Comparison of preprocessing strategies of Text-to-Text Transfer Transformer for Question Generation

Hong Hai Michal Pham Sy

MSc in Data Science
The University of Bath
2021/2022

Comparison of preprocessing strategies of Text-to-Text Transfer Transformer for Question Generation

Submitted by: Hong Hai Michal Pham Sy

Copyright

Declaration

This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of MSC Data Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Abstract

Smart automatic question generation (AQG) reduces the strain on teachers and tutors to reproduce assessments on the annual basis as a part of intelligent tutoring systems (ITS). Traditionally, AQG suffered from scalability issues for rule-based systems or could not contextualize well as in the case of recurrent neural question generation (NQG). However, the recent advancements in deep learning in the form of pre-trained transformers can produce solid AQG frameworks on large knowledge domains.

This document undertakes the optimization of the state-of-the-art Text-to-Text Transfer Transformer (T5) to generate robust answer-aware questions for ITS purposes. The research looks into performances of prepending, highlight and prepending formatting with sentence-extraction encodings for varying learning rates. The evaluation includes standard n-gram metrics, semantic analysis, implementation of a question paraphraser and a question-answering (QA) model to further classify generated questions.

As all models were generating only one question per paragraph, they scored quite poorly in the evaluation with n-gram metrics. The paraphraser improved the n-gram performance of all models indicating that generated questions were similar to the rephrased references. The models proved to be robust with semantically generated questions that were contextually correct based on the findings from the QA framework.

Key Words: Automatic Question Generation (AQG), Intelligent Tutoring Systems (ITS), Text-to-Text Transfer Transformer (T5)

Contents

List of Abbreviations

AI	Artificial Intelligence
AQG	Automatic Question Generation
BERT	Bidirectional Encoder Representations from Transformers
BERTScore	Bidirectional Encoder Representations from Transformers Score
BLEU	Bilingual Evaluation Understudy
C4	Colossal Clean Crawled Corpus
FIBQs	Fill-in-blank Questions
ITS	Intelligent Tutoring System
LCS	Longest Common Sequence
LSTM	Long Short-Term Memory
MCQs	Multiple Choice Questions
METEOR	Metric for Evaluation of Translation with Explicit Ordering
ML	Machine Learning
MLM	Masked Language Model
NER	Named Entity Recognition
NLG	Natural Language Generation
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
NQG	Neural Question Generation
NSP	Next Sentence Prediction
POS	Parts of Speech
QA	Question Answering
QG	Question Generation
QQP	Quora Question Pairs
RL	Reinforcement Learning
RNN	Recurrent Neural Network
ROGUE	Recall-Oriented Understudy for Gisting Evaluation
Seq2Seq	Sequence-to-sequence network
SOTA	State of the Art
SQuAD	Stanford Question Answering Dataset
SRPE	Scalar Relative Positional Embedding
T/FQs	True or False Questions
T5	Text-to-Text Transfer Transformer
VQs	Visual Questions
WHQs	Constituent Questions
WSD	Word Sense Disambiguation

List of Figures

List of Tables

Acknowledgements

I would like to express my sincerest gratitude to Professor Ekaterina Kochmar for her guidance and constant encouragement throughout this project. I was fortunate enough to have her as my supervisor, not only because of her exceptional knowledge and expertise of the topic of this dissertation, but also because of her indulgence, patience and above all professionalism.

Her experience and depth of knowledge provided me with invaluable insight and assisted me in completing my research.

Chapter 1 – Introduction

1.1 Problem Description and Background

Written and online assessments are the core of examination processes for most educational institutions. The students are required to undertake exams every year with an ever-changing curriculum, which puts a significant strain on tutors and teachers who are responsible for producing and generating the assessments on the annual basis. To tackle this issue, the concept of intelligent tutoring systems (ITS) is being explored to provide both feedback and customized instruction to students without the involvement of a human teacher, thus reducing any existing time pressure on teachers and running costs of developing assessments.

Intelligent tutoring systems can create a substantial quantity of educational resources through the implementation of a large-scale and multi-domain knowledge base. And while the majority of ITS' bases are still being manually developed, an increasingly heavier emphasis is put to make this process fully automated. This will ensure that not only assessments are not manually created, but also their level of difficulty is not manually determined. (Tamura, Takase, Hayashi and Nakano, 2015)

Several research works have been undertaken to tackle the scope of generating assessments via a multitude of smart or automatic methods on different dataset bases. In recent years, the concepts of machine learning, deep learning and natural language processing (NLP) are being vividly explored by data scientists to solve this problem. These frameworks are applied to create automatic assessments with a varying range of success in both grammatical correctness and educational appropriateness, as a part of ITS.

The technological advance and availability of digital data sources resulted in the exploration of automatic question generation (AQG) as the potential alternative to manual question generation (QG) for ITS.

1.2 Motivation

Most of the early works focus on systems based on heuristic rules to produce smart question generation systems, these works often suffer from the lack of scalability. The implementation of deep and machine learning frameworks would not only allow new question generation models to be highly scalable but also be transferable across multiple knowledge domains. Nevertheless, automated models will continue to struggle with generating questions due to the complicated nature of human language. The optimal implementation of machine learning models would potentially allow question generation to be universally used for educational purposes in the future with questions resembling as if they were produced by teachers or tutors.

The following dissertation will specifically assess the recent application of the state-of-art text-to-text transfer transformer (T5) based on varying context-answer pairing pre-processing methods and hyper-parameters setups to develop an understanding of the deployment of robust and optimal AQG from text paragraphs. Thereby, a base T5 with different data processing and hyper-parameters

are tested and juxtaposed against each other to find the most advantageous approach to generate contextually sound and grammatically appropriate questions.

1.3 Thesis Structure

The remaining part of the thesis is structured as follows. Section 2 outlines question generation as a natural language processing task and reviews various question generation models highlighting their mechanism principles, advantages, and disadvantages. Section 3 extensively discusses the methodology frameworks and evaluation strategies. Section 4 presents the results and discusses the outcomes of the experiments. Lastly, Section 5 provides a conclusion to the thesis.

Chapter 2 - Literature Review and Technology Survey

This section provides an extensive literature review and technology survey of the various question generation models. The question generation mechanisms, architecture, advantages, and disadvantages of the question generation method are thoroughly discussed. Lastly, the approach of this thesis to generate questions based on the reviewed literature is underlined.

2.1 Question generation task and types of questions

The task of automatically generating a grammatically and syntactically correct interrogative sentence based on some context is known as question generation (Lamba and Hsu, 2021). Some common applications of question generation include human-computer interaction (Mostafazadeh et al., 2016), education (Heilman et al., 2010) and question answering (Wang, Yuan and Trischler, 2017). Aside from these applications, question generation can help in the creation of data sets for question-answering frameworks. That is why in recent years, both the industrial and academic NLP communities have shown a keen interest in question generation due to enormous prospective benefits in a variety of fields (Zhou et al., 2017). Initially, QG tasks were derived from unstructured data sources like sentences and texts (Du et al., 2017; Song et al., 2018). However, researchers have recently broadened the range of sources by incorporating knowledge bases (Bao et al., 2018), triples and images (Li et al., 2018), where questions remain answerable from the initial inputs.

Natural Language Processing (NLP) is a field of artificial intelligence (AI) that facilitates human-computer interaction in a language of the choice with QG task being part of it. NLP engineers attempt to build machine learning systems that are capable of emulating questions as if they were created by humans. Questions can be built in a variety of ways and while most of them are similar in structure and could be easily formulated by humans, machine models' approaches may differ or may require additional steps to generate such questions correctly. The type of questions include:

- **Visual Questions (VQs)** – Questions generated based on given images, connecting both fields of NLP and computer vision. These types of questions could serve the purpose in educational demonstrations or conversations with chatbots (Xie et al., 2022). To generate VQs, different neural models are simultaneously used to analyse visual features and corresponding image captions to generate questions (Zhang et al., 2017). Extra validation of generated questions may take the form of comparison between key terms extracted from a caption and predicted answers to generated questions (Changpinyo et al., 2022). The Visual QG example framework can be seen in Figure 1.

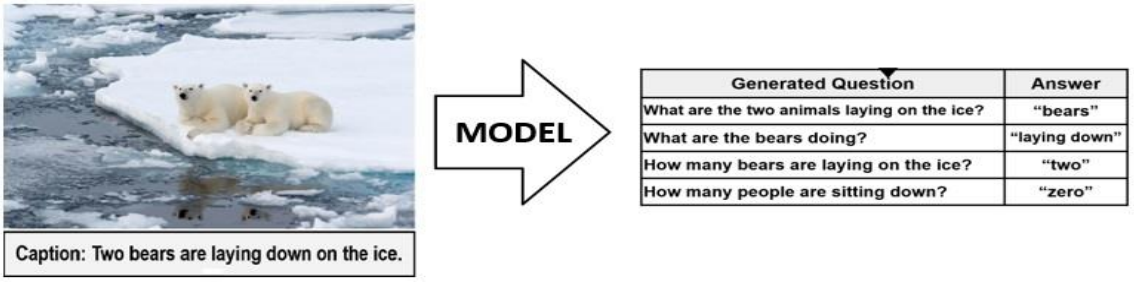

Figure 1: Visual Question Generation example (Changpinyo et al., 2022).

- **Fill-in-the-blank Questions (FIBQs)** – The question or statement is formed through a process of gap selection, where the most adequate word is removed. Rather than using NLP to form semantically and grammatically correct questions, an algorithm for word removal selection could be simply utilized for this NLG task (Brown, Frishkoff and Eskenazi, 2005).

- **Boolean or True-False Questions (T/FQs)** – Similarly to FIBQ types, T/F questions could take the form of statements or questions depending on the modelling framework. The T/F statements could be generated in a multitude of ways by changing entities, adjectives, and verbs or adding negations in the ground truth sentence.

- **WH- or Constituent Questions (WH-Qs)** – Open-type questions that require an agent to understand the context both grammatically and semantically. NLP pre-processing techniques like sentence extraction or comprehension are deemed to be suitable. Novel approaches such as neural QG have proven to achieve significant results in QG of WH-Qs (Duan et al., 2017).

- **Multiple Choice Questions (MCQs)** – Generated on the same basis as constituent questions with an additional generation of distractors (wrong answers) to complement a correct answer. The process of generating distractors would require a prior grammatical and semantical analysis of the answer based on the context or sentence to avoid any word sense disambiguation (WSD), followed by a distractor generation of synonyms or antonyms (Gao et al., 2019).

This document will primarily focus on the QG of constituent questions from the text paragraphs given the answer. The model will also consider T/F questions in the situation where an answer 'true' or 'false' is explicitly mentioned within the context of the paragraph.

2.2 Conventional question generation models

The processing of language is a complex task that requires a model to be familiar with the laws and ambiguity of human languages. Thus, making question generation (QG) tasks being often considered to be very challenging for computational systems as any kind of content generation would require a heavy understanding of natural language intricacies as well as knowledge of the world.

The first approaches to generating questions computationally were centered around rule-based techniques and template-based models. These methods were mainly revolving around the identification of domain-specific words, which was achieved via recognizing recurrent nouns in the text and then modelling them accordingly to pre-defined rules to output questions. Existing rule-based systems can be divided into three distinct categories i.e. semantic-based (Huang and He, 2016), template-based (Mostow and Chen, 2009) and syntax-based models (Kunichika et al., 2001).

- **Template-based:** The template-based approach generates questions via a pre-defined structure of the template that consists of fixed question structuring, a set of requirements and blank spaces. The template-based models are taught to detect common patterns through empirical (Sadigh, Seshia and Gupta, 2012) or computational approaches. They

create questions by filling in blanks with input sources when input sources match the template requirements. In practice, the questions generated from a template model are highly parametrized and individualized allowing the same type of question to be rephrased with new input parameters resulting in a large pool of questions (Le et al., 2014). Table 1 displays how the template-based model works in practice.

Type	Question Template	Question
Definition	What is the definition of <input>?	What is the definition of **solar energy**?
Features	What are the features of <input>? What are the issues with <input>?	What are the features of **solar energy**? What are the issues with **solar energy**?
Example	What are examples of <input>? What are the applications of <input>?	What are examples of **solar energy**? What are the applications of **solar energy?**

Table 1: Template-based question generation mechanisms.

The more advanced type of templates can be utilized for specialized relationships between entities i.e. 'What occurs between <input_1> and <input_2>?'. However, these templates are more suitable for applications with specific purposes within a closed domain, and therefore new question templates will be required, when other types of entity relationships are considered (Le et al., 2014).

- **Syntax-based:** As the name indicates, these models only utilize syntactic (following rules of the language) input information to generate questions. Often a part of speech (POS) is utilized to aid these QG systems. The POS tagging is a feature classification of tokens that indicates each grammatical function of the word within the sentence (Kim et al., 2017). The part of speech tagging mechanism is displayed in Figure 2.

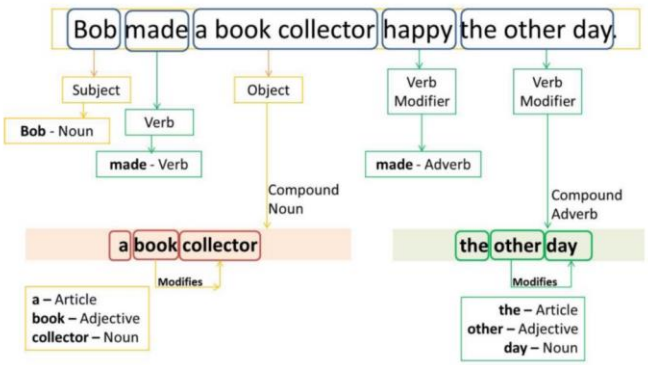

Figure 2: The Part of Speech tagging example (Godayal, 2018)

This process remains challenging due to the ambiguous nature of language, where the same words could be verbs or nouns given not only in the context of the sentence but also in the tense of the phrase. The syntactic system then uses this data to determine the relevancy of the input words to apply syntactic transformations to generate a question.

- **Semantic-based:** More commonly used than syntax-based models. The semantic-based model closely maps an approach proposed by syntactic systems but enriches the input

manually or automatically with additional semantic resources. Semantic models attempt to define semantic entities, roles, relations, and actions within the statement as these describe the subject, its role, what it does in what way and when. Tools such as Named entity recognition (NER) are widely used to predicate semantic roles by looking to define entities such as people, organizations, and locations through information extraction. By fully understanding the functionality of semantic relations, the questions can be formulated in a way that will be more relevant to the overall context (Fattoh, 2014). Table 2 depicts entity recognition functionality.

Entity Name	Entity Type	Sample question types
UK	Location	Where?
Google	Organization	When?
Bill Gates	Person	Who/Whom?

Table 2: Sample Entity Recognition (Fattoh, 2014)

The performance of these models would heavily depend on both the number of rules and their quality. On the other hand, these systems have some advantages in terms of enhanced model interpretability and giving developers more autonomy and control over the model (Lamba and Hsu, 2021). Another variation of the rule-based approach is an overgenerate-and-rank model, which follows the same rules, but can generate multiple questions from a sentence and then provides question grades through a supervised learning ranker (Huang and He, 2016). Although the results are more accurate than a simple rule-based approach, the questions are frequently too obvious and give provide answers in their wording (Grover et al., 2021).

Due to the restricted nature of rule-based and template-based approaches, these QG models often suffered from the scalability and sub-par generalizability of adequate QG, when reliant on human hand-crafted templates or heuristic methods (Mostow & Chen, 2009; Heilman & Smith, 2010). That is why template and rule-based systems would not be suitable to create large and transferable knowledge domains from QG tasks.

To address these issues, a heavier emphasis is put on applications that do not require manually, pre-defined rules and are end-to-end trainable. The adoption of methods to generate knowledge models that do not inherit biases from pre-established rules will allow for broader applicability and transferability to different knowledge domains (Chen, Wu and Zaki, 2022).

2.3 Neural Question Generation (NQG)

2.3.1 Answer-aware versus answer-unaware modelling

Deep neural networks are used to generate questions from a given passage in a process known as neural question generation (NQG) (Kim et al., 2019). The NQG is split into two categories of answer-aware and answer-unaware model types (Cao, Tatinati and Khong, 2020). In answer-aware models, both context and corresponding answer are supplied to generate a question, whereas answer-unaware models are only given the context.

Without a pre-defined selection of answers, answer-unaware models are often required to decide which part of the text is relevant enough to form a question about. In 'Neural models for key phase extraction and question generation' (Subramanian et al., 2018), Subramanian via answer-extraction modelling can generate contextually sound and fluent questions but fails on focusing on relevant, contextual answer takeaways from text passages. Similar follow-up studies by (Willis et al., 2019; Cui et al., 2021; Du and Cardie, 2018) come short to assess whether extracted answers and the corresponding generated questions were relevant to the context passage's broader topic.

On the other hand, Wang et al., 2018 have undertaken a feasibility study of answer-aware NQG for educational purposes. The neural model was trained on the SQuAD dataset consisting of context-answer-ground truth inputs and evaluated on various knowledge domains including but not limited to biology and history.

The results have shown that questions generated from the answer-aware model were both grammatically correct and relevant to the context of the passage. Additionally, the generated questions have shown a high n-gram overlap with human-authored questions (Dugan et al., 2022)).

As of now, SOTA systems are answer-aware NQG as these vastly outperform answer-unaware and non-NQG approaches (Dong et al., 2019). Answer-aware models tend to generate better contextually and syntactically sound questions and therefore will be more suitable for QG tasks in the context of intelligent tutoring systems.

2.3.2 Recurrent Neural Networks

With the invention of Recurrent Neural Networks (RNNs), text analysis and processing became more approachable through deep learning methods. Recurrent neural networks (RNNs) are a type of neural network that excels at processing time series and other sequential data. Neural QG offers an end-to-end framework and data-driven approach, where text paragraphs and question generators could be optimized together (Xie et al., 2022). In general, neural models tend to promote greater question diversity and fluency compared to rule-based models. The RNNs perform the same function on various data inputs until results are achieved, at every timestep the RNN takes the current input and the output at the previous timestep to obtain subsequent outputs, hence the feed-forward mechanism. The RNN string processing is shown in Figure 3.

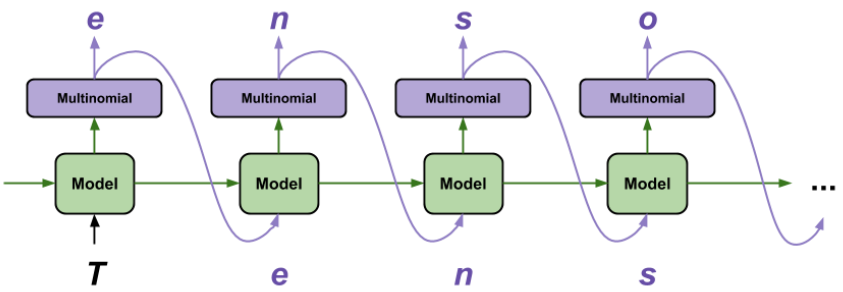

Figure 3: RNN model mechanism (TensorFlow, 2022)

The RNNs became widely used in NLG frameworks as their feed-forward mechanism aids in the processing of various kinds of data. RNNs were capable of processing sequences (Schmidhuber, 2015), which is a crucial feature in the natural language generation (NLG) tasks.

However, while in theory, RNNs could analyze long strings of texts, in practice they would still produce unsatisfactory results due to exploding problems and gradient vanishing when its weights within the neural network are being updated by too much or too little, respectively. To address these issues Long Short-Tern Memory (LSTM) artificial neural network was designed capable of handling long-term contextual dependencies due to the implementation of gated units within the network architecture (Grover et al., 2021). The LSTM model was able to generate contextually sound data better compared to conventional RNNs. The downside of RNN question generation models was their inability to process context paragraphs and questions without forming a prior relationship.

To tackle this problem, the sequence transformation model turning one sequence into another sequence (Seq2seq) was proposed by Du, Shao and Cardie (2017) in their work 'Learning to ask: Neural question generation for reading comprehension. Unlike, the previous models it incorporates a separate encoder and decoder within its architecture and can interpret question generation tasks as sequence-to-sequence problems The encoder of the Seq2Seq model encodes the reference input text into one or multiple fixed-size vectors, while the decoder decodes the target text by creating a hidden vector based on the workings of the encoder. The data-driven approach made the model independent of any manual rules' inputs. All the information about the context is stored within the decoder's hidden vector (Du, Shao and Cardie, 2017)..

The Seq2seq model has vastly outperformed any rule-based and overgenerate-and-rank systems in the context of QG (Grover et al., 2021). Nevertheless, their Seq2Seq model had no control on the part of the context that the corresponding generated question was asking about (Du, Shao and Cardie, 2017).

Further improvements to Seq2Seq models included the implementation of reinforcement learning (RL) algorithms as a reward function associated with adequate question generation. Various reward functions were experimented on to boost the performance of Seq2Seq models.

RNNs, LSTMs and Seq2seq models have become state-of-the-art (SOTA) methods in transduction problems and sequence modelling, which include machine translation and language modelling tasks like QG (Bahdanau, Cho and Bengio, 2014). However, because of the nature of how these neural networks operate at a time step, it limits the abilities of these models to analyze longer sequence lengths due to the lack of parallelization, which directly may result in memory constraint issues. And while the recent works introduce serious improvements in machine efficiency through conditional computation and factorization tricks, the fundamental problem of sequential computation remains in place (Kuchaiev and Ginsburg, 2017).

2.4 Transformers

The introduction of a transformer by (Vaswani et al., 2017) turned out to be a breakthrough advancement in the domain of Seq2Seq modelling. Similarly, to the Seq2Seq, its model architecture

revolves around an encoder-decoder concept with N – numbers of the same layers stacked on top of each other. Figure 4 displays the model architecture of the transformer (Vaswani et al., 2017).

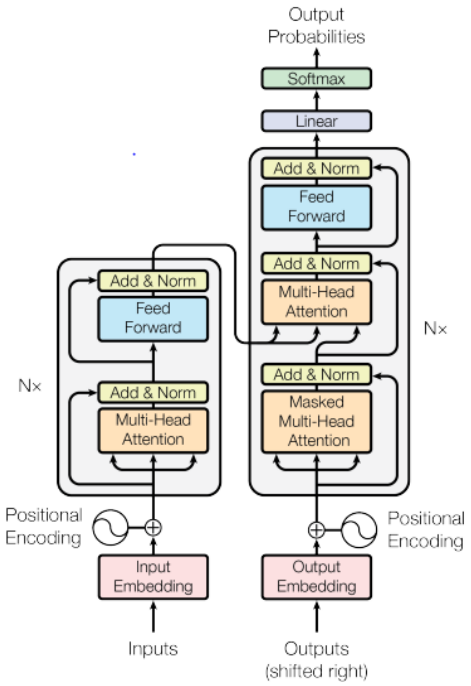

Figure 4: The model architecture of the transformer (Vaswani et al., 2017).

Encoder:
The encoder's structure consists of a stack of 6 identical layers with each layer having two sub-layers. The first sub-layer is a multi-head self-attention mechanism and the second one is a fully connected feed-forward layer responsible for performing linear transformations of inputs. The multi-head self-attention works similarly to the regular attention mechanism, but projects simultaneously queries, keys and values with respective linear projections over dqueries, dkeys and dvalues dimensions. Each two sub-layers are connected by residual connections, followed by a layer of normalization. The product of each sub-layer is accompanied by additive normalization, where both output and input layers are summed together and normalized (Vaswani et al., 2017).

Decoder:
The structure of the decoder is almost the same as the encoder's one. The difference is that the decoder inserts a third sub-layer that runs multi-head attention over the encoder stack's output. The self-attention sub-layer is also modified, preventing positions from attending subsequent positions. As a result, output embedding will be offset by one position, essentially ensuring that predictions for position i can depend solely on the familiar outputs at positions less than i (Vaswani et al., 2017).

2.4.1 Attention mechanism
The role of attention is to empower the model by learning to focus on the relevant parts of the input text. Via attention, the NLP model creates relationships among different words in the same sentence. It artificially creates a vast range of references, overcoming the short-term memory weaknesses of conventional RNNs, and enabling the model to understand the whole context of the input text. Attention mechanism has become a key part of transduction and sequence models in

various frameworks, allowing analyzing model dependencies without any regard for the distance between input and output targets (Bahdanau, Cho and Bengio, 2014; Kim et al., 2017).

The attention mechanism has proven to be effective in almost all NLG and NLP tasks and paved a way for transformers. The transformers forego recurrence in favour of drawing global dependencies between input and output positions by using the attention mechanism (Vaswani et al., 2017).

The transformers proposed by (Vaswani et al., 2017) outperform the best previously made models in the machine translation tasks from English to German at significantly less training costs equating to 25% of the previous SOTA model. The transformers have proven to achieve significantly more accurate results compared to the previous method when only trained for a relatively small period.

One of the major limitations of the first transformers was their decoder's unidirectionality, where every position or token can be only attended by previous ones in the self-attention attention layers. Such constraint is sub-optimal for sentence-level tasks and could be proven detrimental for fine-tuning token-level methods like QG or question-answering (QA) tasks, where it is crucial to understand the context from both directions.

Nevertheless, the vast benefit of using them over the conventional QG methods and recurrent NQG is their ability to operate with parallelization allowing transformers to be pre-trained on extensive knowledge domains and datasets (Vaswani et al., 2017).

2.5 Pre-trained systems

The pre-trained model systems have become more and more popular for NLG tasks as a data-rich pre-training process that allows them to develop generally applicable abilities and all-purpose knowledge that can be transferred to downstream tasks (Raffel et al., 2020).

2.5.1 Bidirectional Encoder Representations from Transformers (BERT)

The Bidirectional Encoder Representations from Transformers (BERT) by Devlin et al (2019) has been proposed to address the issues related to the unidirectionality of initial transformer types proposed by (Vaswani et al., 2017). Unlike other proposals by (Radford and Narasimhan, 2018) that focus on unidirectionality during the pre-training process and (Peters et al., 2018) that implement shallow concatenation of independent left-to-right and right-to-left language models to model the context, Devlin introduces a masked language model (MLM). The MLM masks some of the tokens in the input at random, to predict the original vocabulary ID of the masked word based solely on its context (Devlin et al., 2019).

Another, significant feature complementing the masking language was the implementation of next sentence prediction (NSP) by Devlin et al. 2019. For BERT to understand the context at the sentence level, it was necessary to implement an algorithm to comprehend the relationships between each sentence within a paragraph. Thus, the model was trained to predict sentence sequencing, where sentence A is either followed by a sentence B or a random phrase from the corpus. This additional pre-training step has proven to be beneficial for NLG tasks, and unlike prior works (Jernite,

Bowman and Sontag, 2017; Logeswaran and Lee, 2018), BERT could store all parameters like token, segment and position embeddings to downstream tasks as opposed to only word embeddings (Raffel et al., 2020). Lastly, similar structurally-wise to the transformer of Vaswani et al., 2017, BERT incorporates bidirectional self-attention for context to be interpretable in both ways.

BERT was pre-trained extensively on document-level English Wikipedia (2,500 million words) and BooksCorpus (800 million words) and became SOTA for eleven different NLP tasks at the time proving to be a breakthrough and improvement over the other models (Devlin et al., 2019).

Since its implementation, BERT has also been used for QG tasks. Chan et al. (2019) propose three methods of generating answer-aware questions based on various sequence input representations for BERT models. The BERT's regular sequence input X for QG consists of context-answer pairing separated by different types of tokens. Chan et al. 2019 use three different tokens, [CLS] token inserted as the first token within a sentence, [SEP] token separating consecutive sentences and [HL] token highlighting an answer within a text.

Data pre-processing could be divided as follows (Chan and Fan, 2019):

- **BERT-QG – Prepending Format** – The input sequence X for a given context paragraph C and answer phase A is shown below.

  X = ([CLS], Context, [SEP], Answer [SEP])

- **BERT-SQG – Masking Format** – Alternative method, addressing problems of prepending format related to the lack of previously decoded result information during token generation. Additional consideration information is supplied within the input sequence. The input sequence is as follows:

  X = ([CLS], Context, [SEP], Answer [SEP], Decoder Results [MASK])

  The decoder results and a masking token are following the answer phase. The model works iteratively, each time predicting a generated token until convergence is achieved – a question is formed.

- **BERT-HLSQG – Highlighting Format** – The last formatting method avoids issues related to processing extended context resulting in subpar question quality and reduces ambiguity in the situation where an answer occurs multiple times in the text. The proposed BERT-HLQSG's input sequence highlights the answer with special [HL] tokens within the paragraph.

  X = ([Context left of answer [HL] Answer [HL] Context right of answer])

Chan et al. 2019 have proven that sequential structuring of input is important for decoding text generation. The models were able to generate comprehensive questions with BERT-HLSQG achieving outstanding results and outperforming all other BERT variations and NQG models. BERT-SQG had a very similar performance as BERT-HLSQG with BERT-QG performing the worst out of the three.

2.6 Text-to-Text Transfer Transformer (T5)

One of the takeaways from BERT is the fact that it outputs labels or spans of the input to the actual input sentence. An attempt has been made to produce a completely text-to-text transformer capable of outputting text strings (Chan and Fan, 2019). With an increasing number of proposals and works on different models and pre-trained systems, it has become more difficult to compare various algorithms and understand the impacts of new contributions.

The common-ground solution was achieved by introducing a pre-trained system referred to as Text-to-Text Transfer Transformer (T5) (Raffel et al., 2020). The T5 unifies NLP frameworks including question generation, summarization, sentiment analysis, language modelling, question answering or span extraction. Most importantly, T5 as a text-to-text frame can apply the same model, goal, training steps and decoding to every required task. The structure of T5 is shown in Figure 5.

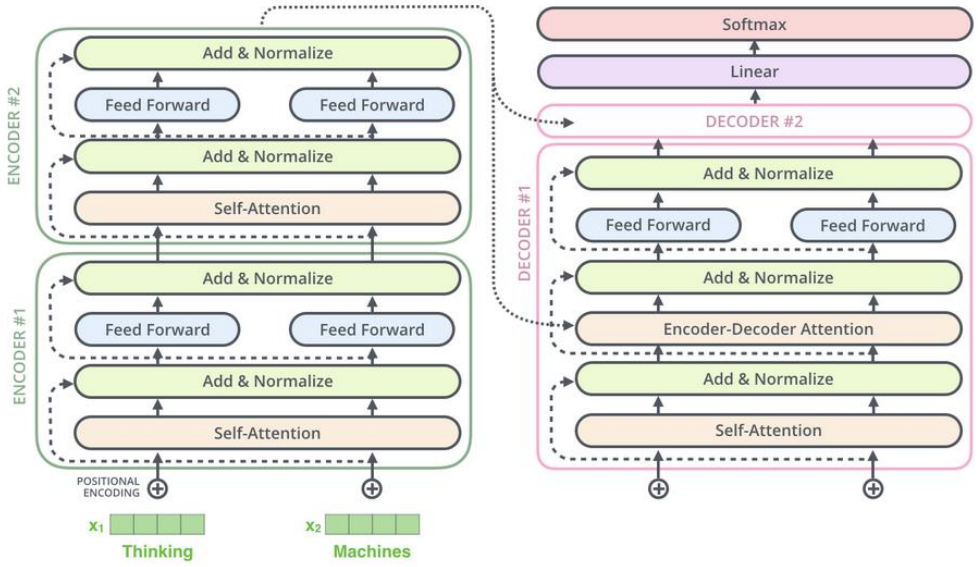

Figure 5: T5 Architecture (Alammar, 2018)

The T5 utilizes multiple encoders and decoders with an overall very similar structure to regular transformers (as shown in Figure 4), however, it additionally utilizes scalar relative positional encoding (SRPE). The SRPE allows the computation of a scalar value depending on the position of the word within a sentence to update the self-attention layers (Wu et al. 2021).

The T5 was trained on the Colossal Clean Crawled Corpus (C4), a dataset consisting of 750GB of cleaned and parsed English text taken from the web. T5-base has been tested with a clean and unfiltered version of C4 to develop an understanding of model performance when trained on various variants of the data with results indicating improved performance on the clean version.

Due to a sheer number of parameters taken into consideration while training, the T5 model has 5 size types, each with varying architecture to accommodate for parameter processing. Table 3 provides a summary of T5 model variations.

Name	Number of parameters
T5-Base (Baseline)	220 million
T5-Small	60 million
T5-Large	770 million
T-3B	2.8 billion
T-11B	11. billion

Table 3: The parameter configuration of T5 variants (Raffel et al., 2020)

The T5 performance is dependent on model size performance with scaled-up models achieving significantly better performance compared to smaller models. On the other hand, smaller models could be utilized in situations, where computational resources are scarce for fine-tuning and inference. All T5 models have been trained and compared against the existing SOTA systems at the time. Table 4 represents the performance of T5 variances on the SQuAD dataset against SOTA at that time; ALBERT (BERT variation) by (Lan et al., 2020).

Name	SQuAD EM (Exact Match)	SQuAD F1-score
SOTA ALBERT (Lan et al., 2020)	**90.10**	**95.50**
T5-Base (Baseline)	79.10	87.24
T5-Small	85.44	92.08
T5-Large	86.66	93.79
T-3B	88.53	94.95
T-11B	**91.26**	**96.22**

Table 4: Performance of T5 variants against ALBERT (Lan et al., 2020) on SQuAD

SQuAD EM and F-1 score are two different metrics used to evaluate the performance of the model on the benchmark. The SQuAD EM measures the overall percentage of candidates that exactly match any of the references, whereas SQuAD F1 computes the average overlap between the candidate and reference. T5-11B variant has outperformed a long-standing SQuAD EM benchmark by over 1 point, whereas most of the other models' improvements could better ALBERT's performance by only a fraction of a percentage point (Raffel et al., 2020). Since then both answer-aware and answer-unaware T5 frameworks have been undertaken.

Pandraju and Mahalingam (2021) used T5 which was pre-trained on SQuAD and Boolean Question dataset to generate answer-aware questions from tabular and textual data. T5 was fed with additional metadata to detect sets of highlighted cells within a table allowing more structured and precise question generation. As a result, their model achieved a BLUE-4 score of 72.66 for T5 base models. BLEU score measures the sequential word overlap between the reference and the candidate.

A different approach to generating questions was done by Zhang, Zhang and Wang (2022), who generate question-answer pairs simultaneously on varying learning-rate hyperparameters with highlight and prepending format encoding. Their pre-processing of the SQuAD dataset included the removal of unsuitable sentences and candidates with inappropriate semantic labels or POS tags with their model achieving Bilingual Evaluation Understudy (BLEU) scores of 22.62 and 23.19 for prepending and highlight encoding respectively.

2.7 Approach:

This document takes part inspiration from Chan and Fan, (2019) and Zhang, Zhang and Wang (2022), to deliver an extensive assessment of pre-processing approaches and hyper-parameter tuning impacts on answer-aware T5 ability to produce both contextually and grammatically correct questions.

Three different pre-processing methods are established during the training process of T5 on the specifically predetermined fraction of Stanford Question Answering Dataset version 1 (SQuAD). The two of these methods (prepend sequential and highlight sequential) were previously used with a different transformer – BERT proposed by Chan and Fan (2019). The novel approach proposed is a hybrid of two methods implementing a prepending format for answers and sentence extraction modelling to highlight the sentence, which contains an answer. Each method is hyper-tuned on six different learning rates to find an optimal T5 performance setup as performed by Zhang, Zhang and Wang (2022). Due to limited computational resources, the tested T5 variance will be T5-Base.

Afterwards, several NLP n-gram metrics are applied to sets of generated questions by each model to assess their ability to generate sound questions. Furthermore, semantically sensitive metrics like BERTScore will be considered as well to provide a deeper assessment of generated questions.

The usage of question paraphraser will be accounted for on ground truth questions to generate additional correlated reference questions to draw a further conclusion on the performance of each model.

Lastly, a question-answering framework will be developed to validate whether generated questions can produce answers resembling ground truth answers provided by SQuAD.

Chapter 3. Methodology

Chapter 3 extensively discusses the methodology of the paper. Firstly, section 3.1 highlights a structure of the question generation pipeline following which all the processes are undertaken. Sections 3.2 and 3.3 describe a dataset used for this document and pre-processing strategies. The hyper-parameter tuning is discussed in section 3.4. Furthermore, post-processing to generate questions is described in section 3.5. Lastly, the evaluation strategies are scrutinized in section 3.6.

3.1 Question Generation Pipeline

The proposed QG pipeline consists of six different stages. The QG framework follows the pipeline proposed in Figure 6. The sections are organized chronologically and follow the logic of the pipeline.

Figure 6: The question generation pipeline

The following sections from 3.2 to 3.6 will discuss in detail the steps undertaken from parsing the dataset to validation and evaluation strategies required to assess the generated questions.

3.2 Dataset

The source of question-generation data is the one provided by Stanford Question Answering Dataset (SQuAD) version 1. SQuAD_v1 is a large-scale reading comprehension dataset consisting of approximately 98 thousand text paragraphs from Wikipedia, each supplemented with a complementary ground truth question and corresponding answer. Additionally, the dataset highlights individually answer locations within each paragraph. Figure 7 represents a random sample from the dataset.

Context: Elway stated, "We've had Super Bowl success in our white uniforms." As the designated home team in the annual rotation between AFC and NFC teams, the Broncos elected to wear their **road white jerseys** with **matching white** pants. The Broncos last wore matching white jerseys and pants in the Super Bowl in Super Bowl XXXIII, Elway's last game as Denver QB, when they defeated the Atlanta Falcons 34-19.

Reference Question: What jersey did the Broncos wear for Super Bowl 50?

Reference Answers: **road white jerseys**, **matching white**, **white**
Answer starts: 112, 136, 117

Figure 7: The sample data triple with highlighted reference answers

3.3 Preprocessing

The two preprocessing strategies have been inspired by Chan and Fan (2019) BERT and adjusted for the T5 environment. The last pre-processing strategy is a proposed; novel strategy and is a combination of prepending and target sentence-level highlight formatting. Unlike BERT, in T5 [CLS] tokens are not required in the input encoder. All QG pre-processing strategies are answer-aware systems, where both answer and context are used together as a text string input to train the model. The formats of train and test datasets were revised according to each of the pre-processing strategies and were later used in the post-processing for corresponding methods to generate questions.

The sub-sections 3.3.1, 3.3.2 and 3.3.3 detail different pre-processing strategies.

3.3.1 Prepending Formatting

The pre-processing strategy input consists of an answer followed by a context. The answer is separated from the context by a designated <SEP> token responsible for separating two various sentences in the same text string input.

Prepending Encoder Input = Answer + **<SEP>** + Context

Input: *road white jerseys* **<SEP>** Elway stated, "We've had Super Bowl success in our white uniforms." As the designated home team in the annual rotation between AFC and NFC teams, the Broncos elected to wear their road white jerseys with matching white pants. The Broncos last wore matching white jerseys and pants in the Super Bowl in Super Bowl XXXIII, Elway's last game as Denver QB, when they defeated the Atlanta Falcons 34–19.

Figure 8: The prepending input

3.3.2 Highlight Formatting

The pre-processing strategy focuses on highlighting an answer within the context paragraph. The answer is highlighted with special <HL> tokenindicating the answer's position within the paragraph.

Highlight Encoder Input = LHS Context + **<HL>** + Answer + **<HL>** + RHS Context

Input: Elway stated, "We've had Super Bowl success in our white uniforms." As the designated home team in the annual rotation between AFC and NFC teams, the Broncos elected to wear their **<HL>** *road white jerseys* **<HL>** with matching white pants. The Broncos last wore matching white jerseys and pants in the Super Bowl in Super Bowl XXXIII, Elway's last game as Denver QB, when they defeated the Atlanta Falcons 34–19.

Figure 9: The highlight input

3.3.3 Hybrid Formatting and sentence-level extraction preprocessing

The hybrid method incorporates elements of both prepending and highlight formatting. Similarly, the prepending formatting answer is followed by a context. However, in addition to that, the target sentence containing the answer is highlighted within the context. As the answer is already provided at the beginning of the input, there is no need to highlight it with <HL> tokens, instead, a proposal is made to highlight the sentence containing the answer.

Hybrid Input = Answer + <SEP> + LHS Context + <HL> + Target Sentence + <HL> + RHS Context

Input: *road white jerseys* <SEP> Elway stated, "We've had Super Bowl success in our white uniforms." <HL> As the designated home team in the annual rotation between AFC and NFC teams, the Broncos elected to wear their road white jerseys with matching white pants. <HL> The Broncos last wore matching white jerseys and pants in the Super Bowl in Super Bowl XXXIII, Elway's last game as Denver QB, when they defeated the Atlanta Falcons 34–19.

Figure 10: The hybrid input

For all methods, the pre-processed inputs were subsequently put as input encodings with corresponding questions used as target encodings.

3.4 Fine-tuned T5 Transformer

The model architecture of T5 is preserved as proposed by Raffel et al. (2020) and is described in detail in section 2.4. The transformers were trained on 26000 training and 5000 validation for 7 epochs with gradient accumulation, training and validation batch sizes of 10 as computational resources were scarce. The data was converted to vector features with a feature extraction function[1] and was subsequently collated with a data collator function[2]. The transformer was fine-tuned to understand the performance of each pre-processing strategy for various learning-rate ranges.

3.4.1 Hyper-parameter Tuning

The hyper-parameter optimization is based on varying learning rates with other hyper-parameters being fixed. The learning rate magnitude indicates how fast the model learns during the training process. When the learning rate is too small, the model will converge very slowly. On the other hand, the large learning rate may help with regularizing the training, but in the situation where it is too big, it may cause training divergence.

It is crucial to optimize learning to maximize the model's convergence rate resulting in adequate and correct question generation. The learning rates used by Zhang, Zhang and Wang (2022) are examined as a part of this research and displayed in Table 5.

LR1	LR2	LR3	LR4	LR5	LR6
0.0005	0.0001	0.00005	0.00003	0.000025	0.00001

Table 5. Model Learning rates

3.5 Postprocessing

Following the training of the model, the questions were generated for previously unseen paragraphs from a test dataset. The questions were generated with a beam search of size 3 in the decoder resulting in one output question generated per each testing example. The beam search strategy chooses the best output question based on the maximum probability. The length penalty was set to 1.0, resulting in no biases toward creating long or short questions. The n-gram repeat size was set

[1] https://github.com/patil-suraj/question_generation/blob/master/prepare_data.py
[2] https://github.com/patil-suraj/question_generation/blob/master/data_collator.py

to 3 ensuring that all n-grams of size 3 can only occur once. Each question was generated as a part of the answer-aware framework on the parsed dataset corresponding to pre-processing strategy undertaken during the training.

3.6 Evaluation and validation

3.6.1 Evaluation Framework

A total of 18 models will be scrutinized as a part of the evaluation framework. The structure of the evaluation framework is shown the Figure 11.

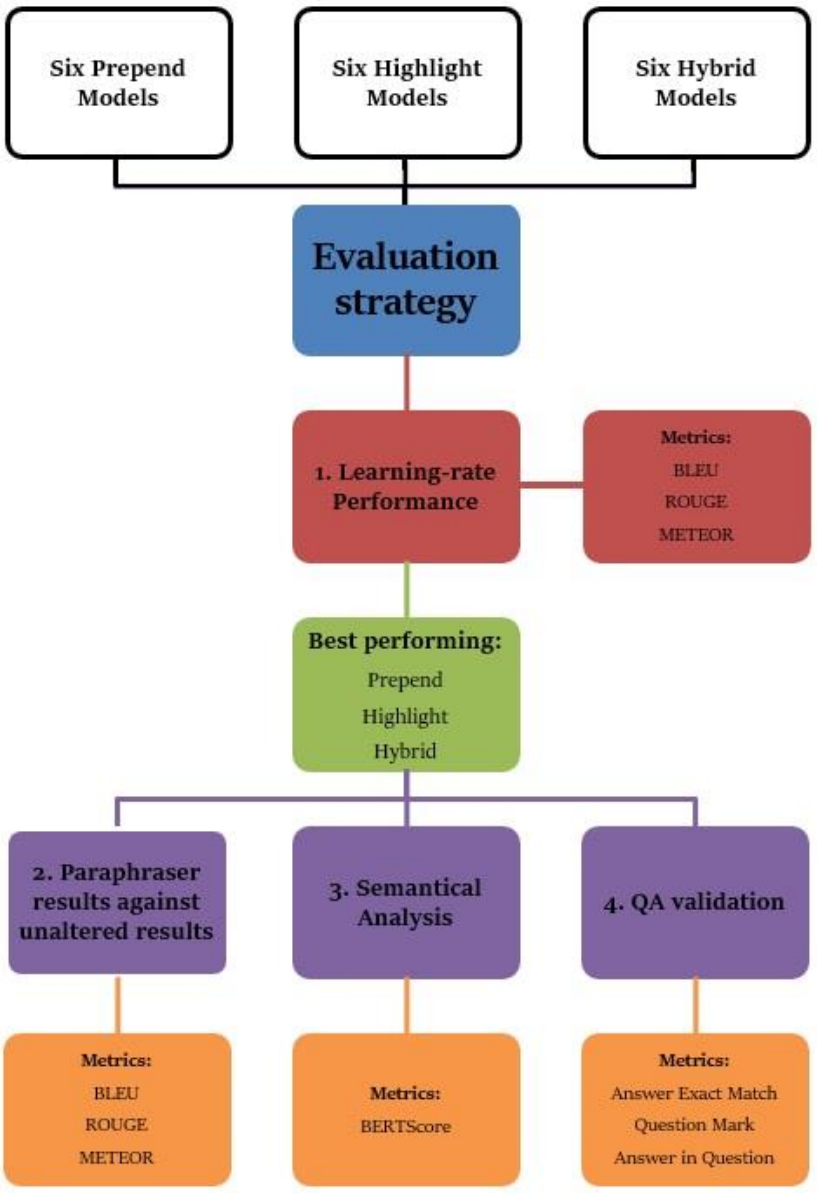

Figure 11: The evaluation framework

Firstly, the learning rate model analysis based on their performance for metrics such as Bilingual Evaluation Understudy (BLEU), Recall-Oriented Understudy for Gisting Evaluation (ROUGE) and

Metric for evaluation of translation with explicit ordering (METEOR) is undertaken.

The best prepending, highlight and hybrid models will be analyzed further. Following the first step, the paraphrased results will be compared against unaltered ones to draw a further comprehension of the performance of each model.

Furthermore, the semantical analysis will be undertaken with Bidirectional encoder representations from transformers score (BERTScore). As the last validation step, the QA model will be employed to answer all generated questions to count the instances of answers' exact matches against reference answers. As, a part of this QA model, the generated questions containing answers or lacking a question mark within their structure will be classified as invalid.

The three n-gram-based BLEU, ROGUE, and METEOR metrics will be used to assess the performance of models for learning-rate optimization and question paraphraser performance. These implementation, mechanism, advantages and disadvantages are discussed in subsections 3.6.1, 3.6.2 and 3.6.3 for BLEU, ROGUE and METEOR, respectively.

3.6.1 Bilingual Evaluation Understudy (BLEU) 1- 4

The primary objective of BLEU is to compare the n-grams (consecutive sub-string) of the candidate to the n-grams of the reference text and count the number of matches, with the score 1.0 of being a perfect match and 0 is a no match. BLEU incorporates the Brevity penalty that penalizes predicted statements that are shorter compared to ground truth statements. If a generator predicts a short sentence, the Brevity Penalty will be small resulting overall in a lower BLEU score (Papineni et al., 2002).

$$Brevity\ Penalty = \begin{cases} 1, & if\ c > r \\ e^{\left(1-\frac{r}{c}\right)}, & if\ c \leq r \end{cases} \tag{3.1}$$

- r is reference sentence length
- c is predicted sentence length

The BLEU score is computed by multiplying the Brevity Penalty with a geometric average precision of N-gram size.

$$Geometric\ Average\ Precision\ (N) = \prod_{n=1}^{N} p_n^{\frac{1}{N}} \tag{3.2}$$

$$BLEU(N) = Brevity\ Penalty * Geometric\ Average\ Precision\ (N) \tag{3.3}$$

- N is an n-gram size
- p_n is a precision

For NLG tasks N-gram size is considered most often between 1 and 4. As a result, the BLEU score is computed differently based on the N. BLEU score is a word-for-word comparison, therefore word sequencing heavily influences its value.

BLEU1-4 scores will be calculated to compare candidate questions generated by T5 with reference questions provided in the SQuAD.

3.6.2 Metric for evaluation of translation with explicit ordering (METEOR)

One of the shortcomings of BLEU is the fact that is based on exact word matching and sequencing inadvertently resulting in low scores for questions that may be formulated differently than the original reference, yet are contextually, semantically, and grammatically fine. This may inflate BLEU scores in the instances, where a candidate is significantly shorter than a reference. METEOR precision-recall-based metric was proposed to specifically address the weaknesses of BLEU (Banerjee and Lavie, 2005).

METEOR creates word alignments – effectively mapping words one-to-one between hypotheses and references. In the situation, where multiple references are provided; METEOR will calculate scores independently for each paring and output the best score (Banerjee and Lavie, 2005). The example alignment is provided in Figure 12.

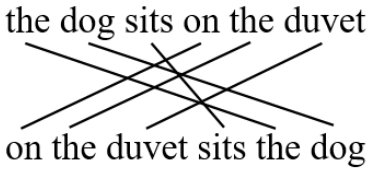

Figure 12: The example of alignment (by Author)

METEOR incorporates the recall metric, which has proven to strongly correlate with human judgement in terms of quality of translation (Lavie, Sagae and Jayaraman, 2004). Alignment, precision, and recall are all used to compute the METEOR score. The Fmean is a harmonic mean of precision (P) and recall (R).

$$Fmean = \frac{10PR}{R + 9P} \tag{3.4}$$

The penalty is a function of alignment and matching unigrams (single words) between the reference and the candidate.

$$Pen = 0.5 * \left(\frac{No\ of\ consecutive\ matching\ strings}{No\ of\ matching\ unigrams}\right)^3 \tag{3.5}$$

The resulting components are used to calculate the METEOR score.

$$METEOR = Fmean * (1 - Pen) \tag{3.6}$$

METEOR geometrically averages n-grams on the sentence level, thus providing a reliant metric for distinction purposes between systems (Banerjee and Lavie, 2005). Unlike other metrics, METEOR also uses stemming and synonymous matching along with the standard exact word matching. Lastly, it integrates higher orders of n-grams to boost grammaticality and fluency. As a result, METEOR proves to be a more robust metric compared to BLEU with results being more strongly correlated to human ones (Banerjee and Lavie, 2005) and therefore will be used as one of the metrics to assess generated questions.

3.6.3 Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

ROUGE is a set of NLP precision and recall-type metric consisting of ROUGE-N, ROUGE-S and ROUGE-L. While usually used for text summarization, ROUGE remains one of the most popular NLG metrics to evaluate the similarities between generated questions and ground-truth questions (Zhu and Hauff, 2021).

ROUGE-N (where N indicates the size of n-grams) measures the coverage of candidate n-grams against reference n-grams in sequence (Brown, Frishkoff and Eskenazi, 2005). ROUGE-1 and ROUGE-2 are usually used in conjunction to show the fluency between the questions. Therefore, ROUGE-1 and ROUGE-2 will compare sequential similarities of unigrams and bigrams and can be calculated by:

$$ROUGE\ Recall(N) = \frac{No\ of\ mathiching\ N-grams}{No\ of\ N-grams\ in\ the\ reference} \tag{3.7}$$

$$ROUGE\ Precision\ (N) = \frac{No\ of\ matching\ N-grams}{No\ of\ N-grams\ in\ the\ candidate} \tag{3.8}$$

Additionally, ROUGE-L is F-measure (binary classification) based on the Longest Common Subsequence (LCS) between a target question and candidate. Unlike the ROUGE-N classification, the ROUGE-L's LCS does not have to be contiguous and is concerned with the overall coverage rather than the sequential matching (Nema and Khapra, 2018). Thus, the ROUGE-L formula is the same as ROUGE-1 except matches are counted with no regard to the word sequential order.

Finally, ROUGE-S in other words skip-gram concurrence metric that investigates consecutive word pairings separated by one-or-more words. ROUGE-S will not be used as a metric due to its leniency, which may result in the lack of adequate question contextualization (Tay et al., 2019). The ROUGE scores are in the range between 0 and 1, where higher scores imply greater similarity with 1 being the exact match, whereas 0 implies no similarity at all. ROUGE-1, ROUGE-2 and ROUGE-L recall will be used to assess question generation systems.

3.6.4 The paraphraser parsing

The SQuAD_v1 dataset only provides a single reference question per paragraph given the answer. However, a semantically, grammatically, and contextually correct question can be formed around

the same topic in multiple ways. And while this task is easily performed and judged by humans, having a single question reference may prove to be problematic for comparison purposes when computational NLP evaluation metrics are used.

Therefore, a pre-trained T5 question paraphraser[3] is used on the reference question to create an extra ground truth question pool to provide an additional comparative framework against the predicted question. The question paraphraser has been previously trained Quora Question Pairs (QQP) dataset that consists of over 400 thousand question pairs. Figure 13 shows sample results from the paraphraser.

Original Reference:
What is the name of the location where the relay ending event was cancelled?

Paraphrased Results:
What is the name of the location where the relay ending event was cancelled?
What is the place where a relay ending event was cancelled?
What is the name of the location where a relay ending event was cancelled?
Where was the relay ending event cancelled?
What is the place where the relay ending event was cancelled?
What is the location where the relay ending event was cancelled?

Figure 13: Sample paraphraser results

It should be noted that the paraphrased results also include the original reference. Both single reference and paraphrased results will be compared against a predicted question to better understand the performance of each model based on BLEU, ROUGE and METEOR.

3.6.5 Bidirectional encoder representations from transformers score (BERTScore)

Unlike the previous metrics that were based on N-grams, BERTScore focuses on the learned contextual embeddings (Pennington, Socher and Manning, 2014). It has been shown that learned word embedding along with contextual embeddings depict better hypothesis representations for capturing semantic and lexical similarity compared to previously aforementioned metrics, which often are based only on surface-form similarity (Raffel et al., 2020).

BERTScore evaluates a similarity score based on reference and candidate sentence tokens. It incorporates learnings from the pre-trained contextual embeddings and matches tokens in reference and candidate sentences through pairwise cosine similarity. The BERTScore is between 0 and 1 with high scores indicating the semantical similarities between the candidates and references. It addresses two common weaknesses of conventional N-gram metrics.

Firstly, BERTScore can detect semantically correct phrases as opposed to other metrics, therefore avoiding issues related to performance underestimation and overestimation, when reference and candidate differ on the surface-level, but encapsulate the contextual or synonymous meaning (Zhang et al., 2020).

Secondly, N-gram metrics tend to struggle with capturing distant dependencies and penalize-critical

[3] https://huggingface.co/ramsrigouthamg/t5_paraphraser

ordering changes that result in different contextual meanings of sentences (Isozaki et al., 2010). Good examples are causations or correlations i.e. “Earth revolves around Sun” and “Sun revolves around Earth” would not be penalized heavily when assessed by standard NLP metrics. In contrast, BERTScore was trained to effectively analyze distant ordering and dependencies (Zhang et al., 2020). The BERTScore outputs F1 measure, precision and recall with all being used as a part of the semantical analysis.

3.6.6 Question-Answer Model

The Question-Answer model has been built as a final validation method to overcome the shortcomings related to the limitations imposed by the metrics and methods. The answer generation and answer-extraction generation models are used as additional verification methods i.e. for visual imaging (Changpinyo et al., 2022) (see section 2.1), where the relevancy of the generated question and corresponding answer are assessed based on the context of the image.

A similar method will be implemented but in the context of assessing generated questions and their generated answer created by the QA model given the context. The QA model is trained with a prepending input and the same hyperparameters as all other QG models on LR = 0.0001. Rather than a whole context, a target sentence containing an answer is provided to review whether a question is formulated well enough to generate a correct answer.

training input = Reference question + <sep> + Target Sentence

The QA generates answers based on the testing input, where a generated question is used instead of a reference question.

testing input = Generated question + <sep> + Target sentence

The generated answers will be compared with the reference answers. The exact matches along with the partial answers; answers where a reference is a subgroup of a candidate will be computed. For example, if a reference is “four” and the candidate is “four days”, the answer will be classified as partial. Additionally, questions, where answers are within the structure of the question and questions with missing question marks, will be quantified as well, as these will not be useful for tutoring systems.

Chapter 4. Results and Discussion

All results in section 4 will be reported accordingly to the logic displayed in the evaluation framework (see Figure 11). Additional validation loss curves will be used along with the learning parameter performances to understand the results of hyper tuning of each model. The paraphrased and unaltered results are displayed for the three best-performing learning rates.

4.1 Results

4.1.1 Learning Parameter Performance

The learning-rate hyper-parameter tuning was analyzed for each pre-processing strategy and its performance is assessed based on n-gram metrics. Both LR1 = 0.0001 and LR2 = 0.0005 achieve higher results at four categories compared to other models. The LR1 yields the best scores for unigram metrics BLEU-1 and ROUGE-1 as well as marginally better results for ROUGE-L and METEOR compared to LR3. On the other hand, LR3 attains greater scores for the higher-order n-gram metrics. Table 6 shows the results of the prepending model.

Prepending Model								
Learning Rate	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**METEOR**
LR1 = 0.0005	0.393	0.223	0.134	0.069	0.402	0.165	0.377	0.367
LR2 = 0.0001	**0.446**	0.284	0.187	0.109	**0.456**	0.227	**0.429**	**0.427**
LR3 = 0.00005	0.424	**0.289**	**0.210**	**0.121**	0.445	**0.237**	0.427	0.424
LR4 = 0.00003	0.376	0.234	0.172	0.094	0.398	0.191	0.376	0.369
LR5 = 0.000025	0.384	0.241	0.176	0.101	0.402	0.196	0.389	0.375
LR6 = 0.00001	0.365	0.223	0.147	0.090	0.386	0.187	0.368	0.354

Table 6: The prepending model evaluation score for varying learning-rates

The highlight model's learning-rate results are similar, but generally marginally lower compared to the ones achieved by a prepending model. Among all the learning rates, LR2 and LR3 are the best with an even split between the evaluation metrics. The only difference in the case of the highlight model and prepending model is the fact that LR2 achieves a better score (by a margin of 0.0 01) than LR3 with an even METEOR. Table 7 shows the results of the highlight model.

Highlight Model								
Learning Rate	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**METEOR**
LR1 = 0.0005	0.391	0.242	0.150	0.091	0.406	0.187	0.382	0.372
LR2 = 0.0001	**0.428**	**0.270**	0.166	0.090	**0.435**	0.209	**0.409**	**0.402**
LR3 = 0.00005	0.405	0.269	**0.184**	**0.093**	0.418	**0.215**	0.396	**0.402**
LR4 = 0.00003	0.378	0.246	0.166	0.093	0.389	0.199	0.376	0.369
LR5= 0.000025	0.380	0.246	0.163	0.084	0.392	0.197	0.378	0.369
LR6 = 0.00001	0.352	0.230	0.145	0.081	0.365	0.185	0.350	0.339

Table 7: The highlight model evaluation score for varying learning-rates

The best performing learning rate for the Hybrid model is LR2. Notably, LR2 achieves the highest scores in all evaluation metrics except BLEU-4 with a score of 0.101 being bested by LR4 with a score of 0.112. Table 8 shows the results of the hybrid model.

Hybrid Model								
Learning Rate	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**METEOR**
LR1 = 0.0005	0.408	0.263	0.158	0.081	0.435	0.209	0.411	0.407
LR2 = 0.0001	**0.422**	**0.276**	**0.194**	0.101	**0.443**	**0.224**	**0.422**	**0.414**
LR3 = 0.00005	0.390	0.264	0.181	0.095	0.415	0.215	0.394	0.395
LR4 = 0.00003	0.401	0.273	0.189	**0.112**	0.423	0.218	0.405	0.404
LR5 = 0.000025	0.380	0.242	0.152	0.077	0.387	0.186	0.371	0.366
Lr6 = 0.00001	0.343	0.195	0.128	0.080	0.365	0.170	0.347	0.329

Table 8: The hybrid model evaluation score for varying learning-rates

The worst performing model for all the pre-processing strategies was trained on LR6 = 0.00001. LR6 has achieved sub-standard results across all metrics compared to other models.

4.1.1.1 Model Validation Losses

The complementary model validation loss diagrams were created to better understand the performance of each model. The LR2 displays the lowest validation loss out of all the models, with LR1 achieving the highest validation loss out of all the models. Figure 14 shows the validation loss curves for a prepending model

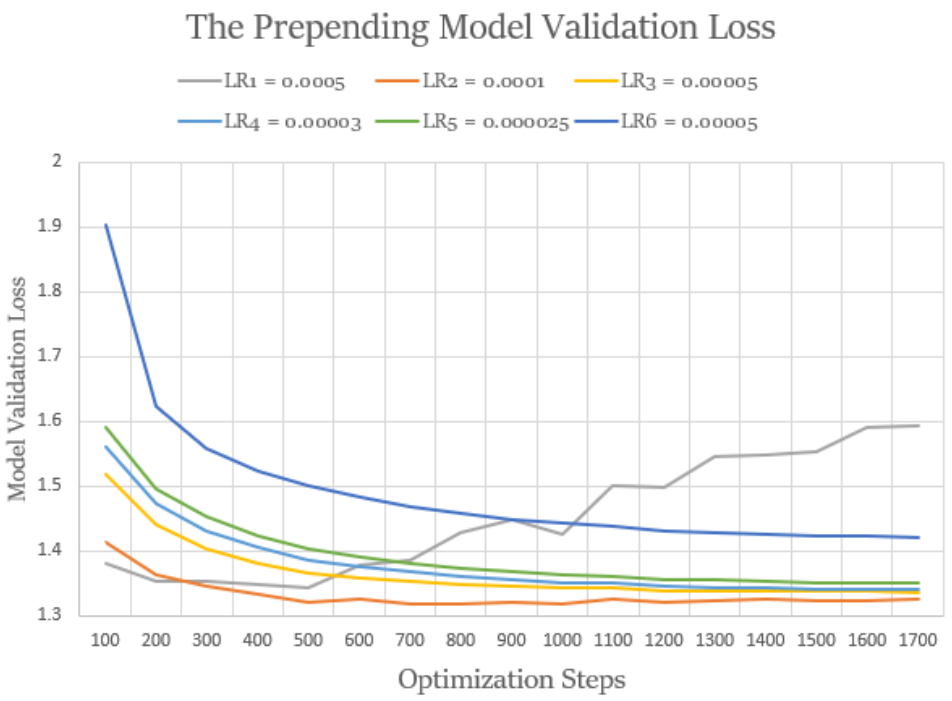

Figure 14. The prepending model validation loss

4.1.2. Paraphrased Question Results

The ground truth questions were paraphrased to create a larger pool of references against the candidate. Afterwards, the n-gram metrics were recalculated for LR2, which has consistently produced the best results as shown in section 4.1.1, for all model types. All three models display a similar net increase performance across all metrics with diminishing improvements for higher-order metrics. The highest improvement was recorded for BLEU-1 scores on average around 0.200 and the lowest for ROUGE-2 between 0.035 and 0.044. Tables 9, 10 and 11 provide a comparison of model performances between paraphrased and unaltered results for prepending, highlight and hybrid models respectively.

Prepending Model for LR2 = 0.0001								
Type	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**METEOR**
Paraphrased	0.644	0.459	0.334	0.227	0.665	0.291	0.632	0.534
Unaltered	0.446	0.284	0.187	0.109	0.456	0.227	0.429	0.427
DELTA	+0.198	+0.175	+0.147	+0.118	+0.088	+0.042	+0.073	+0.107

Table 9: The prepending model performance with paraphrased reference questions.

Highlight Model for LR2 = 0.0001								
Type	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**METEOR**
Paraphrased	0.631	0.442	0.326	0.207	0.626	0.298	0.610	0.515
Unaltered	0.428	0.270	0.166	0.090	0.435	0.209	0.409	0.402
DELTA	+0.203	+0.172	+0.160	+0.117	+0.083	+0.035	+0.074	+0.113

Table 10: The highlight model performance with paraphrased reference questions.

Hybrid Model for LR2 = 0.0001								
Type	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**METEOR**
Paraphrased	0.621	0.434	0.302	0.202	0.524	0.267	0.505	0.521
Unaltered	0.422	0.276	0.194	0.101	0.443	0.224	0.422	0.414
DELTA	+0.199	+0.159	+0.108	+0.101	+0.081	+0.044	+0.083	+0.107

Table 11: The hybrid model performance with paraphrased reference questions.

4.1.3. Semantical Results

The Precision, Recall and F1 BERTScore results were recorded for unaltered questions. The precision, recall and F1 scores for all models were between 0.85 and 0.875. The highlight model creates the most semantically correlated questions with the highest precision, recall and F1 score at 0.872, 0.863 and 0.867. On the other hand, a prepending model performs worse than the other two, but still achieves very high BERTScores across all variations with precision, recall and an F1 score of 0.844. 0.857 and 0.859. The semantical BERTScore are shown in Table 12.

BERTScore for LR2 = 0.0001			
Learning Rate	**Precision**	**Recall**	**F1 Score**
Prepend	0.844	0.857	0.859
Highlight	0.872	0.863	0.867
Hybrid	0.863	0.863	0.862

Table 12: The BERTScores for prepending, highlight and hybrid models

4.1.4. Answer Classification Results

The answer classification was performed for three models on 1000 samples. Figure 15 displays a categorical frequency histogram for prepending, highlight, hybrid models and Table 13. summarizes answer classification findings.

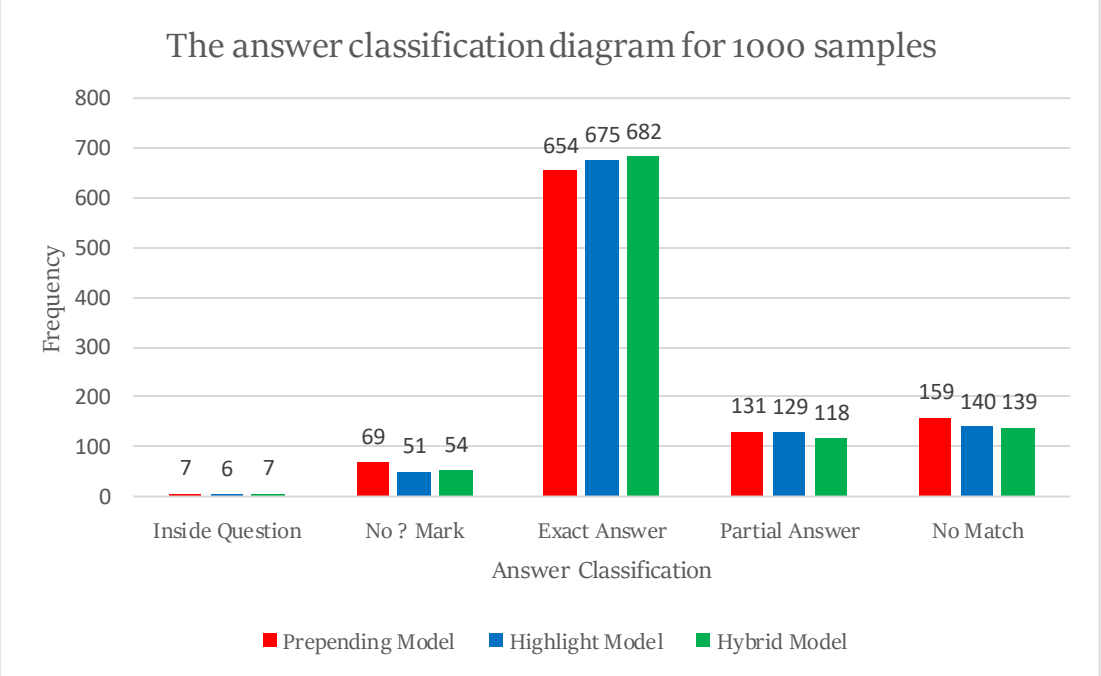

Figure 15: The answer classification diagram for 1000 samples

The hybrid model has achieved the highest number of 682 exact answers compared to other models and overall, it has produced the second greatest number of correctly answerable predicted questions with a total of 800 out of 1000 valid answers. Additionally, all models have achieved a significant percentage of valid answers, which are 78.5%, 80.4% and 80% for prepending,

highlight and hybrid models respectively. The lowest achieving model was a prepending one as it had 21.5% of invalid answers.

Type	Inside Question	No "?" Mark	Exact Answer	Partial Answer	No Match	Valid%	Invalid%
Prepending	7	69	654	131	159	**78.5**	**21.5**
Highlight	6	51	675	124	140	**80.4**	**19.6**
Hybrid	7	54	682	115	139	**80**	**20**

Table 13: The Answer classification summary

4.2 Discussion

4.2.1 Learning parameter and paraphraser results

The evaluation metric performance based on the learning rate for all three models shows relatively moderate variation. The learning rate governs the degree to which the weights inside the neural network are adjusted against the gradient loss. The higher the number the faster it approaches the downward slope as a result local minima might be missed resulting in the model divergence. On the other hand, a small learning rate may result in a prolonged time for the model to converge. The resulting slow convergence could be observed for LR6, which translates to subpar results compared to other models (As shown in Figure 14). The LR2 generally achieves the highest scores with the lowest model validation loss for all models.

It is usually considered that a score between 0.6 and 0.7 for BLEU-1 is a good indication of reference and candidate pairing. All LR2 models have scored BLEU-1 from 0.42 to 0.44, ROUGE-1 and METEOR around 0.42, which would indicate not only subpar but also the lack of fluency and dissimilarities between the candidate and the reference. However, that is not necessarily true, as n-gram metrics mainly observe sequential and word-to-word relationships, whereas questions can be formed in many various ways. Figure 16. represents a question generated by the LR2 prepending model with Table 14 displaying its n-gram scores.

Context: Meditation was an aspect of the practice of the yogis in the centuries preceding the Buddha.
Answer: the yogis

Reference Question: Meditation was an aspect of the practice of who?
Predicted: Who practised meditation in the centuries preceding the Buddha?

Figure 16: Example of generated question by LR2 Prepending Model

BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-2	ROUGE-1	ROUGE-L	METEOR
22.2	0	0	0	0	0.1	0.1	16.7

Table 14: The evaluation metrics scores

The predicted question given the context and answer is grammatically and contextually sound, but scores low due to the awkward structure of the reference question proving inherent flaws related to n-gram metrics. To accommodate for weaknesses related to n-gram metrics, additional references

were created by a question paraphraser[4], resulting in significant improvement across all three models for all evaluation metrics. The new BLEU-1 scores of around 0.63 with significant improvements in METEOR and ROUGE indicate that generated questions are similar to paraphrased references.

4.2.2 Semantic and Answer Classification Analysis

All three models have returned high BERTScore from 0.85 to 0.87 indicating a highly semantical relationship between candidate and reference questions. The BERTScore is strongly correlated with human judgments as in the case of the example displayed in figure 16, the BERTScore provides high scores of 0.811, 0.823 and 0.817 for the precision, recall and F1 categories. Therefore, correctly identifying that the questions are similar, regardless of the different forms and structures. Thus, the high BERTScores for all models indicate that questions carry a similar meaning, unlike what has been initially established with n-gram metrics without a paraphraser. To ensure that generated questions have a correct form, and grammar and are contextually fine, the QA framework was implemented.

The valid percentage answer of 78.4% for prepending model is less compared to the others. This occurs due to the way how prepending input is formatted. As the answer is located in front of the context, the model is not capable of distinguishing the correct answer inside the paragraph in the situation, where the same answer occurs multiple times inside the text. This directly results in the model generating fewer exact and partial answers with more mismatches. As it attempts to create answers around wrong targets.

On the other hand, the highlight model shows only a relatively small improvement over the prepending model with the highest percentage of valid answers at 80.4%. Additionally, it achieves the most exact answers out of all the methods. This is related to the way the method highlights the answer within a text resulting in more exact matches and fewer partial answers when modelled with the QA framework.

It should be noted that it contrasts with what has been suggested by Chan and Fan (2019) who used prepending and highlight methods for QG with the BERT model. The results very heavily favoured a highlighting method based on BLEU, METEOR and ROUGE-L evaluations. However, unlike BERT which implements absolute positional embedding, T5 utilizes SRPE that allows the model to update the self-attention weights with revised scalars based on the position of the word allowing it to successfully encode positional information. The T5 uses this advantage to contextualize better around the word (Wu et al. 2021).

Lastly, the hybrid method which prepends an answer and highlights the sentence results in more exact answers compared to other methods with the fewest number of partial answers. The hybrid model still produces a similar percentage of valid answers as the highlight model and it does not display considerable improvements over the conventional highlight formatting.

[4] https://huggingface.co/ramsrigouthamg/t5_paraphraser

Chapter 5. Conclusions

In this document, a thorough review of state-of-the-art methods for question generation has been undertaken. The literature and technology survey provides a detailed evolutionary path of question generation models from early template-based and rule-based systems to neural question generation models like the RNN, LSTM or Seq2Seq, to recently invented transformers (Vaswani et al., 2017). The self-attentive transformers could perform natural language generation tasks better compared to previous models, while also operating with a high degree of parallelization allowing transformers to be extensively pre-trained resulting in systems like BERT and new state-of-the-art T5.

To find an optimal way to automatically generate questions, the investigation of the pre-processing strategies of encoder inputs under varying learning-rate hyperparameter setups for T5 was carried out on the Stanford Question Answering Dataset (SQuAD) version 1. The prepending, highlight and proposed hybrid methods were scrutinized. The evaluation framework not only incorporated standard n-gram metrics (BLEU, ROGUE and METEOR) and semantical metrics such as BERTScore but also tried to address the weaknesses of these metrics through the implementation of the question paraphraser and question-answer (QA) model to classify the generated questions.

Overall, all three models have proven to successfully create correct questions with the very similar type of performances for standard n-gram metrics both with paraphrased and unaltered results. Furthermore, the generated questions were semantically correlated with the reference questions for all models. Lastly, the answer classification framework was implemented to review whether generated questions contextualize well enough to generate an answer that matches the reference answers. Likewise, all models have achieved a satisfactory percentage of exact and partial matches.

In terms of future works, the implementation of question paraphraser has shown a positive impact on the performance of all models. A different approach could be undertaken, where the model is trained with a paraphraser to produce several candidates with each candidate being evaluated with n-gram and semantic metrics. Finally, the QA model would be used to verify whether generated questions produce valid answers. As a result, the model would be trained with more artificial data. Although similar to the approach used in this research, this approach would increase the number of generated questions. And in the situation where a single one of them produces a correct answer, it would mean that such a question could be used for quiz generation for intelligent tutoring systems.

Chapter 6. Bibliography

1. Alammar, J., 2018. The illustrated transformer - jay alammar - visualizing machine learning one concept at a time.[online] Avalaible at:< http://jalammar.github.io/illustrated-transformer/>
   [Accessed 24 August 2022]
2. Bahdanau, D., Cho, K. and Bengio, Y., 2014. Neural Machine Translation by Jointly Learning to Align and Translate.
3. Banerjee, S. and Lavie, A., 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. IEEvaluation@ACL.
4. Bao, J., Tang, D., Duan, N., Yan, Z., Lv, Y., Zhou, M. and Zhao, T., 2018. Table-to-Text: Describing Table Region with Natural Language.
5. Brown, J., Frishkoff, G.A. and Eskénazi, M., 2005. Automatic Question Generation for Vocabulary Assessment. HLT.
6. Cao, Z., Tatinati, S. and Khong, A.,W., 2020. Controllable Question Generation via Sequence-to-Sequence Neural Model with Auxiliary Information. 2020 International Joint Conference on Neural Networks (IJCNN), 1-7.
7. Chan, Y. and Fan, Y., 2019. A Recurrent BERT-based Model for Question Generation. EMNLP.
8. Changpinyo, S., Kukliansky, D., Szpektor, I., Chen, X., Ding, N. and Soricut, R., 2022. All You May Need for VQA are Image Captions.
9. Chen, Y., Wu, L. and Zaki, M., 2022. Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation.
10. Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y., 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.
11. Cui, S., Bao, X., Zu, X., Guo, Y., Zhao, Z., Zhang, J. and Chen, H., 2021. OneStop QAMaker: Extract Question-Answer Pairs from Text in a One-Stop Approach. ArXiv, abs/2102.12128.
12. Devlin, J., Chang, M., Lee, K. and Toutanova, K., 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv, abs/1810.04805.
13. DiPietro, R. and Hager, G.,D., 2019. Deep learning: RNNs and LSTM. in Handbook of Medical Image Computing and Computer Assisted Intervention. Elsevier, pp. 503-519.
14. Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M. and Hon, H., 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. ArXiv, abs/1905.03197.
15. Du, X. and Cardie, C., 2018. Harvesting Paragraph-level Question-Answer Pairs from Wikipedia. ACL.
16. Du, X., Shao, J. and Cardie, C., 2017. Learning to Ask: Neural Question Generation for Reading Comprehension.
17. Duan, N., Tang, D., Chen, P. and Zhou, M., 2017. Question Generation for Question Answering. EMNLP. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.
18. Fattoh, E., I., 2014 .Semantic Based Automatic Question Generation using Artificial Immune System
19. Gao, Y., Bing, L., Li, P., King, I. and Lyu, M. R., 2019. Generating Distractors for Reading Comprehension Questions from Real Examinations. Proceedings of the AAAI Conference on

Artificial Intelligence, 33(01).
20. Godayal, D., 2018. An introduction to part-of-speech tagging and the Hidden Markov Model. [online] Available at: <https://www.freecodecamp.org/news/an-introduction-to-part-of-speech-tagging-and-the-hidden-markov-model-953d45338f24/> [Accessed 1 September 2022].
21. Grover, K., Kaur, K., Tiwari, K., Rupali P. and Kumar, P., 2021. Deep Learning Based Question Generation Using T5 Transformer.
22. Heilman, M. and Smith, N.A., 2010. Good Question! Statistical Ranking for Question Generation. NAACL.
23. Huang, Y. and He., L., 2016. Automatic generation of short answer questions for reading comprehension assessment.
24. Isozaki, H., Hirao, T., Duh, K., Sudoh, K. and Tsukada, H., 2010. Automatic Evaluation of Translation Quality for Distant Language Pairs. EMNLP.
25. Jernite, Y., Bowman, S.R. and Sontag, D.A., 2017. Discourse-Based Objectives for Fast Unsupervised Sentence Representation Learning. ArXiv, abs/1705.00557.
26. Kim, J., K., Kim, Y., B., Sarikaya, R. and Fosler-Lussier, E., 2017. Cross-lingual transfer learning for POS tagging without cross-lingual resources.
27. Kim, Y., Denton, C., Hoang, L. and Rush, A., 2017. Structured Attention Networks, In International Conference on Learning Representations.
28. Kim, Y., Lee, H., Shin, J. and Jung, K., 2019. Improving Neural Question Generation using Answer Separation. ArXiv, abs/1809.02393.
29. Kuchaiev, O. and Ginsburg, B., 2017. Factorization tricks for LSTM networks.
30. Kunichika, H., Katayama, T., Hirashima, T. and Takeuchi, A., 2001. Automated Question Generation Methods for Intelligent English Learning Systems and its Evaluation.
31. Lamba, D. and Hsu, W.H., 2021. Constraint-Based Neural Question Generation Using Sequence-to-Sequence and Transformer Models for Privacy Policy Documents. International Journal of Knowledge Engineering.
32. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P. and Soricut, R., 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. ArXiv, abs/1909.11942.
33. Lavie, A., Sagae, K. and Jayaraman, S., 2004. The significance of recall in automatic metrics for MT evaluation. AMTA.
34. Le, N., Nguyen, N., Seta, K. and Pinkwart, N., 2014. Automatic question generation for supporting argumentation. pp.117-127.
35. Li, Y., Duan, N., Zhou, B., Chu, X., Ouayang, W., Wang, X. and Zhou, M., 2018. Visual question generation as dual task of visual question answering.
36. Lin, C., 2004. ROUGE: A Package for Automatic Evaluation of Summaries. ACL 2004.
37. Liu, M., & Calvo, R. A., 2012. Using information extraction to generate trigger questions for academic writing support. Paper presented at the Intelligent Tutoring Systems.
38. Logeswaran, L. and Lee, H., 2018. An efficient framework for learning sentence representations. ArXiv, abs/1803.02893.
39. Mostafazadeh, N., Misra, I., Devlin, J., Mitchell, M., He, X. and Vanderwende, L., 2016. Generating Natural Questions About an Image. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1802-1813.
40. Mostow, J. and Chen, W., 2009. Generating Instruction Automatically for the Reading Strategy of Self-Questioning. AIED.
41. Nema, P. and Khapra, M.M., 2018. Towards a Better Metric for Evaluating Question

Generation Systems. EMNLP.
42. Pandraju, S. and Mahalingam, S., 2021. Answer-Aware Question Generation from Tabular and Textual Data using T5. p.256.
43. Papineni, K., Roukos, S., Ward, T. and Zhu, W., 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. ACL.
44. Pennington, J., Socher, R. and Manning, C.D., 2014. GloVe: Global Vectors for Word Representation. EMNLP.
45. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L., 2018. Deep Contextualized Word Representations. NAACL.
46. Radford, A. and Narasimhan, K., 2018. Improving Language Understanding by Generative Pre-Training.
47. Raffel, C., Shazeer, N.M., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. and Liu, P.J., 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. ArXiv, abs/1910.10683.
48. Rus, V., Wyse, B., Piwek, P., Lintean, M., Stoyanchev, S., & Moldovan, C., 2010. Overview of the first question generation shared task evaluation challenge. Paper presented at the Proceedings of the Third Workshop on Question Generation
49. Sadigh, D., Seshia, S.A., and Gupta, M., 2012. Automating exercise generation: a step towards meeting the MOOC challenge for embedded systems. WESE '12.
50. Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Networks, 61, pp.85-117.
51. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G. and Dean, J., 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer.
52. Song, L., Wang, Z., Hamza, W., Zhang, Y. and Gildea, D., 2018. Leveraging Context Information for Natural Question Generation.
53. Stancheva, N., S., Popchev, I., Stoyanova-Doycheva A. and. Stoyanov, S., 2016. Automatic generation of test questions by software agents using ontologies.
54. Subramanian, S., Wang, T., Yuan, X. and Trischler, A., 2018. Neural Models for Key Phrase Extraction and Question Generation. ArXiv, abs/1706.04560.
55. Sutskever, I., Vinyals, O. and Le, Q., 2014. Sequence to Sequence Learning with Neural Networks.
56. Tay, W., Joshi, A., Zhang, X., Karimi, S. and Wan, S., 2019. Red-faced ROUGE: Examining the Suitability of ROUGE for Opinion Summary Evaluation. ALTA.
57. TensorFlow, 2022. [online] Available at: <https://www.tensorflow.org/text/tutorials/text_generation> [Accessed 1 September 2022].
58. Vaswani, A., Shazeer, N.M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I., 2017. Attention is All you Need. ArXiv, abs/1706.03762.
59. Wang, T., Yuan, X. and Trischler, A., 2017. A Joint Model for Question Answering and Question Generation. ArXiv, abs/1706.01450.
60. Willis, A., Davis, G.M., Ruan, S.S., Manoharan, L., Landay, J.A. and Brunskill, E., 2019. Key Phrase Extraction for Generating Educational Question-Answer Pairs. Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale.
61. Wu, J., Zhang, R., Mao, Y., & Chen, J., 2021. On Scalar Embedding of Relative Positions in Attention Models
62. Xie, J., Fang, W., C., Huang Q. and Li, Q., 2022. Knowledge-Based Visual Question Generation.

63. Zhang, C., Zhang, H. and Wang, J., 2022. Downstream Transformer Generation of Question-Answer Pairs with Preprocessing and Postprocessing Pipelines.
64. Zhang, S., Qu, L., You, S., Yang, Z. and Zhang, J., 2017. Automatic generation of grounded visual questions.
65. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q. and Artzi, Y., 2020. BERTScore: Evaluating Text Generation with BERT. ArXiv, abs/1904.09675.
66. Zhang, L. and VanLehn, K., 2016. How do machine-generated questions compare to human-generated questions?
67. Zhang, R., Guo, J., Chen, L., Fan, Y. and Cheng, X., 2022. A Review on Question Generation from Natural Language Text. p. 11.
68. Zhou, Q., Yang, N., Wei, F., Tan, C., Bao, H. and Zhou, M., 2017. Neural Question Generation from Text: A Preliminary Study. NLPCC.
69. Zhu, P. and Hauff, C., 2021. Evaluating BERT-based Rewards for Question Generation with Reinforcement Learning. Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval.