CLARIFICATION QUESTION GENERATION

Lokesh JK
AI21MTECH14001

Karthikeyan Mohanraj
AI21MTECH14007

Tejas Arya
AI21MTECH14004

Kaushiki Dwivedi
AI21MTECH14003

Sarvani Mathigetta
SM21MTECH12004

Abstract

The problem statement is to create a model that asks questions to fill information gaps of the posted question, typically through generating clarification questions. [1]

1. Introduction

Identifying missing information in the given context which is currently missing from text is an under explored aspect of text understanding. Recently proposed task of clarification question generation can aid machine learning models reduce the ambiguity in a given context. Rao and Daumé III (2018, 2019) proposed models for this task which is successful at generating fluent and relevant questions but still falls short in terms of usefulness and identifying missing information.Even with advent of large-scale pre-trained generative models (Radford et al., 2019; Lewis et al., 2019; Raffel et al., 2019) were not successful in going beyond fluency and relevance.To do so, we must first recognise what is lacking, which, if included, would be beneficial to the consumer of the information. This could be achieved by taking humans as inspiration , who in general identify missing information by using global knowledge,i.e recollecting previous experiences and comparing them to the current one to ascertain what information is missing and if added would be the most useful. This project is about inculcating the above mentioned idea into the model so that it generates clarification questions on missing information.

1.1. Motivation

The main motivation is with the advent of high quality speech recognition and text generation systems, we are increasingly using dialog as a mode to interact with devices (Clark et al., 2019). However, these dialog systems still struggle when faced with ambiguity and could greatly benefit from having the ability to ask clarification questions.

[1]Our code is available at https://github.com/LokeshJatangi/Diverse_specific_clarification_questions

2. Literature Review

In paper[1] the author describes a novel approach to the problem of clarification question generation and proposed a Generative Adversarial Network (GAN) based model where the generator is a sequence-to-sequence model that generates questions, and the discriminator is a utility function that models the value of updating the context with the answer to the clarification question. They developed three model variants and compared their performance with the baseline model. The models were trained with Amazon product description and stack Exchange posts datasets, and they evaluated the model on automated evaluation metrics and crowd sourced human judgments. Finally, they concluded that the model with an adversarial training approach produces more useful and specific questions compared to both a model trained using maximum likelihood objective and a model trained using utility reward-based reinforcement learning.

In paper[2], the author formulated the task to tackle the problem of missing information in the product description on e-commerce websites by proposing the task of Diverse CQGen to request various unstated aspects in writing with a group of semantically different questions. To deal with the specificity challenge, determined by the size of its applicable range, proposed a novel model named Keyword Prediction and Conditioning Network(KPCNet) Keywords in CQs. Its keywords can capture the main semantic of a question. The clustering method also explored a novel use of producing coherent keyword groups for keyword selection to generate the correct, specific, and diverse questions. The model is trained on the Home and Kitchen category of the Amazon dataset. The model is evaluated on both automatic metrics and human judgments. They concluded that the model covers various information needs and improves the robustness to problematic generations.

In paper [3], authors described a model that identifies useful missing information in each context (schema) i.e., it

generates clarification questions to reduce ambiguity. They stated that the model fills the missing information from global knowledge or from previous experience just like how humans do. In the first stage, they found what is missing by taking a difference between the global knowledge's schema and schema of the local context and then fed that missing schema to a fine-tuned BART-model to generate a question which is further made more useful using PPLM. They tested this model on two scenarios community-QA (product-description from amazon.com) and dialog history from the Ubuntu Chat forum and evaluated on Automatic and human judgment metrics. Finally, they concluded that the framework works across domains, shows robustness towards information availability, and responds to the dynamic change in global knowledge.

In paper[4], authors investigated the problem of generating informative questions in information asymmetric conversations. In this paper they worked on the scenario where the questionnaire is not given the context from which answers are drawn. The core challenges they worked upon are defining the informativeness of potential questions and exploring the prohibitively large space of potential questions to find good candidates. This paper is the first attempt at studying question generation to seek information in open-domain communication. Authors found out that optimizing metrics(quantify how much new information question reveal) to quantify informativeness of questions via reinforcement learning leads to a better system that behaves pragmatically and has improved communication efficiency.

In paper[5], the author formulated the task of asking clarifying questions in open-domain information-seeking conversational systems. Proposed an offline evaluation methodology for the task and collected a dataset, called Qulac (Questions for lack of clarity), through crowd sourcing. Dataset is built on top of the TREC Web Track 2009-2012 data collections and consists of over 10K question-answer pairs for 198 TREC topics with 762 facets. At the second stage, the proposed model, called NeuQS, aims to select the best question to be posed to the user based on the query and the conversation context. The Question Retrieval Model is described as the BERT- Language Representation based, called BERT-LeaQuR. The aim is to maximize the recall of the retrieved questions, retrieving all relevant clarifying questions to a given query in the top k questions. Hence, illustrated the workflow of a conversational search system, focusing on asking clarifying questions, addressing all the facets in the collection.

In paper [6], introduce the task of clarification question generation about missing information in a given context.They developed a model which is inspired by Expected value of perfect information (EVPI) in which joint neural network composed of LSTM's, models the probability distribution of answer given a (post , question) tuple by generating answer representation and also models a Utility function that calculates utility of updated post as a binary classification problem.The authors have defined task as ranking the most useful questions , and they evaluated against expert human annotations using a StackExchange dataset made up of (posts, questions, and responses) triples.Finally, The authors conclude EVPI model with answer candidates is a promising formalism for the clarification question generation task as they outperform neural baseline models.

3. Problem Setup and Scenarios

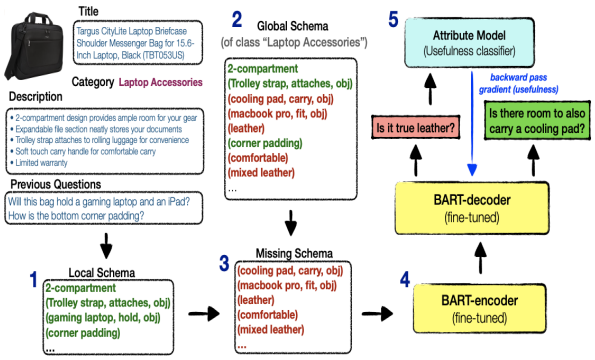

Figure 1. Test-time behaviour of our proposed model for useful clarification question generation based on missing information in a Community-QA (amazon.com) setup. 1. We obtain a local schema from the available context for a product: description and previously asked questions. 2. We obtain the global schema of the category of the product. 3. We estimate the missing schema that is likely to guide clarification question generation. 4. A BART model fine-tuned on (missing schema, question) pairs to generate a question ("Is it true leather?"). 5. A PPLM model with usefulness classifier as its attribute model further tunes the generated question to make it more useful ("Is there room to also carry a cooling pad?")

Our problem statement is vaguely explained in the Figure 1. According to Rao and Daumé III (2018) the task of clarification question generation is termed as generation of a question from the given context by identifying the missing information in the context.We considered the following scenario:

Community-QA: Community-driven question answering has become a trendy way to get answers from the crowd. People frequently ask questions from these forums which have some initial context. We considered the Amazon question-answer dataset (McAuley and Yang, 2016) where the setting is a product description, and the aim is to produce a clarification question that helps a potential buyer better understand the product.

4. Approach

We proposed a two-stage approach for the task of generating a clarification question. In the first stage, we aimed to find the missing information from the given context. First, for each high-level class, we group together all similar contexts in our data to form a global schema. We then extracted the schema of the given context to form a local schema. Finally, we determine the missing schema for the current context by comparing the local and global schemas (of the class to which the context belongs). In the second stage, we feed these (missing schema, questions) pairs to our fine-tuned model BART to generate a clarification question as an output.

4.1. Identifying the missing information

Schema Definition According to (Khashabi et al., 2017) performance of the Question Answering system is improved when essential terms of a question's used. Motivated by this we extracted essential terms which we refer to as a schema in our work. The schema of sentence s is defined as a set of one or more triples consisting of (key phrase, verb, relation) and/or one or more key phrases.

$$schema_s = \{element\}, where$$
$$element \in \{(keyphrase, verb, relation), keyphrase\} \quad (1)$$

Schema Extraction Aiming for the extraction of the schema we adopted (key-phrase, verb, relation) as the basic element of schema in reference to the work (Vedula et al., 2019). For a given sentence in the context, we first extract unigram and bigram key phrases by using YAKE (Yet-Another-Keyword-Extractor) and retain only those that have at least one noun. By obtaining the dependency parse tree we mapped the key phrases to tree nodes. This procedure is described in Algorithm 1. To build a relationship between the key and the verb, we use the path between the key and the nearest verb in the dependency tree at a high level. We only utilize the key phrase as our schema element when there is no path. Figure 2 shows an example dependency tree for a sentence.

Creating local schema Extract schema for each sentence *s* for the given context. Local schema for the given context *c* is defined as the union of schemata of each sentence *s* in the context.

$$local\_schema_c = \cup_{s \in c} schema_s \quad (2)$$

Creating global schema Global schema is defined at class level, where class is a subset of several similar contexts. For Amazon, classes consist of groups of similar products. The global schema of a class *K* is a union of local schemata of all contexts *c* belonging to *K*.

$$global\_schema_K = \cup_{c \in K} local\_schema_c \quad (3)$$

A naive union of all local schemata can produce a global schema with low-frequency elements.

Creating missing schema Given a context *c*, we first determine the class *K* to which the context belongs. Missing schema is computed by taking the set difference between

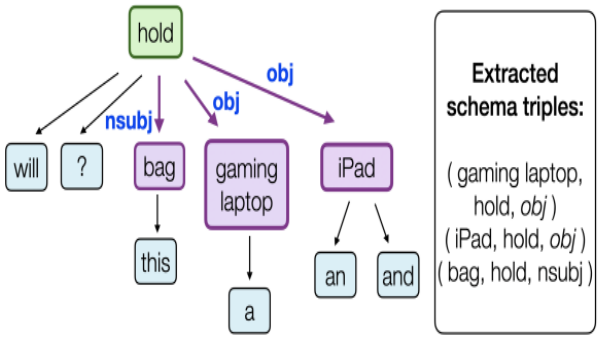

Figure 2. Dependency tree and paths showing how we obtain schema triples for sentence: "Will this bag hold a gaming laptop and an iPad?" (from Figure 1).

Algorithm 1 Pseudocode for extracting (key-phrase, verb, relation) triple

Initialize with empty path (path length ∞) for all possible pairs of verbs $\in$ (VB, VBG, VBZ) and key-phrases in the sentence
for Each verb and key-phrase pair **do**
  Search for the key-phrase among the children of the verb in the dependency tree
  if A key-phrase is found and path is shorter than the stored path **then**
    Update the path between the key-phrase and the verb pair
  end if
end for
for Each verb and key-phrase pair **do**
  if The key-phrase is the immediate child of the verb **then**
    Create the triple (key-phrase, verb, relation) using the relation in the path
  else
Traverse backward from the key-phrase, stop at the first verb, use the relation with its immediate child in the path to create (key-phrase, verb, relation)
  end if
end for

the global schema of class K and the local schema of the context c:

$$missing\_schema_c = global_K \backslash local_c \quad (4)$$

4.2. Generating Useful Questions

In this section we look at how we generate clarification questions by using missing information.

BART based generation model
BART is an encoder and decoder model. We start with a pre-trained BART model that has a 6-layer encoder and decoder. We fine-tune this model on the data where inputs are (missing schema, question) pairs and output is a question. The elements of the missing schema in the input are separated by a special [SEP] token. Since the elements in our input do not have any order, we use the same positional encoding for all input positions. We use a token type embedding layer with three types of tokens: key-phrases, verbs, and relations.

PPLM-based decoder
BART does not provide useful questions each time. To overcome this aspect, it is suggested to use a usefulness classifier. We propose a classifier Plug-and-Play-Language-Model (PPLM) (Dathathri et al., 2019) during decoding (at test time).The attribute model of the PPLM in our case is a usefulness classifier trained on bags-of-words of questions. In order to train such a classifier, we need usefulness annotations on a set of questions. For the Amazon dataset, we collect usefulness scores (0 or 1) on 5000 questions using human annotation.

5. Datasets

Amazon Review Dataset: The Amazon review dataset (McAuley et al., 2015) consists of descriptions of products on amazon.com. It consists of Product ids, title, category,description, product reviews of N products.
Amazon QuAC dataset:The Amazon question-answering dataset (McAuley and Yang, 2016) consists of questions (and answers) asked about products and the task is to generate a clarification question that helps a potential buyer better understand the product.Now our dataset combines the above two datasets based on product-ids and hence it consists of product-ids, title, category, description,questions about product. Given a product description and N questions asked about the product, we create N instances of (context, question) pairs where context consists of the description and previously asked questions.

6. Implementation

6.1. Schema Generation

The model architecture explains a two-stage approach,for the task of clarification question generation. In the first stage, we identify the missing information in a given context. We consider (keyphrase, action verb, relation) as the basic element of our schema. Such triples have been found to be representative of key information in previous work (Vedula et al., 2019)
The missing schema is constructed as follows: For a particular product id,we have title,description,category,questions which forms the local schema for a product by extracting the context.

For instance, here we take an example of description of products and create global schema by appending all the unique descriptions together(for a context of same class).Likewise is done for product questions also. Now for missing schema we take a difference between the global schema and local schema to obtain missing schema.

6.2. Data preprocessing

Firstly, we define schema of sentences as set consisting of one or more triples of the form (key-phrase, verb, relation) and/or one or more key-phrases.
schema = element ; where element (key-phrase, verb, relation), key-phrase.
As an input here we pass as an example a sequence of concatenated particular schema: Title [SEP] category [SEP] description keywords[SEP] (which includes the list of separated keywords).The output we get is a list of questions all separated by separator [SEP].
We created a new dataframe finally consisting of the product-ids,title,category,description-schema,table-schema,questions.In the dataframe there are 356 rows and 5 columns .To add separator between the key-phrase,verb,relation we add [SEP] to form the question-schema.
The title,category,descriptions are flattened together to form title_category_desc column.The final dataframe consists of question-schema,questions as labels,title_category_desc columns.Thus there are 356 rows and 3 columns.

6.3. Training the model

In the second stage, we train a model to generate a question about the most useful information in the missing schema. For this, we fine-tune a BART model (Lewis et al., 2019) on (missing schema, question) pairs.The tokenizer used here is **BartTokenizer**. We finetune this model on our data where the inputs are the missing schema and the output is the question. The elements of the missing schema in the input are separated by a special [SEP] token.Since the elements in our input do not have any order, we use the

same positional encoding for all input positions. We use a token type embedding layer with three types of tokens: key-phrases, verbs, and relations.

6.4. Sequence2Sequence Model

Sequence-to-sequence models are best suited for tasks revolving around generating new sentences depending on a given input, such as summarization, translation, or generative question answering.The model here used is **BartForConditionalGeneration**. The encoder_max_length = 512 and decoder_max_length = 128 considered. The dataframe is split into train and validation data with test_size = 0.1. In the train_data we have 320 rows and validation data consists of 36 rows.For passing into the BART model we tokenize the data labels and then batch preprocess it.

The trainer used here is the **Seq2SeqTrainer** . by passing it all the objects constructed the model, the training and validation datasets,data collator and the tokenizer.With the following hyperparameters: per_device_train_batch_size= 4,num_train_epochs= 50. The function that is responsible for putting together samples inside a batch is called collate function. Transformers library provides us with such a function via Collator with Padding. It takes a tokenizer when it is instantiated (to know which padding token to use, and whether the model expects padding to be on the left or on the right of the inputs). Dynamic padding means the samples in this batch should all be padded to a length of the maximum length inside the batch. Thus the model after decoding generates by taking as input the following parameters input_ids,attention_mask,max_length,num_beams.The outputs are the clarification questions generated by the model for product ids by the BART model.

7. Evaluation Metrics

7.1. BLEU

Bilingual Evaluation Understudy is a metric for machine translation. The BLEU score is a number between zero and one that measures the similarity of the machine-translated text to a set of high-quality reference translations. BLEU scores are used as a benchmark for text generation, used Pairwise –BLEU and Avg BLEU as an evaluation metric for diverse machine translation.

7.2. Distinct-3

Distinct-3 is an algorithm for evaluating the textual diversity of the generated text by calculating the number of distinct n-grams.

7.3. METEOR

METEOR (Metric for Evaluation of Translation with Explicit Ordering) is a metric for evaluating the machine-translated output. It is used for individual-level automatic evaluation.

8. Experimental Results

Model Name	Product Description	Output question
BART	Nikon ML L3 Wireless Remote Control for Nikon Coolpix P7000, P7100, P7700, D40, D40x, D50, D60, D70, D70S, D80.	Will this item work on a Nikon P7100 camera?

Figure 3. BART Model generation for an example product from Amazon.

Model Name	Product Description	Output question
BART + Miss_info	Dymo LabelWriter DUO 300dpi 55 labels per minute Label Printer; 180dpi D1 tape Label Printer	Will this work with a projector?

Figure 4. BART+Miss_info Model generation for an example product from Amazon.

From Fig:3 It is clear that the BART model is able to generate clarification question pertaining to the given description. But, the BART+Miss_info model output as in Fig.4 is able to generate question which is able to ask information that is not available in the description implies it is able to ask more useful questions.

9. Conclusion

The BART model used to generate clarification questions based on the missing information obtained by taking difference between Global knowledge and local view. The questions generated were from missing schema keywords.

10. Code Link

Our code is available on https://github.com/LokeshJatangi/CQ-gen-using-Global-Knowledge

References

[1] Rao, S. and Daumé III, H., 2019. " Answer-based adversarial training for generating clarification questions." arXiv preprint arXiv:1904.02281.

[2] Zhang, Z. and Zhu, K., 2021, April. " Diverse and Specific Clarification Question Generation with Keywords ". In Proceedings of the Web Conference 2021 (pp. 3501-3511).

[3] Majumder, B.P., Rao, S., Galley, M. and McAuley, J., 2021. " Ask what's missing and what's use-

ful: Improving Clarification Question Generation using Global Knowledge. ” arXiv preprint arXiv:2104.06828.

[4] Qi, P., Zhang, Y. and Manning, C.D., 2020. ” Stay hungry, stay focused: Generating informative and specific questions in information-seeking conversations. ” arXiv preprint arXiv:2004.14530.

[5] Aliannejadi, M., Zamani, H., Crestani, F. and Croft, W.B., 2019, July. ” Asking clarifying questions in open-domain information-seeking conversations.” In Proceedings of the 42nd international acm sigir conference on research and development in information retrieval (pp. 475-484).

[6] Rao, S. and Daumé III, H., 2018. ” Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information.” arXiv preprint arXiv:1805.04655.

[7] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 1532–1543.

[8] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 3111– 3119.