Table 1: The most common 4-grams in clarifying question answers [22] and their corresponding questions with example generations from GPT-2 for the query "I am looking for information about South Africa." Duplicated Examples are omitted.

1st word	2nd word	3rd word	4th word	Corresponding question	Example
Yes	I	would	like	would you like to	would you like to [take pictures of]
		want	to	do you want to	do you want to [see pictures of them]
No	I	am	interested	are you interested in	are you interested in [taking pictures of them]
			looking	are you looking for	are you looking for [pictures of South Africa]
		just	want	do you want to	...
			need	do you need to	do you need to [send pictures to us]
		need	to	do you need to	...
			information	do you need information	do you need information [or pictures of South Africa]
		want	information	do you want information	do you want information [and pictures of South Africa]
			the	do you want to	...
			to	do you want to	...
		would	like	would you like to	...
	Im	looking	for	are you looking for	...
I	am	looking	for	are you looking for	...
	dont	know	.	do you want to know	do you want to know [about the pictures and videos]
	want	to	know	do you want to know	...
	would	like	to	would you like to	...

(3) **Grouping Step:** Group the remaining candidates by the facet words they contain. This will result in $2^{|f|}$ groups. However, notice that some group could be empty.
(4) Keep the best candidate from each of the groups. Again, there may be less than $2^{|f|}$ candidates left now. Just keep at most the best k candidates with the highest $p(x_{1:t})$ in the beam and move onto decoding the $(t+1)$th token.

We recommend a more vivid demonstration in the original paper [29]. We now explain why NeuroLogic Decoding could better constrain the decoder to generate facet-related questions by highlighting some key steps. First, the top-β filtering in step (2) is the main reason for promoting facet words in generations. Because of this filtering, NeuroLogic Decoding tends to discard generations with fewer facet words regardless of their generation probability. Therefore, facet-related generations with low probability will more likely stand out against greedy high-probability generations without using facet words. Then, the grouping in step (3) is the key for NeuroLogic Decoding to explore as many branches as possible. Because this grouping method keeps the most cases ($2^{|f|}$) of facet word inclusions, allowing the decoder to cover the most possibilities of ordering constraints in generation.

As mentioned in [29], the asymptotic runtime of NeuroLogic Decoding is $O(NK)$, where N is the text sequence length and k is beam search size. This is the same as normal beam search and faster than most previous constrained language generation algorithms, making it fairly applicable in real cases.

3.2 Multiform Question Prompting and Ranking

Another challenge is guiding zero-shot GPT-2 to generate clarifying questions instead of narrative or other types of questions. We use clarifying question templates as the starting text of the generation and let the decoder generate the rest of question body. For example, if the query is "I am looking for information about South Africa." Then we give the decoder "I am looking for information about South Africa. [SEP] would you like to know" as input and let it generate the rest. From our observation, GPT-2 is much better at finishing a question like this than asking a new question by itself.

In our system, we use multiple prompts because we want to both cover more ways of clarification with different prompts and avoid making users bored with monotonic questions. A previous study about the effect of clarifying question [22] shows the most common 4-grams for answering clarifying questions, as shown in Table 1. Inspired by their work, we reverse these most common answers to their original question forms (eight in total) and use them as our prompt candidates. For each query, we will append these eight prompts to the query and form eight inputs. Eventually, we will generate eight clarifying question candidates. For example, our generated questions for the query "I am looking for information about South Africa." with facet "map" is shown in Table 1.

In real applications, our system should return one question in the form of the concatenation of the prompt and the GPT-2 output. To find the best question, we explore various ranking methods to rank our prompted generations:

Perplexity is a commonly used method that ranks clarifying questions by the perplexity of query-question concatenation computed by pre-trained and GPT-2.

AutoScore is another commonly used method that ranks clarifying questions by weighted sum of automatic natural language generation scores including BLEU [35], ROUGE [25], and METEOR [5]. These scores are computed using generated questions as hypotheses and queries as references.

Cross-encoder [20] is a typical and commonly used dense retrieval structure. It ranks question candidate by its relevance which is computed by a transformer encoder followed by a linear scoring layer. The cross-encoder is pre-trained on millions of Reddit dialogues [31]. Directly using the pre-trained checkpoint is potentially suboptimal because the prompted generation ranking objective differs from the pretraining task.

NTES [34] is a clarifying question ranking model that wins the ConvAI3 challenge [1] on ClariQ dataset. The clarifying question ranking subtask in ConvAI3 requires a system to rank clarifying question candidates given query and facet. We consider this task highly similar to our prompted generation ranking task. The NTES model finetunes pretrained ELECTRA [9] as its ranker on ClariQ dataset. We use their finetuned checkpoint and aim to leverage the clarifying question ranking knowledge in this model.

Weighted Sequential Dependency Model (WSDM) [7] is a document ranking method based on *query-candidate* overlap in terms of unigram, and ordered/unordered bigram within a context window. Our system treats the center words in the original query together with facet words as *query* and prompted generation as *candidates*. The motivation of WSDM is to rank those generations with facet words co-occurring together higher. For example, given the query "I'm looking for information about South Africa" and facet "map", the WSDM model will rank all the prompted questions using "South Africa map" as the query. Questions like "do you need information about the map of South Africa" will be ranked higher than "do you want to buy a map that is made in South Africa" because "South Africa" and "map" are closer in the first question and more likely to be more useful clarifications.

We have empirically compared different versions of our model using the ranking methods above in the experiments and find that WSDM achieves the best empirical performance overall. Thus, if not mentioned, we use WSDM as our primary ranker.

4 Experiments

4.1 Research Questions and Experiment Design

We design experiments to answer the following research questions:

RQ1. How well can we do in zero-shot clarifying question generation with existing baselines?

In this research question, we show the performance of some clarifying question generation baselines in the zero-shot setting:

(1) **Q-GPT-0** simply uses GPT-2 to generate the clarifying question given the query. This and the next baseline are the zero-shot version of approaches in [44].
(2) **QF-GPT-0** appends the facet to the front of the query and generates the clarifying question.
(3) **Prompt-based GPT-0** is a prompt-based GPT-2 approach which includes a special instructional prompt as input:
q "Ask a question that contains words in the list [f]."
(4) **Template-0** is a template-guided approach using GPT-2. As mentioned earlier, a common problem for zero-shot GPT-2 is that it mostly generates narratives instead of questions. The Template baseline add the eight question templates during decoding and generate the rest of the question, which is similar to approaches in [13, 47].

RQ2. How effective is facet information for clarifying question generation if utilized efficiently?

To the best of our knowledge, no previous work has explored clarifying question generation using the ambiguous query as the only source of information. Previous works such as [44, 50, 54] propose various facet-specific clarifying question generation methods using facet or aspect of the query. Despite this, most of them did not or failed to experimentally demonstrate the importance of additional information for facet-specific clarifying question generation. Particularly in [44], it is shown that adding facet does not significantly improve the quality of generated questions. These works leave the effectiveness of facet in doubt. We argue that the way facet information is utilized in these works is inefficient.

To answer this research question, we compare our proposed zero-shot facet-constrained approach with a facet-free variation that uses subject words from the query as constraints. For example, the subject words of the query "I am looking for information about South Africa." is "South Africa". Using a part-of-speech tagger, we extract the nouns or proper nouns as subject from the query.

RQ3. How does our zero-shot facet-constrained approach compare to existing facet-driven baselines?

To answer this research question, we include some existing methods and a few other reasonable solutions not mentioned by previous works as our baseline models. Some of them are zero-shot, while others are not. However, we still compare their performances jointly to demonstrate our zero-shot approach's power. We divided the dataset into training and evaluating sets. All the finetuning methods can access the training set to finetune pre-trained GPT-2 checkpoint, while our zero-shot system cannot access them. Then, we evaluate all the methods on the evaluation set.

We compare our model against the following baseline models:

(1) **Template-facet** is a clarifying question rewriting baseline which appends the facet word right after the question template. For a fair comparison, we also apply multiform question templates and ranking. For example, given query q and facet f, we first generate eight questions by appending facet to each of the eight templates in Table 1. Then we rank these questions by their language perplexity. This baseline is not ideal. Admittedly, it can generate good questions such as:
q: "I am looking for information about South Africa."
f: "population"
cq: "Are you interested in [population]"
However, sometimes the facet itself is not meaningful:
q: "I am interested in poker tournaments."
f: "online"
cq: "Are you interested in [online]"
(2) **QF-GPT** [44] is a GPT-2 finetuning version of **QF-GPT-0**. It initializes with pretrained GPT-2 and finetunes on a set of (facet f, query q, clarifying question cq) tuples in the form as f [SEP] q [BOS] cq [EOS] paragraphs, where [SEP] is the separator token, [BOS] the beginning-of-sentence token, and [EOS] the end-of-sentence token.
(3) **Prompt-based finetuned GPT** is a finetuning version of **Prompt-based GPT-0** The motivation is that simple facet-as-input finetuning is highly inefficient in informing the decoder to generate facet-related questions by observing a facet coverage rate of only 20%. Inspired by recent advances in prompt studies, especially for natural language generation such as [23, 27, 42], we add a sentence "Ask a question that contains words in the list [f]" between q and cq, aiming to instruct GPT-2 the inclusion of facet words in the clarifying question. Hence, we finetune GPT-2 with the structure:
q "Ask a question that contains words in the list [f]." cq

4.2 Dataset

We use ClariQ-FKw [44] dataset for our main experiments. ClariQ dataset is originally from ConvAI3 challenge [1]. This dataset has rows of (q, f, cq) tuples, where q is an open-domain search query, f is a search facet, and cq is a human-generated clarifying question regarding the facet. The facet in ClariQ is in the form of a faceted search query. ClariQ-FKw extracts the keyword of the faceted query as its facet column and samples a dataset with 1756 training examples and 425 evaluation examples. We report the performances of all of our proposed and baseline systems on its evaluation set. Because we aim to solve the problem in a zero-shot setting, our proposed system does not access the training set. The other supervised learning systems can access the training set for finetuning.

4.3 Evaluation

We use automatic metrics for natural language generation and human annotators to evaluate system performances to label the generated questions. Following previous works [1, 2, 44], we use the human generated question from the dataset as gold reference. It is worth to clarify that this type of evaluation and its metrics are only meant to measure the the ability of a system to ask clarifying questions about the facet specifically, instead of generally relevant questions to query. However, defining other types of evaluation will be challenging given what we have in the dataset.

4.3.1 Automatic Metrics The automatic metrics we use are BLEU [35], ROUGE [25], METEOR [5], and Coverage [24]. BLEU and ROUGE are based on word match, while METEOR uses more general word forms to compute alignments between reference and generation. Coverage is computed as the average frequency of facet words in generations. Because these automatic metrics are mostly based on word overlap, we propose two different ways of computing them. We argue that not all the words in the generation are equally important. Such as the words in the question templates are less important than those in the actual question body. For example:

> q: "I am looking for information about South Africa."
> f: "picture"
> *ref*: "would you like to [see some pictures of South Africa]"
> *cq1*: "would you like to [take **pictures of**]"
> *cq2*: "are you looking for [**pictures of South Africa**]"

In this example, *ref* is the gold reference clarifying question, *cq1* and *cq2* are two candidate generations. Underline means word overlap between candidates and references. *cq1* has more word overlaps than *cq2*, including a 4-gram overlap "would you like to" with the reference. However, we humans can quickly tell that *cq1* is a worse clarifying question than *cq2*. The reason for this discrepancy is that the question template does not contain real information. Therefore it can easily bias generations with the same template but a bad question body. We are concerned that the conventional evaluation approaches computed on full questions could have limitations in our task. Plus the fact that the question templates are not actually 'generated' by the models. (They are given.) Hence, we propose another way to compute these metrics without the question template and on the actual question body. In the example, we bold the word overlap between candidates and reference within the question body. *cq2* has more overlap than *cq1* in this way, which corresponds to the human judgment of good and bad clarifying questions.

After reading Section 3.1 and 4.3, a reasonable concern would be: wouldn't the facet-constrained generation naturally improve the automatic metrics? Because using facets as constraints will make these true-positive words more likely to be included. We are aware of this concern, and we design multiple experiments and evaluations to ensure the performances of our system are meaningful. First, in RQ3, we compare our system with the Template-facet rewriting baseline, which benefits even more than our system because of the guaranteed facet inclusion. We will show that our proposed system can achieve even higher scores than Template-facet in Section 5. Second, we include human evaluations. Human annotators are free of this bias because they will evaluate generated questions by the quality of entire sentences, not word overlaps. Last, we want to highlight that facet is used in one way or another by all facet-driven models in RQ3. Using them as constraints or inputs is a modeling choice that does not break fair comparison principles.

4.3.2 Human Evaluation Metrics Like the example above, automatic metrics are reported [6] for not necessarily corresponding to true generation quality. Therefore, following previous works [40, 44, 54], we employ human annotators to evaluate the generated clarifying question qualities on 425 test examples. The annotators are provided randomly shuffled generations from all the models in RQ3 and asked to label them without knowing their sources. We provide the annotators with a detailed guideline to annotate the generated question into two labels: usefulness and naturalness. For each label, the annotators must decide whether the question is good, fair, or bad. The guideline can be found in appendix A.

Naturalness is defined as the general fluency and understandability of the generated question. The naturalness of a question is independent of its coherence to the topic of the query. By our definition, this label mainly evaluates the overall language modeling capacities of the model. Generally speaking, a zero-shot GPT-2-based decoder would keep the same language capacity as the original GPT-2 because it uses the same model. However, finetuning could downgrade the capacity due to the bias of the limited-sized finetuning set.

Usefulness is defined as whether the question is relevant to both the query and the facet and makes the query easier to answer. Typical bad usefulness questions can fall into one of the categories: duplicate, prequel, miss-intent, too general, or too specific [40]. These questions are relevant to the query but not useful for clarification. For example, for the query "Tell me about computer programming." and facet "courses", "are you interested in computer programming." is a duplicate question with the original query, and "are you looking for computer programming courses for children." is too-specific.

4.4 Implementation Details

We use NeuroLogic Decoding algorithm from the author's GitHub implementation[2]. We implement QF-GPT by ourselves with pre-trained GPT-2 checkpoints from Huggingface and achieve similar performances as the original work [44]. Similarly, we implement prompt finetuning by changing the input of QF-GPT.

The perplexity ranker is implemented using the Huggingface pre-trained GPT-2 checkpoint for our question candidate rankers.

[2]https://github.com/GXimingLu/neurologic_decoding