query and its corresponding answer: $r_{\mathcal{K}}(f^*) = a^*$.

b. A predicted formal query f_1 that returns a non-error answer: $r_{\mathcal{K}}(f_1) = a_1 \neq \epsilon$.

c. A predicted formal query f_2 that results in an execution error: $r_{\mathcal{K}}(f_2) = \epsilon$.

Then we must have that $m(f_1, f^*) > m(f_2, f^*)$. This also holds if $a_1 = \emptyset \neq a^*$.

Note that this axiom assumes a strict definition of syntax correctness. If syntax correctness can be measured by degrees, then an additional axiom for D2 may be appropriate.

Our third axiom (A3) is corresponds to D3, concerning answer correctness.

***Axiom A3 - Answer Completeness**:*
Assuming an answer a as a set $\mathcal{T}_a$ of result tuples τ_a retrieved after executing a formal query f. If we have the following:

a. A set comparison function $g(a, a^*) \in [0, 1]$ that compares the predicted and ground truth answers as sets of elements $\tau_a \in \mathcal{T}_a$ and $\tau_{a^*} \in \mathcal{T}_{a^*}$, such that a correctly predicted answer $a = a^* \implies g(a, a^*) = 1$.

b. Two predicted answers a_1 and a_2 where a_1 is a better prediction than a_2, i.e., where $1 > g(a_1, a^*) > g(a_2, a^*) \geq 0$.

Then we must also have that $m(a_1, a^*) > m(a_2, a^*)$.

Table 6.1: Evaluating measures with axioms.

Measure	Axioms A1	Axioms A2	Axioms A3	General properties: Instance level	General properties: Partial reward
Established measures					
Exact Match (Query)	◐[1]	◐	◐	●	○
Exact Match (Answer)	○	●	◐[2]	●	○
Acc/F_1/R/P (Answer)	○	●	●	●	●
Perplexity	○	○	○	●	●
BLEU	◐[1]	○	○	◐	●
ROUGE-L	◐[1]	○	○	●	●
Novel measures					
GEK-1	◐[1]	●	●	●	●
GEK-2	●	●	●	●	●
GEK-3	●	●	●	●	●

[1] Satisfies only A1.a. [2] Satisfies only A3.a.

Both A1.b and A3.b require a specific form of partial reward.

6.3.2 Measure Analysis

Having established axioms derived from task-specific desiderata for evaluation measures, we can now analyze relevant established measures in terms of these axioms. We summarize our findings in Table 6.1, where ● indicates that a measure satisfies an axiom or general property, while ○ indicates it does not, and ◐ indicates a partial addressing of the axiom or property. Specifically, the general properties indicate whether a measure can be evaluated at an *instance level* and whether the measure can give a *partial reward* for a partially successful task.

We see that measures with n-gram-based matching like BLEU and ROUGE-L, as well as Exact Match (applied to the formal query) are able to address A1 partially, satisfying A1.a in that a perfect prediction will indeed give a maximal score of 1, assuming a single valid ground truth formal query for a given NL question. However, these measures do not capture specific semantic properties of the formal query, and so cannot fully satisfy A1. Analogously, Answer Exact Match can only partly satisfy A3, unlike set-based measures like Accuracy and F_1 that can fully satisfy A3. A major shortcoming shared by all the established measures is a failure to explicitly address whether a predicted formal query is executable. Assuming that the ground truth formal query is executable, which should be the case in principle, then Exact Match on the predicted formal query does satisfy A2 and A3. In practice, however, Query Exact Match by itself does not intrinsically guarantee A2 and A3, hence it can only partially satisfy those axioms.

Since D3 depends on D2, any evaluation of answer correctness assumes that the ground truth formal query has been correctly executed, which demonstrates syntax correctness D2. Hence, Answer Exact Match and set-based measures applied only to answer correctness do imply the satisfaction of A2.

Exact Match cannot give partial reward, since only complete success is rewarded. We also note that while BLEU is not created to be evaluated on the instance level, it is possible to apply the measure in an instance-level manner. Finally, we note that Perplexity [23] does not satisfy any of the axioms even partially, and thus is not suitable for evaluating model performance for SP-KGQA. As can be seen from the analysis of measures against axioms, only our novel compound measures GEK-2 and GEK-3 fully satisfy all three axioms.

6.4 Experiments

We investigate the proposed SP-KGQA framework, and more specifically GEK-1..3 and established measures, with two experiments. First, we investigate the sensitivity of measures in Sect. 6.4.2 by evaluating *synthetic runs* with controlled degradation of the ground truth. Second, we evaluate neural models trained on a large complex SP-KGQA dataset in Sect. 6.4.3.

6.4.1 Experimental Setup

We use the DBNQA [57] dataset, which consists of 894,499 instances, generated from 5,217 templates extracted from the LC-QuAD [131] and QALD-7-Train [136] datasets. Specifically, to avoid the reported information leakage issues described in Chap. 4 [78], a sanitized partitioning of DBNQA is used, here referred to as DBNQA* (called Sanitized-1 in Chap. 4 [78]), where the training and test splits are partitioned based on the underlying templates. In terms of the Chakraborty et al. [29] categories for neural KGQA, our work focuses on translation-based SP-KGQA, and our setting is fully supervised machine learning.

Table 6.2: Overview of transformations.

Original query		`SELECT DISTINCT ?uri where { dbr:Villa_Sturegården dbp:locationCountry ?uri }`
Trans.	**Desid.**	**Example degraded formal ground truth query**
T_1	○ ● ●	`SELECT DISTINCT ?uri where { dbr:Villa_Sturegården dbp:locationCountry ?uri`
T_2	● ○ ●	`SELECT DISTINCT ?uri where { dbr:Yorkshire_1 dbp:champion ?uri }`
T_3	● ○ ○	`SELECT DISTINCT ?uri where { dbr:Jonas_Kullhammar dbp:origin ?uri }`

To retrieve answers, all formal queries, including the ground truth, those predicted by models, and those generated for the synthetic runs, are executed against a Virtuoso endpoint holding DBpedia 2016-10 [21] as the KG. The retrieved results are considered the respective ground truth and predicted answers.

During training, prediction, and query evaluation, the formal queries were tokenized and encoded in the manner of Soru et al. [124, 125]. We refer to Yin et al. [151] for an encoding example. For calculating GEK-1..3, we set the value $\gamma = 10^{-4}$.

6.4.2 Experiment 1: Synthetic Experiments

To investigate the sensitivity of SP-KGQA measures, we simulate predictions errors without training models by constructing *synthetic runs* by degrading ground truth test data in a controlled manner. The more degradation applied, the worse the synthetic predictions; consequently, the more evaluation scores should decrease. This demonstrates that our novel compound measures are sensitive and balanced to all of the tested error types, as compared to established measures.

Transformations

We devise a set of *transformations*, that simulate particular types of prediction errors. Each of these transformations preserve the prediction quality with respect to one desideratum while degrading with regards to others. Table 6.2 provides an overview of the transformations, where desiderata marked with ○ are preserved,

Table 6.3: Synthetic experiments. Here † means moderately sensitive response ($\Delta > 0.10 \times x\%$ relative to T_0; ≤ 0.990 for $T_{i,10\%}$, and ≤ 0.980 $T_{i,20\%}$). ‡ means sensitive response ($\Delta > 0.50 \times x\%$ relative to T_0; $\leq$ 0.950 for $T_{i,10\%}$, and $\leq$0.900 for $T_{i,20\%}$).

Transf.	**Established Measures**				**Novel Component Meas.**				**Novel Compound Meas.**		
	Exact Match Query	**Answer**	**BLEU Inst.**	**ROU.**	**Exec**	$\mathbf{F_{1,Ans}}$	$\mathbf{F_{1,Sem}}$	$\mathbf{F_{1,Tri}}$	**GEK-1**	**GEK-2**	**GEK-3**
$\mathbf{T_0}$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$\mathbf{T_{1,10\%}}$	0.899‡	0.899‡	0.990†	0.995	0.899‡	0.899‡	1.000	1.000	0.899‡	0.899‡	0.899‡
$\mathbf{T_{1,20\%}}$	0.799‡	0.799‡	0.981	0.990	0.799‡	0.799‡	1.000	1.000	0.799‡	0.799‡	0.799‡
$\mathbf{T_{2,10\%}}$	0.900‡	0.900 ‡	0.929‡	0.962†	0.961†	0.900‡	0.900‡	0.900‡	0.900‡	0.900‡	0.900‡
$\mathbf{T_{2,20\%}}$	0.799‡	0.800‡	0.859‡	0.925†	0.923†	0.800‡	0.799‡	0.799‡	0.800‡	0.799‡	0.799‡
$\mathbf{T_{3,10\%}}$	0.942‡	1.000	0.965†	0.983	1.000	1.000	0.957†	0.948‡	0.965‡	0.957‡	0.948‡
$\mathbf{T_{3,20\%}}$	0.885‡	1.000	0.931‡	0.966‡	1.000	1.000	0.915‡	0.898‡	0.931†	0.915†	0.898‡

while those with ● are disrupted. Each transformation is randomly applied to 10% or 20% of the test split.

- T_1: Remove the closing curly bracket (`}`) from the `WHERE` clause. This mostly preserves D1 (both n-gram overlaps and semantic properties, like URIs and SPO triple patterns), but disrupts D2, and hence D3. This creates the case of correct semantics but wrong result.
- T_2: Replace each entity (predicate) URI with a random entity (predicate) URI. This mostly preserves D2, but deteriorates D1 and hence D3.
- T_3: Replace the query with another that yields the exact same answer, if another such query exists in the dataset. This preserves D2 and D3, but disrupts D1.

Results

Table 6.3 presents the results of the synthetic experiments, where results meeting the sensitivity thresholds are indicated by daggers.[1] For each transformation T_i and for each measure, we see that established measures, novel component measures, and novel compound measures are reduced either proportionally by the degradation, or else negligibly. For example, T_1 affects Exact Match proportionally at a one-to-one rate, while BLEU and ROUGE-L reduce at a lower rate. This

[1]We omit corpus BLEU because results correlate closely with instance-level BLEU.

illustrates the independence of D1 from D2 and D3. In contrast, $F_{1,\text{Sem}}$ and $F_{1,\text{Tri}}$ are not affected by T_1. Crucially, we see that GEK-1..3 and Query Exact Match are sensitive to all transformations. Out of these measures, GEK-3 is the only measure considered here that satisfies all the axioms and also shows sensitivity to all the synthetic prediction errors tested. Therefore, if a single measure is to be used, we recommend that to be GEK-3.

6.4.3 Experiment 2: Neural Methods

So far, we have shown both theoretically (with our axiomatic analysis in Sect. 6.3.1) and empirically (with our controlled degradation experiments in Sect. 6.4.2) that our novel GEK-1..3 measures indeed capture the desired properties of the SP-KGQA task. Having a principled and validated measurement instrumentation at our disposal, we are interested in applying the novel measures to evaluate state-of-the-art NMT SP-KGQA models, and assess whether the findings agree with the reported results in previous works using established measures.

Methods

- **NSpM [124, 125]:** architectures Baseline, Attention 1, and Attention 2 were based on the original Tensorflow NMT implementation. Hyperparameters were 50 000 training steps, 2 layers, dropout 0.2. Based on [151], Attention 1 used the normed Bahdanau [14] attention mechanism, while Attention 2 used the scaled Luong [83] attention mechanism.
- **ConvS2S [50]:** architecture based on PyTorch implementation,[2] using default hyperparameters, except optimizing with stochastic gradient descent with learning rate of 0.075 and momentum of 0.99.
- **Transformer [140]:** architecture based on the same PyTorch project as ConvS2S, using only default hyperparameters.

Some NMT models perform better with sub-word tokenization, such as Byte Pair Encoding (BPE) [116], so for the Transformer and ConvS2S models we additionally used Sentencepiece[3] BPE restricted to maximum 32k terms. Increased volumes of training data are expected to produce better performance if the model is

[2]https://github.com/bentrevett/pytorch-seq2seq/
[3]https://github.com/google/sentencepiece

Table 6.4: Evaluation of SP-KGQA methods in terms of established and novel measures. Best scores in each block are boldfaced.

Method	**Training**	**Established Measures**					**Novel Compound Meas.**		
	data	**Exact Match**		**BLEU**		**ROUGE**	**GEK-1**	**GEK-2**	**GEK-3**
		Query	**Answer**	**Corpus**	**Instance**				
NSpM	12.5%	**0.000**	**0.059**	**0.437**	**0.374**	**0.679**	**0.009**	**0.006**	0.002
	25%	**0.000**	0.034	0.434	0.371	0.676	**0.009**	**0.006**	**0.003**
	50%	**0.000**	0.019	0.432	0.371	0.678	0.006	0.002	0.001
	100%	**0.000**	0.023	0.432	0.371	**0.679**	0.006	0.004	0.001
NSpM+Att1	12.5%	0.012	0.036	0.486	0.417	0.713	0.022	0.022	0.017
	25%	0.024	0.050	0.484	0.423	0.721	0.037	0.035	0.029
	50%	0.045	0.074	0.511	0.451	0.740	0.060	0.060	0.052
	100%	**0.081**	**0.117**	**0.548**	**0.498**	**0.778**	**0.105**	**0.105**	**0.093**
NSpM+Att2	12.5%	0.008	0.036	0.478	0.408	0.705	0.017	0.016	0.012
	25%	0.029	0.053	0.498	0.436	0.731	0.041	0.041	0.035
	50%	0.049	0.076	0.522	0.460	0.747	0.066	0.066	0.058
	100%	**0.078**	**0.119**	**0.558**	**0.503**	**0.785**	**0.107**	**0.107**	**0.093**
ConvS2S	12.5%	0.042	0.062	0.485	0.444	0.759	0.056	0.056	0.050
	25%	0.066	0.106	0.536	0.490	0.795	0.094	0.100	0.089
	50%	0.084	0.119	0.577	0.519	0.817	0.110	0.112	**0.100**
	100%	**0.085**	**0.126**	**0.582**	**0.525**	**0.821**	**0.115**	**0.114**	**0.100**
Transformer	12.5%	0.051	0.089	0.489	0.436	0.764	0.082	0.088	0.073
	25%	0.077	0.154	0.528	0.480	0.791	0.137	0.147	0.123
	50%	0.102	0.202	0.560	0.510	0.809	0.179	0.193	0.163
	100%	**0.113**	**0.229**	**0.570**	**0.522**	**0.810**	**0.199**	**0.217**	**0.180**

learning. We train models from each of the five architectures on four different training splits from DBNQA*: 12.5%, 25%, 50%, and 100%. We can then compare the measures on models expected to show performance improvement.

Results

Table 6.4 presents the results of SP-KGQA neural models trained on different training data volumes. The NSpM models show negligible change in all measures; this indicates that despite the additional training instances no improvement has occurred. All other methods show improvements in all measures with increased training data. Note that different models improve more on some measures than others. For example, Transformer is not the best model with respect to BLEU or ROUGE-L, but is clearly the best with respect to Query and Answer Exact Match.

As Table 6.4 shows, comparing Exact Match for queries and answers, there can be a large gap between semantic parsing quality and answer correctness. Syntax correctness also shows an SP-KGQA system's mastery of the formal query syntax (e.g., SPARQL). Therefore, our novel compound measures give partial reward to models which generate SPARQL queries which are executable without any errors.

Yin et al. [151] concluded that "the ConvS2S model consistently, significantly outperformed all other models [including Transformer] at a margin," but our results constitute evidence to the contrary. The change between the relative ordering of ConvS2S and Transformer is a crucial one, especially considering the absolute performance difference. While the two are close in terms of BLEU and ROUGE-L (with a slight advantage to ConvS2S), the Transformer model produces substantially more executable queries (see Answer Exact Match scores). Hence, measures of individual components in the SP-KGQA task may give a distorted impression of the system performance on the task as a whole. This shows the advantage of a compound measure being simultaneously sensitive to all desiderata.

6.5 Discussion

Being a first study in this direction, the work presented is not without limitations.

6.5.1 An Initial Axiomatic Effort

We identified the desiderata based on the essential qualities of the KGQA task [29], following a standard methodology of axiomatic development of evaluation measures in Sect. 2.6.2. We acknowledge that our desiderata and axioms represent one possible perspective and it is by no means claimed to be exhaustive. For example, the answer completeness axiom makes strong assumptions about how a retrieved answer is to be treated, i.e., as a set of result tuples. This restricted view of answer correctness does not consider the relationships between the elements of an answer. In the future, additional or alternative axiom-level formulations of answer correctness may be developed, which would regard it as degrees of plausibility, e.g., with respect to the type of the answer retrieved [93]. Further desiderata may be included, e.g., to express a preference for simpler and shorter predicted formal queries. Syntax correctness could also be extended to address degrees of partial correctness. Furthermore, we plan to further expand the set of axioms and component measures to consider structural elements of semantic representation, like reserved words, brackets, and query graph structure.

6.5.2 Choice of Component Measures

In the present work, BLEU and ROUGE are interpreted as reflecting semantic representation because they would partly capture this aspect, at least matching the entity and predicate URI unigrams in a formal query. As for the novel component measures of semantic representation, entities and predicates are treated as more important semantic signifiers than structural elements of the formal queries because they are the explicit KG properties which must be matched. However, the structural elements play an important semantic role, and future work could incorporate these.

Validating whether a formal query provides a good semantic representation of a given NL question would require expert human evaluation efforts. This is similar to the challenge that motivated measures such as BLEU and ROUGE initially. We introduce our overlap-based measures as a first effort at automating the evaluation in terms of salient semantic elements from the formal query. Future work can investigate which semantic representation measures might correlate best with human evaluations of semantic representation. Importantly, the key message of this chapter is not about the specific component measures suggested, but about the framework of taking multiple desiderata of a single task into account simultaneously in a formally grounded manner.

6.5.3 Weighting of Component Measures

The component measures corresponding to the three task desiderata are equally weighted in our proposed compound measure framework because all the desiderata are necessary for the SP-KGQA task. There is no *a priori* difference in importance between the three desiderata. However, as required for a particular system or application scenario, our framework enables different weightings of component measures.

6.5.4 Generalizing to Multiple Knowledge Graphs

We have experimentally addressed KGQA on a single, well-maintained ontology, DBpedia. Since this is an initial iteration on an axiomatic approach to developing grounded evaluation measures for SP-KGQA, we have restricted the scope of the work to general aspects of KGs and formal queries. This has been a helpful constraint to be able to clearly express our framework. However, in future work it may be interesting to develop concepts in our framework with respect to semantic knowledge representation, e.g., as expressed using the OWL Web Ontology Language. Distinct URIs may represent the same underlying property or entity, in which case they are connected by the OWL predicate `owl:sameAs`. If two predicates represent mutually inverse properties, the are connected by `owl:inverseOf`.

Extending the first NL question example in Sect. 6.2.1, with a pair of predicates that are mutual inverses such as `parentOf` and `childOf`, we can replace the triple pattern `?person childOf ?parent` with `?parent parentOf ?person`. We then get yet another formal query formulation

```
SELECT ?person WHERE { ?parent parentOf ?person
                     . ?parent childOf Albert_Einstein }
```

that we can recognize as semantically equivalent to the two formal queries in the Sect. 6.2.1 example, such that all three formal queries correctly represent the meaning of the NL question "Who are the grandchildren of Albert Einstein?" This illustrates how semantic technologies, like OWL, may be used to expand the concept of semantic equivalence. Developments in this direction would also enable our framework to evaluate QA approaches over multiple KGs. The complexity of evaluating SPARQL semantic representations in a multi-KG ontology may be addressed by resolving the synonymy of different URIs for the same entity or predicate.

6.6 Summary

Currently, there is no agreed-upon way of evaluating SP-KGQA systems. Previous work uses multiple measures that evaluate individual aspects, such as either the quality of the semantic parse or the quality of the answer retrieved. It is clear that researchers also want to consider multiple aspects, but there is no other way of doing that other than reporting on a set of different measures. There is no systematic and principled way to combine these aspects in a single unified evaluation measure. Because of that, one particular measure (implicitly or explicitly) becomes “privileged” and gets optimized, at the expense of others. This carries the risk of overfitting and may lead to an imbalanced view of true system performance. It is therefore clear that there is a need to unify SP-KGQA measures in a formally grounded manner.

We have looked through an axiomatic lens at the measures used to evaluate systems’ performance on the SP-KGQA task. We have introduced a probabilistic framework for a family of compound measures capable of addressing all the identified task desiderata. With this framework we have instantiated novel compound measures, designed specifically for the SP-KGQA task. After postulating axioms for SP-KGQA evaluation measures, our axiomatic analysis found insufficiencies in established measures that our novel compound measures resolve. We have also validated the novel measures by evaluating synthetic predictions, before evaluating real predictions by state-of-the-art NMT-based SP-KGQA models. From the experiments, we see that established measures are generally sensitive to some but not all desiderata, unlike our novel compound measures. The discrepancy between established and novel measures we have observed in state-of-the-art NMT-based SP-KGQA models indicates a need for re-examination of the results of previous works.

Chapter 7

Discussion and Conclusions

In the previous four chapters we have presented research done to address the four main research questions and their respective subquestions. In this chapter, we first reflect on the research questions in light of our findings in Sect. 7.1. Next, we consider the limitations of the present work and speculate about possible mitigations in Sect. 7.2. Finally, we discuss possible future work and broader perspectives in Sect. 7.3.

7.1 Reflections on Research Questions

In this section, we discuss our research questions and what we observe from the results presented in Chaps. 3–6.

In Chap. 3, we considered research question **RQ 1**, and its subquestions, regarding training data volume for QA models.

RQ 1 How does training data volume impact QA performance?

RQ 1.1 To what extent do QA models improve when trained on a larger volume of data?

RQ 1.2 How sensitive are the models to fractional changes in training data volume?

With a series of QA models trained on subsets of WikiQA in Chap. 3, we have found that the QA models improved at best modestly when trained on increasingly larger WikiQA subsets. Since the initial dataset was so small, increasing the training data did not make much difference to the trained QA model's performance (**RQ 1.1**), even when the training data was increased by factors of, e.g., 2, 4, or 10 (**RQ 1.2**). In contrast, in Chaps. 4 and 6 with neural KGQA models we have observed a distinct improvement when the training data was increased by factors of 2. While fractionally varying volumes of training data based on a very small training dataset may not affect QA models very much, it is nevertheless clear that increasing volumes of training data even with smaller datasets does improve the performance of the trained QA models (**RQ 1**).

In Chap. 4, we considered research question **RQ 2**, and its subquestions, regarding information leakage between data splits during training and evaluation of KGQA models.

RQ 2 How does it affect model performance or behavior if the test split includes instances that were generated from templates *seen* during training?

RQ 2.1 How is the performance of trained neural KGQA models affected by whether testing templates are *seen* or *unseen*?

RQ 2.2 How is the ability to generalize to instances based on *unseen* templates affected by the volume of training data used?

RQ 2.3 How does the proportion of *unseen* templates to *seen* templates affect the trained models' ability to generalize?

From our experiments with sanitized data partitioning, performance dropped for all methods when test instances were from unseen templates as opposed to seen templates (**RQ 2.1**). However, increasing training data volume improved performance even on the sanitized test split, indicating that additional training data supported generalization to test instances from unseen templates (**RQ 2.2**). Next, different methods showed different patterns of small changes to the experiment with a smaller sanitized test split (**RQ 2.3**). From our experiments in Chap. 4, it is clear that trained KGQA models tested with instances from seen templates display better performance, obscuring the true generalization gap (**RQ 2**).

In Chap. 5, we considered research question **RQ 3**, and its subquestions, regarding the quality of genuine naturalness of NL questions that exist in datasets for KGQA.

RQ 3 How can we ensure or enhance the genuine naturalness of NL questions in KGQA datasets?

RQ 3.1 Can the crowdsourced NL questions in existing KGQA datasets be considered as genuinely natural?

RQ 3.2 What are the properties of a high quality NL question?

RQ 3.3 What happens to the performance of KGQA models when testing against genuinely natural questions?

After crowdsourcing additional paraphrases of NL questions sampled from three different KGQA datasets and voting for the best version, the majority of the NL questions were replaced by improved NL questions. The NL questions that were rewritten were also considered by the crowd workers to be of lower quality in terms of question naturalness than the NL questions which were not rewritten due to being "already perfect" to the crowd workers. This indicates that NL questions in KGQA datasets are lacking in genuine naturalness (**RQ 3.1**). Conversely, by the same data we see that the NL questions were rated as higher quality with respect to the same dimensions of question naturalness, indicating that the dimensions we have identified do reflect the quality of NL questions (**RQ 3.2**). Notably, the dimension of factuality was less directly correlated with NL question quality, which is appropriate sense since the crowd workers were not informed of the NL questions' relationship to KGQA. This shows that a high-quality NL question may still not be appropriate for fact-based QA, such as KGQA. We have also found experimentally that trained KGQA models perform worse on the same sets of sampled NL questions *after* they were rewritten and improved by crowd workers (**RQ 3.3**). On the one hand, this is expected from machine learning models, i.e., testing on instances that are from a distribution less like the training distribution is expected to yield lower performance. On the other hand, we have two observations: (1) many KGQA datasets rely on crowdsourcing to make pseudo-NL questions into genuinely NL questions, and (2) additional crowdsourced paraphrasing and quality control applied to improved the question naturalness results in reduced KGQA performance. This implies either that the common approach to crowdsourcing NL questions is yielding NL questions of inadequate quality, or else that the crowdsourcing approach to NL question paraphrasing and quality control has an inherent limit to the quality that can be achieved (**RQ 3**).

In Chap. 6, we considered research question **RQ 4**, and its subquestions, regarding the appropriateness of evaluation measures used with SP-KGQA models.

RQ 4 What are appropriate evaluation measures for SP-KGQA performance evaluation?

RQ 4.1 What, if any, are the shortcomings of commonly used measures in SP-KGQA evaluation?

RQ 4.2 Can we reason axiomatically about appropriate SP-KGQA evaluation measures?

RQ 4.3 If so, can we construct measures for SP-KGQA evaluation that are formally grounded?

As we have stated, the SP-KGQA approach to KGQA holds the advantage of representing the system's "understanding" of the NL question as a single formal query. However, as the purpose of KGQA is to provide an answer, it is unfortunate that SP-KGQA often only evaluates performance in terms of the predicted formal query (**RQ 4.1**). By performing an axiomatic analysis grounded in the SP-KGQA task desiderata, we have found further shortcomings of commonly used measures (**RQ 4.2**). We have also shown that a formally grounded framework can be used to construct novel evaluation measures for SP-KGQA (**RQ 4.3**) with simultaneous sensitivity to all identified task desiderata, as shown with the synthetic experiments in Sect. 6.4.2. While our axiomatic analysis and novel measures for SP-KGQA represent an initial effort, both the axiomatic analysis and experimental results give evidence for the hypothesized shortcomings of established measures for SP-KGQA. Appropriate measures for SP-KGQA should be developed with an awareness of these shortcomings, and our approach provides strategies to both identify and avoid such shortcomings when constructing novel evaluation measures (**RQ 4**).

7.2 Limitations and Mitigations

Here we acknowledge and address some limitations of our work. Where possible, we speculate about ways of mitigating these limitations.

7.2.1 Set of Axioms

The axioms proposed in Chap. 6 for measures of SP-KGQA were derived from the definition of the SP-KGQA task itself, and based on a deeper consideration of what this task actually entails. We want to make clear that we in no way assert this set of axioms to be complete. Nevertheless, the approach of elucidating task-specific desiderata and axioms has raised important questions about KGQA evaluation. Indeed, this part of our work has illustrated the importance of making explicit assumptions in all aspects of KGQA evaluation.

Furthermore, as the set of axioms of SP-KGQA measures is open to be expanded, it is hoped that in the future SP-KGQA measures will be developed to evaluate partial success in syntax correctness. Such an advance would mean that a corresponding axiom on partial syntax correctness is needed. The challenge lies not so much in formulating such an axiom, but in operationalizing it.

7.2.2 Datasets

In Chaps. 4 and 6, we have used a single KGQA dataset, DBNQA, to train models. This might be a concern in terms of making broad conclusions over KGQA. However, as we have seen in Chap. 5, when we investigate additional KGQA datasets based on different KGs (i.e., URIs are expressed in different namespaces), the quality of naturalness of NL questions show very similar patterns when evaluated by crowd workers. In addition, the KGQA models have shown similar effects on performance when NL questions rewritten for improved naturalness are used to test the models.

7.2.3 Single Language

We have looked at a single natural language, English, and a single formal query language, SPARQL. Regarding the NL questions, there is little reason to expect that a different NL would yield different quality of NL questions. Instead, the quality of NL questions depends on the trade-off made when incorporating crowd-sourcing into the dataset construction workflow, as explored in Chap. 5. As for the formal query language, there is indication from previous work [53] that using S-expressions instead of full SPARQL queries may be easier for models to learn. However, S-expressions are in such cases translated to SPARQL before execution, and we are interested in systems learning the task in an end-to-end manner as much as possible.

7.2.4 Ontology Agnosticity

As previous work [54] acknowledges, a major challenge in SP-KGQA, is the size of the formal query vocabulary due to the many entity and predicate URIs. This is a bigger challenge in SP-KGQA than in similar tasks using a semantic parsing approach, such as text-to-SQL.

Nevertheless, KGs are commonly not only constructed as a set of RDF triples, but obey a set of rules, defined by an ontology, which may be specified, e.g., in OWL. It would certainly be interesting if KGQA can be extended into an ontology-aware form, where the model learns not only a specific RDF namespace, but also learns the governing ontological rules in OWL, and is able to interact with or reason over both. This would potentially mean inferring new facts in the KG based on responding to NL questions, which is important given the challenge of populating a KG with all relevant and current real-world facts.

7.3 Future Work and Perspectives

Finally, we consider future possibilities regarding KGQA specifically, as well as more general topics such as QA, information access, and artificial intelligence (AI).

7.3.1 KGQA

The task of KGQA and the systems created to address this task have evolved in the last decade to address more complex information needs. However, the field is still comprised of diverse positions with regards to its own internal taxonomy and foundational assumptions, such as the space of possible answers. In the end, underlying assumptions will remain individual to the specific KG that a given KGQA dataset addresses or the manner in which a model is applied to the KGQA task.

Conceptually, the distinction between SP-KGQA and IR-KGQA appears initially clear, but in practice these categories say little about the internals of the systems which they encompass. Pragmatically, we can say that unlike IR-KGQA systems, an SP-KGQA system predicts a single formal query, which produces a single predicted answer. However, for IR-KGQA systems that internally rank candidate formal queries or candidate query graphs which may deterministically be translated into candidate formal queries, this categorical distinction is more an issue of whether the top-ranked formal query is recorded and evaluated. As for the IR-KGQA systems which only predict a ranking of candidate answers, the categorical distinction is still intrinsic to the design of the system.

We hope that future work can address where common conventions in the field are carried forward as historical artifacts of previous practical constraints, and where current techniques allow a revision and broadening of fundamental assumptions about what the task entails.

It is possible that some lessons learned from our work on SP-KGQA might also apply to IR-KGQA. For example, with continued reliance on large language models, the issues of training data volume and generalizability will likely constitute challenges for the next generation of IR-KGQA approaches as much as for SP-KGQA. In addition, as we have pointed out in Sect. 2.6, the rank-based evaluation of IR-KGQA is actually disconnected from the end-user task—unlike in other ranking problems where a ranked list of items (documents, entities) is the answer, for KGQA users would really expect a single answer, i.e., the ultimate binary measure would be success@1. To develop measures that can give partial reward, our axiomatic analysis of SP-KGQA evaluation measures may constitute a good starting point for an analogous effort on IR-KGQA evaluation measures.

7.3.2 Question Answering, Information Access, and Artificial Intelligence

QA is often studied in a fact-based QA setting, in which case using a KG as evidential basis is suitable. Intuitively, the QA task where a single NL question yields a single direct and correct answer is one which is very desirable to solve. Nevertheless, we still find ourselves using ad hoc retrieval tools like search engines on the

Web to satisfy our information needs in daily work and leisure. Extending QA beyond the fact-based setting is an important challenge to make QA a primary mode for users to interact with their information systems. Many information needs are not simply factual, and managing multiple subsystems addressing different types of information needs may be a suitable task for an AI system. Indeed, while KGQA is studied largely as an isolated task, one can imagine the KGQA model as a subsystem of a broader dialogue system, e.g., for conversational assistance. Furthermore, it would be useful for the system to reliably inform the user which NL questions, even if factual in nature, are not answerable with a specific KG. The handling of unanswerable questions has so far not been thoroughly explored for the KGQA task.

We have seen rapid development of increasingly large-scale pre-trained language models with applications to text generation, dense retrieval, and dialogue systems, to name but a few. In some cases, such as the LaMDA model [33], which is intended for dialogue applications, the system is trained to use an external information retrieval system as an external "knowledge source." It is a small leap to imagine that KGs can likewise be used as a tool and external knowledge for a dialogue system. Another possible feature would be for a system to support the use of natural language to build, or extend, a KG for future reference. Furthermore, in this way we can hope that users will be able to interact with systems using completely natural language and take advantage of structured data such as KGs for fact-based information needs, alongside other functions of a dialogue system.

In broad terms, what is next for KGQA and QA? These tasks sit at the intersection of information access and artificial intelligence. QA is far from solved. If we compare AI applied to QA with AI applied to playing strategy games, we would consider the current state of progress in QA equivalent to the time period when IBM's Deep Blue beat Garry Kasparov at chess [115]. However, we are not yet at a milestone such as when Google DeepMind's AlphaGo beat Lee Sedol at Go [121, 122]. With the research presented in this thesis, we hope to have made a small but significant move in the direction of helping everyone win at the game of information access.

Appendix A

Appendix: Resources

The work culminating in this thesis includes the development of some resources that are shared with the community for reproducibility and further research.

- **IQN-KGQA:** as part of work on NL question naturalness in the context of KGQA [79], the test collection IQN-KGQA was created. The test collection can be found at `https://github.com/iai-group/IQN-KGQA`.
- **ictir2022-kgqaeval:** as part of work on formally grounded evaluation measures for SP-KGQA [80], the synthetic experiments are shared at `https://github.com/iai-group/ictir2022-kgqaeval`.

Bibliography

[1] Wendy G. Lehnert and Martin H. Ringle (Eds.). 2014. *Strategies for Natural Language Processing*. Routledge. Reprint, originally published 1982 by Lawrence Erlbaum Associates.

[2] Abdalghani Abujabal, Mohamed Yahya, Mirek Riedewald, and Gerhard Weikum. 2017. Automated Template Generation for Question Answering over Knowledge Graphs. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. 1191–1200.

[3] Ameer Albahem, Damiano Spina, Falk Scholer, Alistair Moffat, and Lawrence Cavedon. 2018. Desirable Properties for Diversity and Truncated Effectiveness Metrics. In *Proceedings of the 23rd Australasian Document Computing Symposium (ADCS '18)*.

[4] Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA Corpora Generation with Roundtrip Consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL '19)*. 6168–6173.

[5] Enrique Amigó, Hui Fang, Stefano Mizzaro, and ChengXiang Zhai. 2017. Axiomatic Thinking for Information Retrieval: And Related Tasks. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. 1419–1420.

[6] Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A Comparison of Extrinsic Clustering Evaluation Metrics Based on Formal Constraints. *Information Retrieval Journal* 12, 4 (2009), 461–486.

[7] Enrique Amigó, Julio Gonzalo, Felisa Verdejo, and Damiano Spina. 2019. A Comparison of Filtering Evaluation Metrics Based on Formal Constraints. *Information Retrieval Journal* 22, 6 (2019), 581–619.

[8] Enrique Amigó and Stefano Mizzaro. 2020. On the Nature of Information Access Evaluation Metrics: A Unifying Framework. *Information Retrieval Journal* 23, 3 (2020), 581–619.

[9] Enrique Amigó, Damiano Spina, and Jorge Carrillo-de Albornoz. 2018. An Axiomatic Analysis of Diversity Evaluation Metrics: Introducing the Rank-Biased Utility Metric. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. 625–634.

[10] Jacob Andreas, Andreas Vlachos, and Stephen Clark. 2013. Semantic Parsing as Machine Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (ACL '13)*. 47–52.

[11] Jaime Arguello, Adam Ferguson, Emery Fine, Bhaskar Mitra, Hamed Zamani, and Fernando Diaz. 2021. Tip of the Tongue Known-Item Retrieval: A Case Study in Movie Identification. In *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval (CHIIR '21)*. 5–14.

[12] Devansh Arpit, Stanisław Jastrzundefinedbski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A Closer Look at Memorization in Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*. 233–242.

[13] Leif Azzopardi, Maarten de Rijke, and Krisztian Balog. 2007. Building Simulated Queries for Known-item Topics: An Analysis Using Six European Languages. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '07)*. 455–462.

[14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Pro-*

ceedings of the 3rd International Conference on Learning Representations (ICLR '15).

[15] Krisztian Balog. 2014. Semistructured Data Search. In *Bridging Between Information Retrieval and Databases: PROMISE Winter School 2013, Bressanone, Italy, February 4-8, 2013. Revised Tutorial Lectures*. 74–96.

[16] Krisztian Balog. 2018. *Entity-Oriented Search*. The Information Retrieval Series, Vol. 39. Springer.

[17] Krisztian Balog. 2018. Entity Retrieval. In *Encyclopedia of Database Systems*. Springer, 1326–1331.

[18] Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-Based Question Answering with Knowledge Graph. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING '16)*. 2503–2514.

[19] Hannah Bast and Elmar Haussmann. 2015. More Accurate Question Answering on Freebase. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM '15)*. 1431–1440.

[20] 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 '13)*. 1533–1544.

[21] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia - A Crystallization Point for the Web of Data. *Journal of Web Semantics* 7, 3 (2009), 154–165.

[22] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale Simple Question Answering with Memory Networks. arXiv:1506.02075 [cs.LG]

[23] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer. 1992. An Estimate of an Upper Bound for the Entropy of English. *Computational Linguistics* 18, 1 (1992), 31–40.

[24] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems (NeurIPS '20, Vol. 33)*. 1877–1901.

[25] Luca Busin and Stefano Mizzaro. 2013. Axiometrics: An Axiomatic Approach to Information Retrieval Effectiveness Metrics. In *Proceedings of the 2013 Conference on the Theory of Information Retrieval (ICTIR '13)*. 22–29.

[26] Jianyu Cai, Zhanqiu Zhang, Feng Wu, and Jie Wang. 2021. Deep Cognitive Reasoning Network for Multi-Hop Question Answering over Knowledge Graphs. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. 219–229.

[27] Qingqing Cai and Alexander Yates. 2013. Large-scale Semantic Parsing via Schema Matching and Lexicon Extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL '13)*. 423–433.

[28] Marco Antonio Calijorne Soares and Fernando Silva Parreiras. 2020. A Literature Review on Question Answering Techniques, Paradigms and Systems. *Journal of King Saud University - Computer and Information Sciences* 32, 6 (2020), 635–646.

[29] Nilesh Chakraborty, Denis Lukovnikov, Gaurav Maheshwari, Priyansh Trivedi, Jens Lehmann, and Asja Fischer. 2021. Introduction to Neural Network-based Question Answering over Knowledge Graphs. *WIREs Data Mining and Knowledge Discovery* 11, 3 (2021), e1389.

[30] Junghwan Cho, Kyewook Lee, Ellie Shin, Garry Choy, and Synho Do. 2015. How Much Data Is Needed to Train a Medical Image Deep Learning System to Achieve Necessary High Accuracy? arXiv:1511.06348 [cs.LG]

[31] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP '14)*. Doha, Qatar, 1724–1734.

[32] Dan C. Cireşan, Ueli Meier, Jonathan Masci, Luca M. Gambardella, and Jürgen Schmidhuber. 2011. Flexible, High Performance Convolutional Neural Networks for Image Classification. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI '11)*. 1237–1242.

[33] Aaron Daniel Cohen, Adam Roberts, Alejandra Molina, Alena Butryna, Alicia Jin, Apoorv Kulshreshtha, Ben Hutchinson, Ben Zevenbergen, Blaise Hilary Aguera-Arcas, Chung ching Chang, Claire Cui, Cosmo Du, Daniel De Freitas Adiwardana, Dehao Chen, Dmitry (Dima) Lepikhin, Ed H. Chi, Erin Hoffman-John, Heng-Tze Cheng, Hongrae Lee, Igor Krivokon, James Qin, Jamie Hall, Joe Fenton, Johnny Soraker, Kathy Meier-Hellstern, Kristen Olson, Lora Mois Aroyo, Maarten Paul Bosma, Marc Joseph Pickett, Marcelo Amorim Menegali, Marian Croak, Mark D□az, Matthew Lamm, Maxim Krikun, Meredith Ringel Morris, Noam Shazeer, Quoc V. Le, Rachel Bernstein, Ravi Rajakumar, Ray Kurzweil, Romal Thoppilan, Steven Zheng, Taylor Bos, Toju Duke, Tulsee Doshi, Vincent Y. Zhao, Vinodkumar Prabhakaran, Will Rusch, YaGuang Li, Yanping Huang, Yanqi Zhou, Yuanzhong Xu, and Zhifeng Chen. 2022. LaMDA: Language Models for Dialog Applications. arXiv:2201.08239 [cs.CL]

[34] Tri Dao, Albert Gu, Alexander Ratner, Virginia Smith, Chris De Sa, and Christopher Re. 2019. A Kernel Theory of Modern Data Augmentation. In *Proceedings of the 36th International Conference on Machine Learning (ICML '19)*. 1528–1537.

[35] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (NAACL-HLT '19). 4171–4186.

[36] Dennis Diefenbach, Andreas Both, Kamal Deep Singh, and Pierre Maret. 2020. Towards a Question Answering System over the Semantic Web. *Semantic Web* 11, 3 (2020), 421–439.

[37] Dennis Diefenbach, José Giménez-García, Andreas Both, Kamal Singh, and Pierre Maret. 2020. QAnswer KG: Designing a Portable Question Answering System over RDF Data. In *Proceedings of the 2020 European Semantic Web Conference: The Semantic Web (ESWC '20)*. 429–445.

[38] Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. 2018. Core Techniques of Question Answering Systems over Knowledge Bases: A Survey. *Knowledge and Information Systems* 55, 3 (2018), 529–569.

[39] Dennis Diefenbach, Thomas Tanon, Kamal Singh, and Pierre Maret. 2017. Question Answering Benchmarks for Wikidata. In *Proceedings of the 16th International Semantic Web conference (ISWC '17)*.

[40] Heng Ding and Krisztian Balog. 2018. Generating Synthetic Data for Neural Keyword-to-Question Models. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '18)*. 51–58.

[41] Jiwei Ding, Wei Hu, Qixin Xu, and Yuzhong Qu. 2019. Leveraging Frequent Query Substructures to Generate Formal Queries for Complex Question Answering. 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 '19)*. 2614–2622.

[42] Li Dong and Mirella Lapata. 2016. Language to Logical Form with Neural Attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL '16)*. 33–43.

[43] Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. LC-QuAD 2.0: A Large Dataset for Complex Question

Answering over Wikidata and DBpedia. In *Proceedings of the 18th International Semantic Web Conference (ISWC '19)*. 69–78.

[44] Yixing Fan, Liang Pang, Jianpeng Hou, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2017. MatchZoo: A Toolkit for Deep Text Matching. arXiv:1707.07270 [cs.IR]

[45] Marco Ferrante, Nicola Ferro, and Eleonora Losiouk. 2020. How Do Interval Scales Help Us with Better Understanding IR Evaluation Measures? *Information Retrieval Journal* 23, 3 (2020), 289–317.

[46] Marco Ferrante, Nicola Ferro, and Maria Maistro. 2015. Towards a Formal Framework for Utility-Oriented Measurements of Retrieval Effectiveness. In *Proceedings of the 2015 International Conference on the Theory of Information Retrieval (ICTIR '15)*. 21–30.

[47] Marco Ferrante, Nicola Ferro, and Silvia Pontarollo. 2017. Are IR Evaluation Measures on an Interval Scale?. In *Proceedings of the ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '17)*. 67–74.

[48] Marco Ferrante, Nicola Ferro, and Silvia Pontarollo. 2019. A General Theory of IR Evaluation Measures. *IEEE Transactions on Knowledge and Data Engineering* 31, 3 (2019), 409–422.

[49] Francisco Florez-Revuelta. 2021. EvoSplit: An Evolutionary Approach to Split a Multi-Label Data Set into Disjoint Subsets. *Applied Sciences* 11, 6 (2021).

[50] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional Sequence to Sequence Learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML '17)*. 1243–1252.

[51] David Golub, Po-Sen Huang, Xiaodong He, and Li Deng. 2017. Two-Stage Synthesis Networks for Transfer Learning in Machine Comprehension. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP '17)*. 835–844.

[52] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[53] Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond I.I.D.: Three Levels of Generalization for Question Answering on Knowledge Bases. In *Proceedings of the Web Conference 2021 (WWW '21)*. 3477–3488.

[54] Yu Gu, Vardaan Pahuja, Gong Cheng, and Yu Su. 2022. Knowledge Base Question Answering: A Semantic Perspective. In *Proceedings of the 4th Conference on Automated Knowledge Base Construction (AKBC '22)*.

[55] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-Hoc Retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16)*. 55–64.

[56] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W. Bruce Croft, and Xueqi Cheng. 2020. A Deep Look Into Neural Ranking Models for Information Retrieval. *Information Processing & Management* 57, 6 (2020).

[57] Ann-Kathrin Hartmann, Edgard Marx, and Tommaso Soru. 2018. Generating a Large Dataset for Neural Question Answering over the DBpedia Knowledge Base. In *Workshop on Linked Data Management, co-located with the W3C WEBBR 2018 (WEBBR '18)*.

[58] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DEBERTA: Decoding-Enhanced Bert with Disentangled Attention. In *Proceedings of the 9th International Conference on Learning Representations (ICLR '21)*.

[59] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory F. Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. 2017. Deep Learning Scaling is Predictable, Empirically. arXiv:1712.00409 [cs.LG]

[60] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional Neural Network Architectures for Matching Natural Language Sen-

tences. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS '14)*. 2042–2050.

[61] Wenpeng Hu, Ran Le, Bing Liu, Jinwen Ma, Dongyan Zhao, and Rui Yan. 2020. Translation vs. Dialogue: A Comparative Analysis of Sequence-to-Sequence Modeling. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING '20)*.

[62] Xin Hu, Jiangli Duan, and Depeng Dang. 2021. Natural Language Question Answering over Knowledge Graph: The Marriage of SPARQL Query and Keyword Search. *Knowledge and Information Systems* 63, 4 (2021), 819–844.

[63] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning Deep Structured Semantic Models for Web Search Using Clickthrough Data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (CIKM '13)*. 2333–2338.

[64] Edwin T. Jaynes. 2003. *Probability Theory: The Logic of Science*. Cambridge University Press.

[65] Ida Kathrine Hammeleff Jørgensen and Toine Bogers. 2020. "Kinda like The Sims... But with Ghosts?": A Qualitative Analysis of Video Game Re-Finding Requests on Reddit. In *International Conference on the Foundations of Digital Games (FDG '20)*.

[66] V. Roshan Joseph and Akhil Vakayil. 2022. SPlit: An Optimal Method for Data Splitting. *Technometrics* 64, 2 (2022), 166–176.

[67] Daniel Jurafsky and James H. Martin. 2022. *Speech and Language Processing*. (3rd ed. draft, accessed online.).

[68] Khalid M. Kahloot and Peter Ekler. 2021. Algorithmic Splitting: A Method for Dataset Preparation. *IEEE Access* 9 (2021), 125229–125237.

[69] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and

Olivier Bousquet. 2020. Measuring Compositional Generalization: A Comprehensive Method on Realistic Data. In *Proceedings of the 8th International Conference on Learning Representations (ICLR '20)*.

[70] Unni Krishnan, Alistair Moffat, Justin Zobel, and Bodo Billerbeck. 2020. Generation of Synthetic Query Auto Completion Logs. In *Advances in Information Retrieval: Proceedings of the 42nd European Conference on IR Research (ECIR '20)*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). 621–635.

[71] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the Association for Computing Machinery* 60, 6 (2017), 84–90.

[72] Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A Survey on Complex Knowledge Base Question Answering: Methods, Challenges and Solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI '21)*. 4483–4491.

[73] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[74] Eelke B. Lenselink, Niels ten Dijke, Brandon Bongers, George Papadatos, Herman W. T. van Vlijmen, Wojtek Kowalczyk, Adriaan P. IJzerman, and Gerard J. P. van Westen. 2017. Beyond the Hype: Deep Neural Networks Outperform Established Methods Using a ChEMBL Bioactivity Benchmark Set. *Journal of Cheminformatics* 9, 1 (2017), 45.

[75] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *In Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop (ACL '04)*. 74–81.

[76] Trond Linjordet. 2020. Neural (Knowledge Graph) Question Answering Using Synthetic Training Data. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM '20)*. 3245–3248.

[77] Trond Linjordet and Krisztian Balog. 2019. Impact of Training Dataset Size on Neural Answer Selection Models. In *Advances in Information Retrieval (ECIR '19)*.

[78] Trond Linjordet and Krisztian Balog. 2020. Sanitizing Synthetic Training Data Generation for Question Answering over Knowledge Graphs. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval (ICTIR '20)*. 121–128.

[79] Trond Linjordet and Krisztian Balog. 2022. Would You Ask it that Way? Measuring and Improving Question Naturalness for Knowledge Graph Question Answering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 3090–3098.

[80] Trond Linjordet, Krisztian Balog, and Vinay Setty. 2022. Towards Formally Grounded Evaluation Measures for Semantic Parsing-based Knowledge Graph Question Answering. In *Proceedings of the 2022 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '22)*.

[81] Vanessa Lopez, Christina Unger, Philipp Cimiano, and Enrico Motta. 2013. Evaluating Question Answering over Linked Data. *Web Semant.: Science, Services and Agents on the World Wide Web* 21 (2013), 3–13.

[82] Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge Base Question Answering via Encoding of Complex Query Graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP '18)*. 2185–2194.

[83] Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP '15)*. 1412–1421.

[84] Gaurav Maheshwari, Priyansh Trivedi, Denis Lukovnikov, Nilesh Chakraborty, Asja Fischer, and Jens Lehmann. 2019. Learning to Rank Query Graphs for Complex Question Answering over Knowledge Graphs. In *Proceedings of the 18th International Semantic Web Conference (ISWC '19)*. 487–504.

[85] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval.* Cambridge University Press.

[86] John McCarthy. 1977. Epistemological Problems of Artificial Intelligence. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence (IJCAI '77).* 1038–1044.

[87] Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. 2020. Aligning Superhuman AI with Human Behavior: Chess as a Model System. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20).* 1677–1687.

[88] Bhaskar Mitra and Nick Craswell. 2017. Neural Models for Information Retrieval. arXiv:1705.01509 [cs.IR]

[89] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to Match Using Local and Distributed Representations of Text for Web Search. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17).* 1291–1299.

[90] Raymond J Mooney. 2007. Learning for Semantic Parsing. In *International Conference on Intelligent Text Processing and Computational Linguistics (CICLing '07).* 311–324.

[91] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. (2016).

[92] Thi Huyen Nguyen, Hoang H. Nguyen, Zahra Ahmadi, Tuan-Anh Hoang, and Thanh-Nam Doan. 2021. On the Impact of Dataset Size: A Twitter Classification Case Study. In *Proceedings of the 20th IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT '21).* 210–217.

[93] Christos Nikas, Pavlos Fafalios, and Yannis Tzitzikas. 2021. Open Domain Question Answering over Knowledge Graphs Using Keyword Search, Answer Type Prediction, SPARQL and Pre-trained Neural Models. In *Pro-*

ceedings of the 20th International Semantic Web Conference (ISWC '21). 235–251.

[94] Sergey I. Nikolenko. 2021. *Synthetic Data for Deep Learning*. Springer Optimization and Its Applications, Vol. 174. Springer Cham.

[95] Kezban Dilek Onal, Ye Zhang, Ismail Sengor Altingovde, Md Mustafizur Rahman, Pinar Karagoz, Alex Braylan, Brandon Dang, Heng-Lu Chang, Henna Kim, Quinten McNamara, Aaron Angert, Edward Banner, Vivek Khetan, Tyler McDonnell, An Thanh Nguyen, Dan Xu, Byron C. Wallace, Maarten de Rijke, and Matthew Lease. 2018. Neural Information Retrieval: At the End of the Early Years. *Information Retrieval Journal* 21, 2 (2018), 111–182.

[96] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI '16)*. 2793–2799.

[97] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02)*. 311–318.

[98] Javier Parapar and Filip Radlinski. 2021. Towards Unified Metrics for Accuracy and Diversity for Recommender Systems. In *Proceedings of the Fifteenth ACM Conference on Recommender Systems (RecSys '21)*. 75–84.

[99] Panupong Pasupat and Percy Liang. [n.d.]. Compositional Semantic Parsing on Semi-Structured Tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (ACL '15)*.

[100] 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 (EMNLP '18)*. 554–558.

[101] Joan Plepi, Endri Kacupaj, Kuldeep Singh, Harsh Thakkar, and Jens Lehmann. 2021. Context Transformer with Stacked Pointer Networks for Conversational Question Answering over Knowledge Graphs. In *Proceedings of the 2021 European Semantic Web Conference: The Semantic Web (ESWC '21)*. 356–371.

[102] Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W. Bruce Croft, and Mohit Iyyer. 2020. Open-Retrieval Conversational Question Answering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 539–548.

[103] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. 2007. Self-Taught Learning: Transfer Learning from Unlabeled Data. In *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*. 759–766.

[104] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP '16)*. 2383–2392.

[105] Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-Scale Semantic Parsing without Question-Answer Pairs. (2014).

[106] Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2022. QA Dataset Explosion: A Taxonomy of NLP Resources for Question Answering and Reading Comprehension. *ACM Computing Surveys* (2022).

[107] Rishiraj Saha Roy and Avishek Anand. 2022. *Question Answering for the Curated Web: Tasks and Methods in QA over Knowledge Bases and Text Collections*. Synthesis Lectures on Information Concepts, Retrieval, and Services, Vol. 76. Morgan & Claypool Publishers.

[108] Dana Ruiter, Josef van Genabith, and Cristina España-Bonet. 2020. Self-Induced Curriculum Learning in Self-Supervised Neural Machine Translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP '20)*. 2560–2571.

[109] Bertrand Russell. 2009. *Our Knowledge of the External World: As a Field for Scientific Method in Philosophy*. Routledge. Original work published 1914.

[110] Tetsuya Sakai and Zhaohao Zeng. 2019. Which Diversity Evaluation Measures Are "Good"?. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. 595–604.

[111] Gerard Salton and Donna Harman. 2003. Information Retrieval. In *Encyclopedia of Computer Science*. John Wiley and Sons Ltd., 858–863.

[112] Jürgen Schmidhuber. 2015. Deep Learning in Neural Networks: An Overview. *Neural Networks* 61 (2015), 85–117.

[113] Fabrizio Sebastiani. 2015. An Axiomatically Derived Measure for the Evaluation of Classification Algorithms. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval (ICTIR '15)*. 11–20.

[114] Fabrizio Sebastiani. 2020. Evaluation Measures for Quantification: An Axiomatic Approach. *Information Retrieval Journal* 23, 3 (2020), 255–288.

[115] Yasser Seirawan, Herbert A. Simon, and Toshinori Munakata. 1997. The Implications of Kasparov vs. Deep Blue. *Commun. ACM* 40, 8 (1997), 21–25.

[116] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural Machine Translation of Rare Words with Subword Units. arXiv:1508.07909 [cs.CL]

[117] Tao Shen, Xiubo Geng, Tao Qin, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-Task Learning for Conversational Question Answering over a Large-Scale Knowledge Base. 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 '19)*. 2442–2451.

[118] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning Semantic Representations Using Convolutional Neural Networks for Web Search. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*. 373–374.

[119] Jiaxin Shi, Shulin Cao, Liangming Pan, Yutong Xiang, Lei Hou, Juanzi Li, Hanwang Zhang, and Bin He. 2020. KQA Pro: A Large Diagnostic Dataset for Complex Question Answering over Knowledge Base. arXiv:2007.03875 [cs.CL]

[120] Connor Shorten and Taghi M. Khoshgoftaar. 2019. A Survey on Image Data Augmentation for Deep Learning. *Journal of Big Data* 6, 60 (2019).

[121] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature* 529, 7587 (2016), 484–489.

[122] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. 2017. Mastering the Game of Go without Human Knowledge. *Nature* 550, 7676 (2017), 354–359.

[123] Kuldeep Singh, Ioanna Lytra, Arun Sethupat Radhakrishna, Saeedeh Shekarpour, Maria-Esther Vidal, and Jens Lehmann. 2020. No One Is Perfect: Analysing the Performance of Question Answering Components over the DBpedia Knowledge Graph. *Journal of Web Semantics* 65 (2020).

[124] Tommaso Soru, Edgard Marx, Diego Moussallem, Gustavo Publio, André Valdestilhas, Diego Esteves, and Ciro Baron Neto. 2017. SPARQL as a Foreign Language. (2017). arXiv:1708.07624 [cs.CL]

[125] Tommaso Soru, Edgard Marx, André Valdestilhas, Diego Esteves, Diego Moussallem, and Gustavo Publio. 2018. Neural Machine Translation for

Query Construction and Composition. In *ICML Workshop on Neural Abstract Machines & Program Induction (NAMPI v2) (ICML '18)*.

[126] Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gür, Zenghui Yan, and Xifeng Yan. 2016. On Generating Characteristic-rich Question Sets for QA Evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP '16)*. 562–572.

[127] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. 2017. Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV '17)*. 842–852.

[128] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS '14)*. 3104–3112.

[129] Alon Talmor and Jonathan Berant. 2018. The Web as a Knowledge-Base for Answering Complex Questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (NAACL-HLT '18)*. 641–651.

[130] Mohamed Trabelsi, Zhiyu Chen, Brian D. Davison, and Jeff Heflin. 2021. Neural Ranking Models for Document Retrieval. *Information Retrieval Journal* 24, 6 (2021), 400–444.

[131] Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. 2017. LC-QuAD: A Corpus for Complex Question Answering over Knowledge Graphs. In *Proceedings of the 16th International Semantic Web Conference (ISWC) (ISWC '17)*. 210–218.

[132] Alan Mathison Turing. 1950. I.–Computing Machinery and Intelligence. *Mind* LIX, 236 (1950), 433–460.

[133] Christina Unger, Corina Forascu, Vanessa Lopez, Axel-Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano, and Sebastian Walter. 2014. Ques-

tion Answering over Linked Data (QALD-4). In *Working Notes of the 7th Conference and Labs of the Evaluation Forum (CLEF '14)*. 1172–1180.

[134] Christina Unger, André Freitas, and Philipp Cimiano. 2014. *An Introduction to Question Answering over Linked Data.* 100–140.

[135] Ricardo Usbeck, Ria Hari Gusmita, Axel-Cyrille Ngonga Ngomo, and Muhammad Saleem. 2018. 9th Challenge on Question Answering over Linked Data (QALD-9). In *Joint proceedings of the 4th Workshop on Semantic Deep Learning (SemDeep-4) and NLIWoD4: Natural Language Interfaces for the Web of Data (NLIWOD-4) and 9th Question Answering over Linked Data challenge (QALD-9), co-located with the 17th International Semantic Web Conference (ISWC '18, Vol. 2241)*. 58–64.

[136] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Bastian Haarmann, Anastasia Krithara, Michael Röder, and Giulio Napolitano. 2017. 7th Open Challenge on Question Answering over Linked Data (QALD-7). In *Proceedings of Semantic Web Challenges - 4th SemWebEval Challenge (ESWC '17)*. 59–69.

[137] Ricardo Usbeck, Michael Röder, Michael Hoffmann, Felix Conrads, Jonathan Huthmann, Axel-Cyrille Ngonga-Ngomo, Christian Demmler, and Christina Unger. 2019. Benchmarking Question Answering Systems. *Semantic Web* 10, 2 (2019), 293–304.

[138] Svitlana Vakulenko, Javier David Fernandez Garcia, Axel Polleres, Maarten de Rijke, and Michael Cochez. 2019. Message Passing for Complex Question Answering over Knowledge Graphs. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*. 1431–1440.

[139] Jesper E. van Engelen and Holger H. Hoos. 2020. A Survey on Semi-Supervised Learning. *Machine Learning* 109, 2 (2020), 373–440.

[140] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS '17)*. 6000–6010.

[141] Ellen M. Voorhees. 2001. The TREC Question Answering Track. *Natural Language Engineering* 7, 4 (2001), 361–378.

[142] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI '16)*. 2835–2841.

[143] Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a Semantic Parser Overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJNLP '15)*. 1332–1342.

[144] Terry Winograd. 1972. Understanding natural language. *Cognitive Psychology* 3, 1 (1972), 1–191.

[145] Peiyun Wu, Xiaowang Zhang, and Zhiyong Feng. 2019. A Survey of Question Answering over Knowledge Base. In *Knowledge Graph and Semantic Computing: Knowledge Computing and Language Understanding*. 86–97.

[146] Liu Yang, Qingyao Ai, Jiafeng Guo, and W. Bruce Croft. 2016. aNMM: Ranking Short Answer Texts with Attention-Based Neural Matching Model. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16)*. 287–296.

[147] Qian Yang, Zhouyuan Huo, Dinghan Shen, Yong Cheng, Wenlin Wang, Guoyin Wang, and Lawrence Carin. 2019. An End-to-End Generative Architecture for Paraphrase 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-IJNLP '19)*. 3132–3142.

[148] Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A Challenge Dataset for Open-Domain Question Answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP '15)*. 2013–2018.

[149] Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (ACL-IJCNLP '15)*. 1321–1331.

[150] Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The Value of Semantic Parse Labeling for Knowledge Base Question Answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (ACL '16)*. 201–206.

[151] Xiaoyu Yin, Dagmar Gromann, and Sebastian Rudolph. 2021. Neural Machine Translating from Natural Language to SPARQL. *Future Generation Computer Systems* 117 (2021), 510–519.

[152] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP '18)*. 3911–3921.

[153] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding Deep Learning Requires Rethinking Generalization. In *Proceedings of the 5th International Conference on Learning Representations (ICLR '17)*.

[154] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding Deep Learning (Still) Requires Rethinking Generalization. *Commun. ACM* 64, 3 (2021), 107–115.

[155] Jinghua Zhang, Chen Li, Yimin Yin, Jiawei Zhang, and Marcin Grzegorzek. 2022. Applications of artificial neural networks in microorganism image analysis: a comprehensive review from conventional multilayer perceptron to popular convolutional neural network and potential visual transformer. *Artificial Intelligence Review* (2022).

[156] Shiyue Zhang and Mohit Bansal. 2019. Addressing Semantic Drift in Question Generation for Semi-Supervised Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJNLP '19)*. 2495–2509.

University of Stavanger

4036 Stavanger

Tel: +47 51 83 10 00

E-mail: post@uis.no

www.uis.no

Cover Illustration: from ``Kunstformen der Natur'' (1904) by Ernst Haeckel.
Used under Creative Commons CC0 license, public domain.