Question-answering model leveraging Quora-QuAD

Github Repository Link: https://github.com/ArushiGupta26/Quora_QA

Introduction

In the rapidly evolving landscape of artificial intelligence and natural language processing, Question Answering (QA) systems have emerged as a significant area of research and application. Closed-book question answering (CBQA) is an advanced task in natural language processing (NLP) where a model is required to answer questions based solely on its pre-trained knowledge, without access to any external text or document during inference. Unlike open-book question answering, which allows the model to refer to documents or databases to find answers, CBQA challenges the model to leverage its internalized understanding and reasoning capabilities.

The CBQA task is particularly significant because it tests the model's ability to understand, memorize, and generalize information from its training data. It is an essential step toward developing more autonomous and intelligent AI systems that can operate independently of external information sources. This capability is useful in various applications, such as virtual assistants, customer support, and educational tools, where real-time, accurate, and contextually appropriate responses are crucial.

Developing effective CBQA models involves training them on large and diverse datasets, ensuring they are exposed to a wide range of topics and question types. Fine-tuning these models on specific domains can further enhance their performance by aligning their pre-existing knowledge with the particular needs of the target application.

Several famous models have been developed for CBQA, showcasing the progress in this field:

1. **GPT-3 (Generative Pre-trained Transformer 3)**: Developed by OpenAI, GPT-3 is one of the largest and most powerful language models, known for its ability to generate coherent and contextually relevant responses to a wide range of questions based on its extensive training data.
2. **T5 (Text-to-Text Transfer Transformer)**: Created by Google Research, T5 treats all NLP tasks as a text-to-text problem, making it versatile and effective for CBQA tasks. It can generate accurate answers by transforming input questions into output answers.
3. **BERT (Bidirectional Encoder Representations from Transformers)**: Developed by Google, BERT is known for its bidirectional training approach, which helps in understanding the context of words in a sentence. It has been fine-tuned for various question-answering tasks, including CBQA.

4. **RoBERTa (Robustly optimized BERT approach)**: An optimized version of BERT by Facebook AI, RoBERTa improves upon the pre-training methods, resulting in enhanced performance for CBQA tasks.
5. **ALBERT (A Lite BERT)**: Also developed by Google Research, ALBERT reduces the number of parameters compared to BERT while maintaining strong performance, making it efficient for CBQA.
6. **BART (Bidirectional and Auto-Regressive Transformers)**: Developed by Facebook AI, BART is a denoising autoencoder that combines the strengths of both bidirectional and autoregressive models. It has shown strong performance in a variety of NLP tasks, including CBQA, by effectively handling text generation and understanding tasks.

In summary, closed-book question answering represents a challenging and promising area of research in NLP, pushing the boundaries of what AI models can achieve in terms of knowledge retention and application. The advancements in models like GPT-3, T5, BERT, RoBERTa, ALBERT, and BART demonstrate the progress and potential in this field, paving the way for more intelligent and autonomous AI systems.

Quora Question Answer Dataset (Quora-QuAD) contains 56,402 question-answer pairs scraped from Quora. I will fine tune **Flan-T5 and BART** for closed-book question answering using the above question-answer pairs scraped from Quora.

Literature Survey

Question Generation (QG) is a crucial task in natural language processing (NLP) with significant implications for various downstream applications. While recent advancements in open-book QG—where models generate questions based on provided answer-context pairs—have shown considerable progress, the challenge of generating natural questions in a closed-book setting persists. This setting lacks supporting documents, making it more practical yet difficult.

The study [3] introduces an innovative QG model specifically designed for the closed-book scenario. This model aims to enhance the semantic understanding of long-form abstractive answers and improve information retention within its parameters. It achieves this through the integration of contrastive learning and an answer reconstruction module. They use BART-base [5], a widely used sequence-to-sequence framework with 12 layers and a hidden size of 1024, as the backbone model.

Key aspects of this research include:

1. **Model Design**: The proposed QG model is tailored to better grasp the semantics of answers and store information effectively through advanced techniques like contrastive learning and answer reconstruction.
2. **Empirical Validation**: The model's performance was validated through experiments on public datasets and a new WikiCQA dataset. The results demonstrated that the proposed model outperformed baseline models in both automatic and human evaluations.
3. **Enhancing QA Systems**: The study also explored how the proposed QG model could be leveraged to improve existing question-answering systems, showcasing its broader applicability and effectiveness.

The empirical results indicate that this QG model significantly enhances the performance of closed-book question generation, making it a valuable contribution to the field. The study underscores the model's potential to improve closed-book question-answering tasks, thereby advancing the capabilities of NLP systems in practical applications.

Flan-T5 is a powerful text-to-text language model introduced by Google, excelling in a range of tasks including summarization, translation, and question answering. Presented in the paper "Scaling Instruction-Fine Tuned Language Models," [1] Flan-T5 is an extension of the original T5 architecture, enhanced through a method known as instruction finetuning. This approach involves training the model on a diverse array of tasks, significantly improving its performance on previously unseen tasks.

In the context of closed-book question answering (CBQA), Flan-T5's capabilities are particularly noteworthy. CBQA tasks require the model to answer questions from memory without access to

external information during inference, testing its internalized understanding and generalization of the training data.

One significant advantage of Flan-T5 is its accessibility and efficiency. Unlike larger models such as Llama 2 and GPT-NeoX, Flan-T5 is relatively lightweight and can be fine-tuned on a free Colab GPU, making it an attractive option for research and practical applications.

As mentioned in [2], the fine-tuning process for Flan-T5 on CBQA tasks involves several key steps:

1. **Dataset Preparation**: The dataset used for fine-tuning consists of 56,402 question-answer pairs scraped from Quora. This dataset was collected using a rotating proxy session from Bright Data, ensuring a comprehensive and diverse set of examples for training.
2. **Library Installation**: The necessary libraries, including transformers, tokenizers, datasets, evaluate, rouge_score, sentencepiece, and huggingface_hub, are installed to facilitate the fine-tuning process.
3. **Loading the Dataset**: The Quora dataset is loaded and split into training and test sets using the HuggingFace datasets library.
4. **Model and Tokenizer Initialization**: Flan-T5-Base and its corresponding tokenizer are loaded from the HuggingFace hub. A data collator is also set up to handle the batching of sequences for training.
5. **Data Preprocessing**: The preprocessing function prefixes the questions with "answer the question:" and tokenizes both the questions and answers. This ensures that the model learns to generate answers based on the tokenized input questions.
6. **Evaluation Metrics**: The Rouge score is used to evaluate the model's performance during training. This metric compares the generated text with the target text by analyzing their n-grams.
7. **Training Configuration**: Training arguments are configured to optimize batch size and training speed on a GPU with 16GB VRAM. Parameters such as learning rate, batch size, weight decay, number of epochs, and evaluation strategy are set to maximize performance.
8. **Model Training**: The Seq2SeqTrainer from the HuggingFace transformers library is used to train the model. The training process involves monitoring the Rouge score to ensure the model's effectiveness in generating accurate answers.

The integration of Flan-T5 with the HuggingFace ecosystem and the detailed fine-tuning procedure demonstrates the model's versatility and effectiveness for CBQA tasks. This approach offers a scalable and accessible solution for developing sophisticated question-answering systems, making significant strides in the field of natural language processing.

The study [4] discusses how Large language models (LLMs) have demonstrated strong performance in answering questions and generating long-form texts, particularly in few-shot closed-book settings. Evaluating the quality of short-form question answering can be done using well-known metrics, but long-form text evaluation presents more challenges. Researchers have addressed this by combining both tasks: generating long-form answers to questions that require comprehensive, multifaceted responses.

This approach acknowledges that such questions often contain ambiguities or require information from multiple sources. To handle this complexity, the study introduces query refinement prompts that guide LLMs to explicitly address the multifaceted nature of questions, producing long-form answers that cover various aspects of the inquiry.

The effectiveness of this method was tested on two long-form question answering datasets, ASQA and AQuAMuSe. The results showed that using query refinement prompts allowed the models to outperform fully fine-tuned models in the closed-book setting. Furthermore, these models achieved performance levels comparable to retrieve-then-generate open-book models, which have access to external information sources.

In summary, this technique of using query refinement prompts for generating multifaceted, long-form answers in closed-book question answering demonstrates a significant advancement. It not only enhances the quality of the answers but also bridges the gap between closed-book and open-book models, highlighting a promising direction for future research in NLP.

Links:

[1] https://arxiv.org/pdf/2210.11416
[2] https://www.toughdata.net/blog/post/finetune-flan-t5-question-answer-quora-dataset
[3] https://aclanthology.org/2023.eacl-main.230.pdf
[4] https://arxiv.org/abs/2210.17525
[5] https://aclanthology.org/2020.acl-main.703/

Methodology

Data Exploration, Cleaning, and Preprocessing

Data Exploration

1. **Dataset Overview**:
    - The dataset, sourced from Hugging Face, consists of question-answer pairs.
    - It was initially split into training (45121 rows) and testing (11281 rows) datasets.
2. **Data Examination**:
    - Converted the dataset into pandas DataFrames (df_train and df_test).
    - The training set has 45121 entries with columns question and answer.
    - The distribution of question and answer lengths was analyzed, showing a range of lengths and typical values.
3. **Profiling and Analysis**:
    - Generated a profiling report using ydata_profiling on a sample of the data, revealing insights into data types, distributions, and potential issues.
    - Detected the presence of emojis and URLs in the text using regex.

Data Cleaning

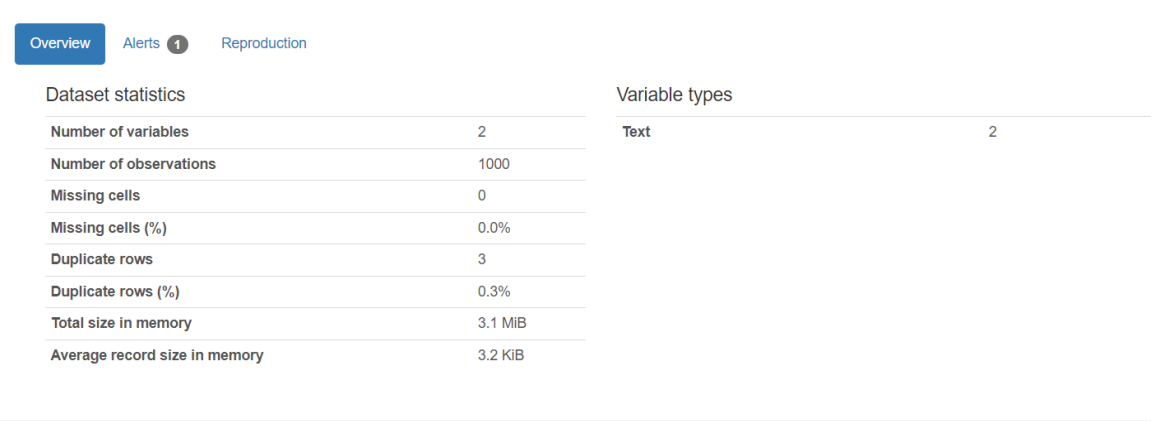
1. **Text Cleaning**:
    - URLs were removed.
    - Emojis were converted to textual descriptions.
    - Text was normalized by converting to lowercase, removing HTML tags, punctuation, and extra spaces.
2. **Preprocessing**:
    - **Stop Words Removal**: Removed common stop words using NLTK.
    - **Tokenization and Normalization**: Tokenized text, applied lemmatization (or stemming if chosen), and reconstructed cleaned text.
    - **Handling Duplicates and Null Values**: Duplicates and rows with null values were removed.
3. **Data Transformation**:
    - Preprocessed DataFrames were saved as CSV files.
    - Converted preprocessed DataFrames back to Hugging Face dataset format for consistency.

Preprocessing Improvements and Analysis

1. **Data Comparison**:
    - **Distribution**: Compared length distributions of original and preprocessed data. Noted differences in question distribution and slight changes in answer distribution.

- **Vocabulary Analysis**: Found reductions in vocabulary size and redundancy, with preprocessed data having a smaller yet relevant vocabulary.
- **Readability**: Readability scores showed increased complexity in preprocessed data, but semantic meaning was largely retained.

2. **Profiling Reports**:
   - Generated post-preprocessing profiling report, confirming removal of stop words, duplicates, and handling of missing values.



Overview | Alerts 1 | Reproduction

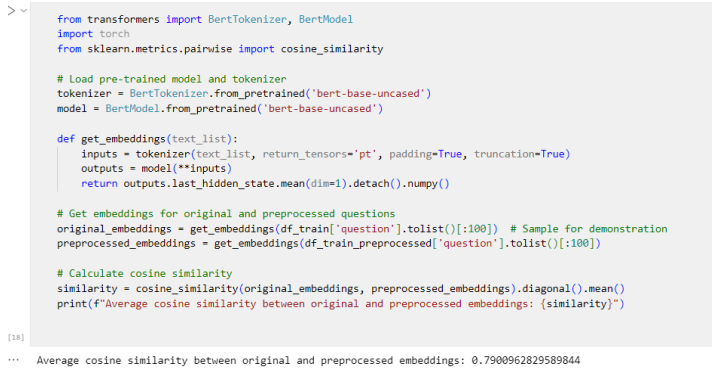
Dataset statistics

Number of variables	2
Number of observations	1000
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	3
Duplicate rows (%)	0.3%
Total size in memory	3.1 MiB
Average record size in memory	3.2 KiB

Variable types

Text	2

3. **Cosine Similarity**:
   - Evaluated semantic similarity between original and preprocessed texts using BERT embeddings. The average cosine similarity indicated that semantic meaning was well-preserved through preprocessing.



```
from transformers import BertTokenizer, BertModel
import torch
from sklearn.metrics.pairwise import cosine_similarity

# Load pre-trained model and tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = BertModel.from_pretrained('bert-base-uncased')

def get_embeddings(text_list):
    inputs = tokenizer(text_list, return_tensors='pt', padding=True, truncation=True)
    outputs = model(**inputs)
    return outputs.last_hidden_state.mean(dim=1).detach().numpy()

# Get embeddings for original and preprocessed questions
original_embeddings = get_embeddings(df_train['question'].tolist()[:100])  # Sample for demonstration
preprocessed_embeddings = get_embeddings(df_train_preprocessed['question'].tolist()[:100])

# Calculate cosine similarity
similarity = cosine_similarity(original_embeddings, preprocessed_embeddings).diagonal().mean()
print(f"Average cosine similarity between original and preprocessed embeddings: {similarity}")
```

[18]

```
Average cosine similarity between original and preprocessed embeddings: 0.7900962829589844
```

The overall preprocessing steps enhanced data cleanliness and structure, with improvements in handling language-specific content and potential model compatibility.

Model Selection and Evaluation:

In the evolving landscape of Natural Language Processing (NLP), the choice of models for specific tasks is crucial for achieving optimal performance. For tasks such as text generation and closed-book question answering, models like BART and T5 have shown exceptional

capabilities. This report discusses why BART and T5 are chosen over other models for these tasks, with a focus on their effectiveness in closed-book question answering.

Model Overview

1. BART (Bidirectional and Auto-Regressive Transformers)

BART is a versatile model designed by Facebook AI to handle a wide range of NLP tasks. It combines the strengths of bidirectional and autoregressive transformers, which significantly contributes to its performance in text generation and closed-book question answering.

- **Architecture**: BART's bidirectional encoder (similar to BERT) captures context from the entire input sequence, while its autoregressive decoder (similar to GPT) generates coherent output. This dual structure allows BART to understand the full context and produce relevant answers.
- **Pre-training Tasks**: The denoising autoencoder pre-training objective helps BART recover corrupted text and learn robust representations, which is beneficial for generating accurate responses in closed-book question answering.
- **Text Generation**: BART’s ability to handle complex text generation tasks, such as summarization and text completion, translates well into generating answers to questions when fine-tuned appropriately.

2. T5 (Text-To-Text Transfer Transformer)

T5, developed by Google Research, is designed to handle all NLP tasks as text-to-text problems, making it highly effective for a range of applications including text generation and closed-book question answering.

- **Unified Framework**: T5’s approach of framing every NLP task as a text generation problem allows it to address diverse tasks with a single model architecture. This uniform framework is particularly useful for generating answers to questions by treating them as text completion tasks.
- **Pre-training Objective**: T5's span corruption objective, where the model predicts masked spans of text, helps it learn to generate coherent and contextually relevant text. This objective enhances its performance in generating precise answers in closed-book scenarios.
- **Scalability**: T5's availability in various sizes allows for adjustments based on computational resources and performance needs, making it suitable for both large-scale and resource-constrained environments.

Reasons for Choosing BART and T5

1. Superior Text Generation Capabilities

- **Contextual Understanding**: Both BART and T5 excel in understanding and generating text by leveraging transformer architectures. BART's bidirectional encoder and T5's text-to-text framework ensure high-quality and contextually accurate outputs.
- **Fine-Tuning Flexibility**: Both models can be fine-tuned on specific datasets to adapt to text generation and closed-book question answering tasks, offering customization for various applications.

2. Effectiveness in Closed-Book Question Answering

- **BART**: BART's bidirectional encoder allows it to grasp the context of a question comprehensively, while its autoregressive decoder generates precise and relevant answers. This capability makes BART effective in closed-book question answering, where generating accurate responses without relying on external knowledge is crucial.
- **T5**: T5's text-to-text approach is well-suited for closed-book question answering, as it treats the task as a text generation problem where the input question is transformed into an appropriate answer. The span corruption pre-training helps T5 generate accurate and contextually relevant answers even when no external information is available.

3. Pre-training and Objective Alignment

- **BART**: The denoising autoencoder pre-training is advantageous for generating coherent text and recovering information from incomplete input, which is beneficial for answering questions without additional context.
- **T5**: The span corruption objective trains T5 to handle diverse text generation scenarios, including closed-book question answering, by learning to generate complete and contextually appropriate answers from partial information.

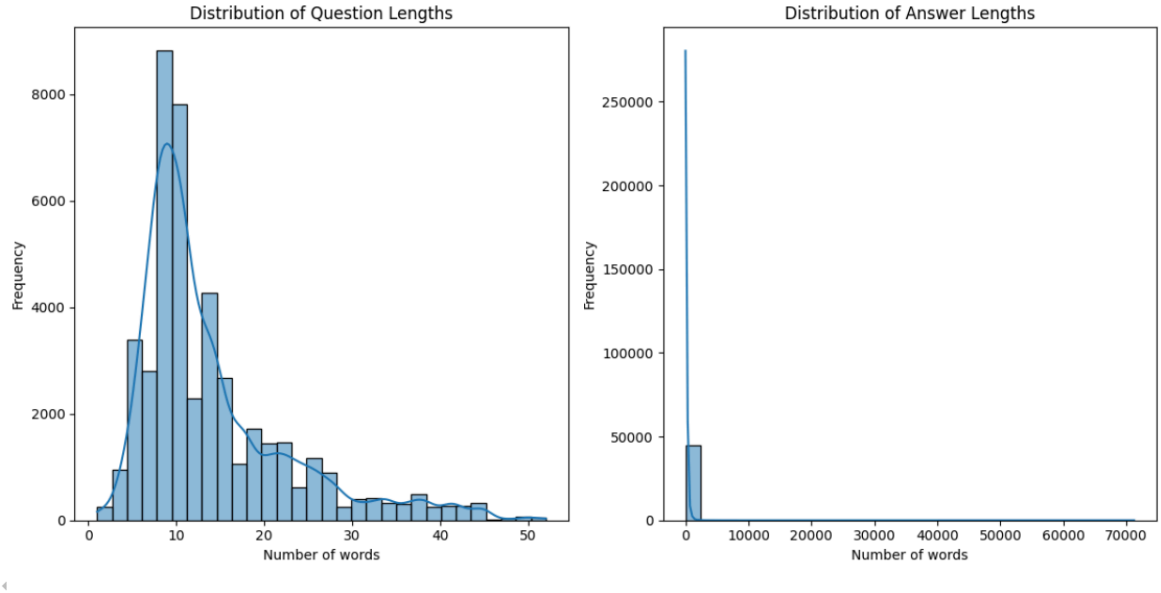
Evaluation

Performance and Metrics

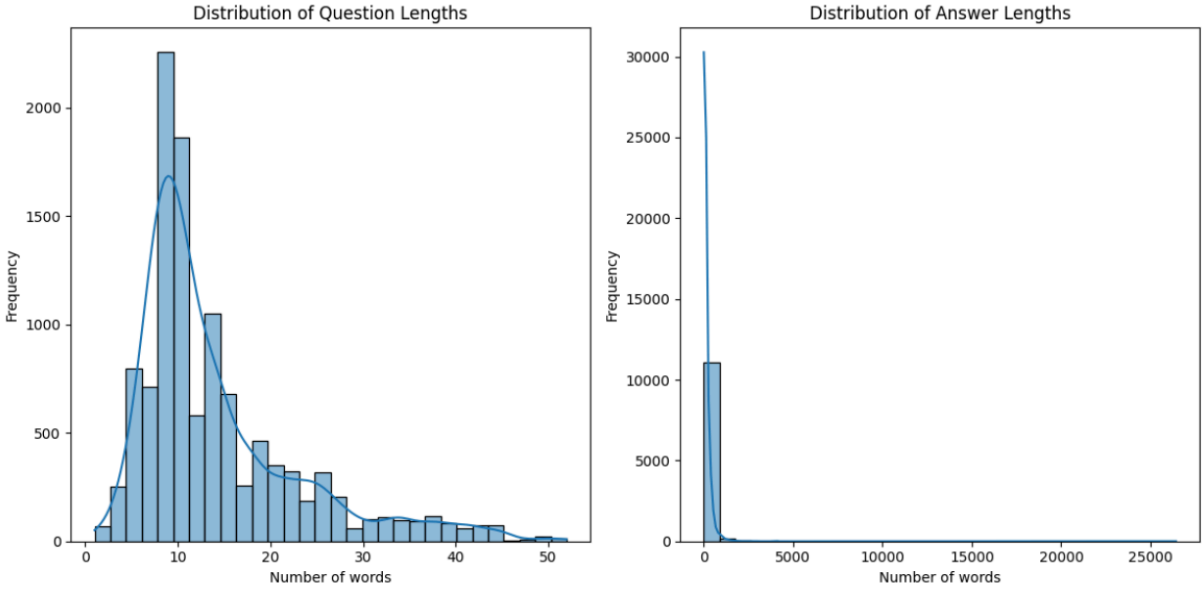
Empirical Performance: Both models have demonstrated strong performance on benchmarks for text generation and closed-book question answering. BART and T5 have achieved high scores on metrics such as ROUGE for summarization and SQuAD for question answering, validating their effectiveness in these tasks.

Visualization

Distributions

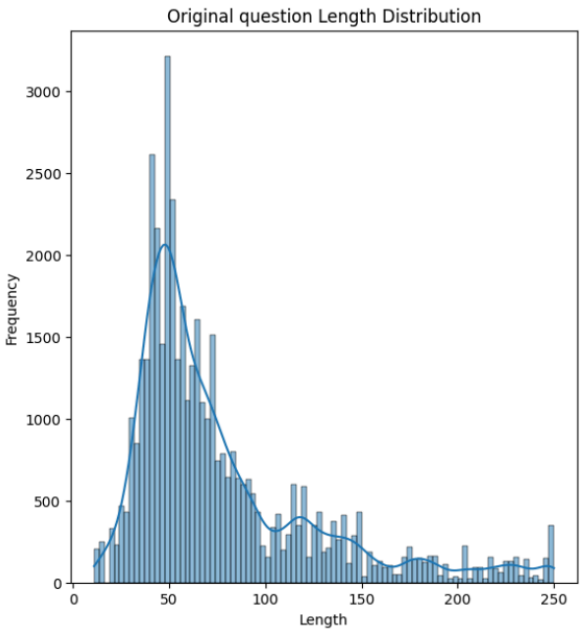


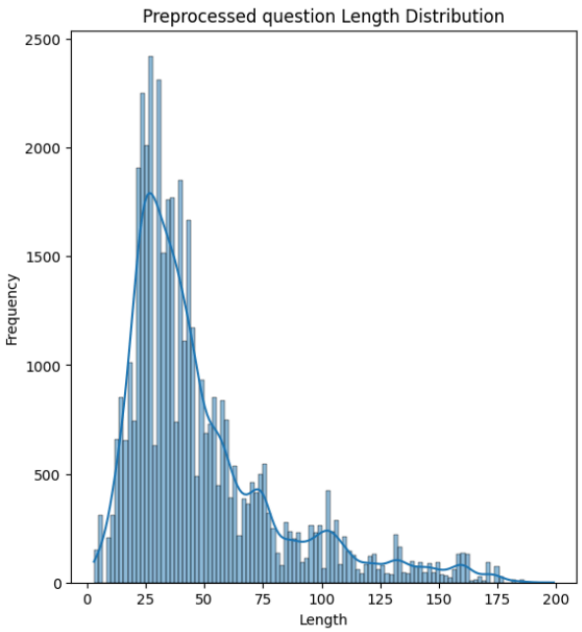
Distribution of Training Dataset

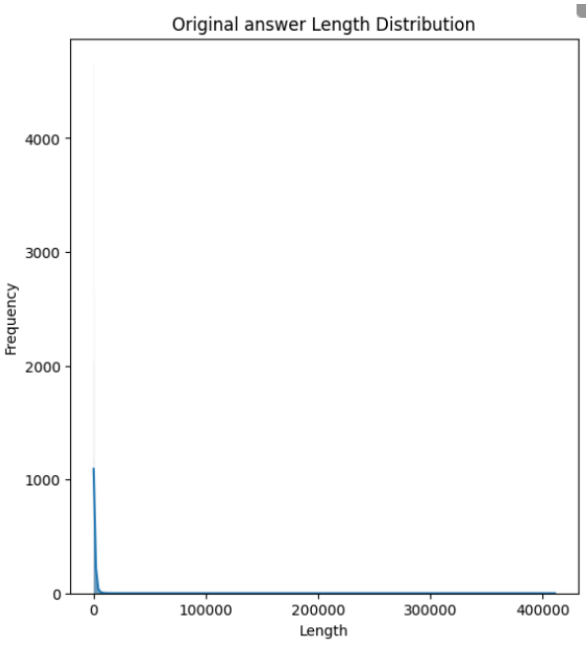


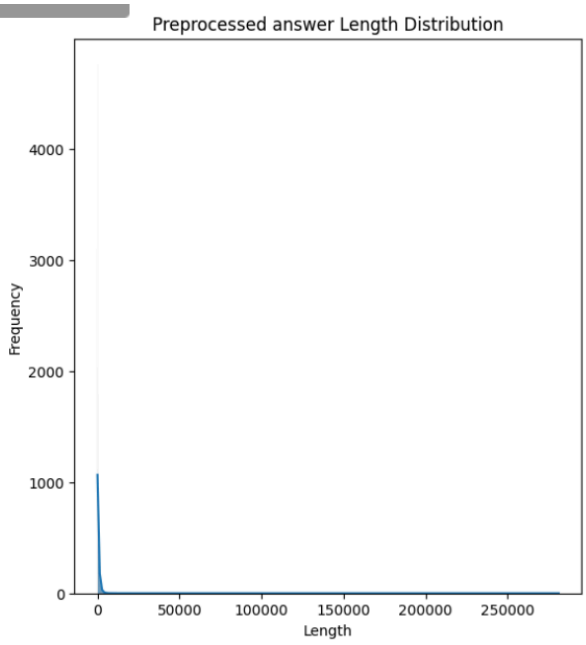
Distribution of Testing Dataset

Comparison of distributions of original and preprocessed datasets


Original question Length Distribution
Frequency
Length
0
500
1000
1500
2000
2500
3000
0
50
100
150
200
250


Preprocessed question Length Distribution
Frequency
Length
0
500
1000
1500
2000
2500
0
25
50
75
100
125
150
175
200


Original answer Length Distribution
Frequency
Length
0
1000
2000
3000
4000
0
100000
200000
300000
400000


Preprocessed answer Length Distribution
Frequency
Length
0
1000
2000
3000
4000
0
50000
100000
150000
200000
250000

Model Comparison

1. FLAN-T5

Flan-T5 results of ROUGE scores for 1 epoch on no additional preprocessing

Epoch	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Rougelsum
1	3.186700	2.975493	0.109332	0.023861	0.089627	0.099526
2	3.022900	2.934042	0.114524	0.025549	0.093881	0.104382

Flan-T5 results of ROUGE scores for 1 epoch on preprocessed dataset for comparison

Epoch	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Rougelsum
1	4.668900	4.482350	0.048693	0.012087	0.045558	0.045534

2. BART

BART results for 5 epochs for 0.01% of the dataset: After preprocessing

Epoch	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Rougelsum	Bleu	Precision	Recall	F1
1	No log	6.003388	0.069918	0.017823	0.064195	0.064698	0.009473	0.009732	0.007853	0.003966
2	6.366800	5.882290	0.068643	0.016262	0.061535	0.062297	0.002736	0.033945	0.005605	0.004766
3	6.366800	5.847336	0.076381	0.016538	0.067989	0.068209	0.002907	0.033471	0.006572	0.005172
4	5.352600	5.867768	0.075656	0.017388	0.067451	0.067289	0.002205	0.004880	0.006977	0.003990
5	5.352600	5.875725	0.077282	0.018857	0.068459	0.068821	0.001992	0.037075	0.006017	0.006036

BART results for 5 epochs for 0.01% of the dataset: No additional preprocessing

[285/285 05:40, Epoch 5/5]

Epoch	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Rougelsum	Bleu	Precision	Recall	F1
1	No log	3.312821	0.111979	0.027892	0.093121	0.101193	0.001819	0.007000	0.005891	0.005754
2	4.562900	3.241678	0.094782	0.022877	0.082504	0.088422	0.000180	0.002246	0.001887	0.001990
3	4.562900	3.218993	0.106030	0.021605	0.088603	0.096301	0.000524	0.003310	0.003931	0.003514
4	3.066400	3.246038	0.106175	0.026001	0.087058	0.094645	0.001490	0.007895	0.009587	0.008478
5	3.066400	3.245896	0.100348	0.022755	0.082588	0.089890	0.000821	0.003168	0.002385	0.002584

Additional Reports for Visualisation

Pandas Profiling Report: Original Dataset:https://github.com/ArushiGupta26/Quora_QA/blob/main/pandas_profiling_report_sample.html

Pandas Profiling Report: Preprocessed Dataset: https://github.com/ArushiGupta26/Quora_QA/blob/main/pandas_profiling_report_sample_after.html

Weights and Biases Model Performance Reports for Flan-T5 :https://github.com/ArushiGupta26/Quora_QA/blob/main/Flan-T5-Report%20_%20huggingface%20%E2%80%93%20Weights%20%26%20Biases.pdf

Insights and Recommendations:

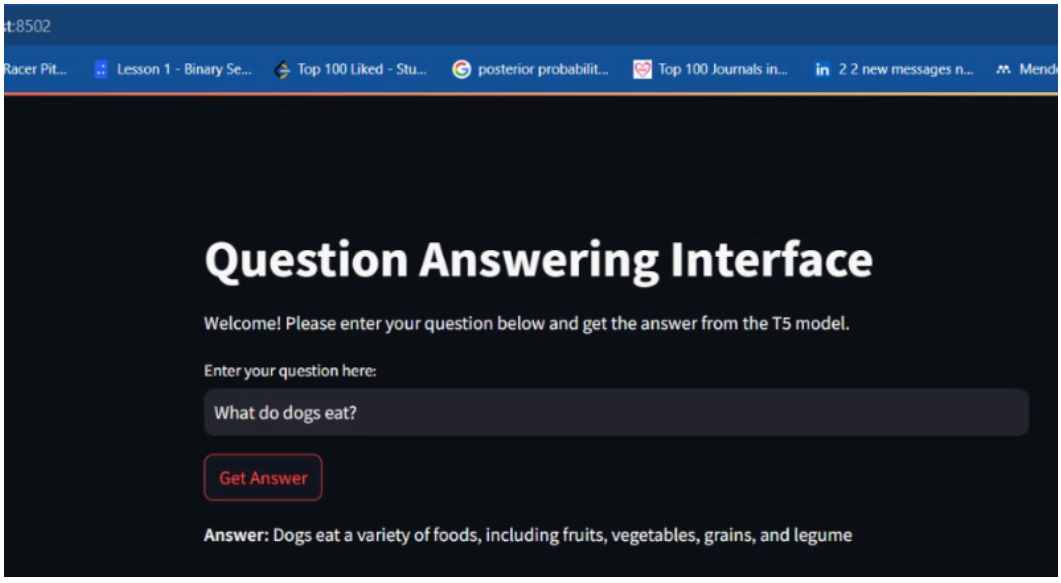
- The dataset contains characters from many languages, links, emojis, punctuation and other special characters.
- We can include BERT and GPT models even if they are not Seq2Seq or Text2Text, the data can be augmented for their use and may result in a better performance than T5 and BART
- The preprocessed dataset has reduced readability which implies that the preprocessing too heavily on the dataset can result in a degradation of performance.
- The original dataset's word cloud shows stopwords like 'the' implying their high frequency, indicating stop word removal may be a good choice.

- Since tokenization and stemming are included in respective tokenizers for the models, performing these operations separately may result in increased operational complexity and loss of information.
- K-fold cross validation can be done to improve model fitting, however it may not be necessary since training and testing sets have similar distributions
- Validation and testing datasets can be
- Models may be trained on smaller subsets due to computational limitations. Additionally smaller batch size, gradient accumulation and other optimisations may be performed to address time and computational constraints.
- The models need to be trained for more epochs to improve metrics score as demonstrated by decreasing training and validation losses
- The models may show more meaningful ROUGE, BLEU, F1-Measure, Precision and Recall metrics by training for longer.

Results

Results suggest that preprocessing heavily actually led to a degradation in the model performance, at least in the early stages. This suggests that the dataset must not be preprocessed heavily to retain the semantic meaning and readability. Given computational and time constraints, it is possible these results improve later. However, the decreasing validation and training losses demonstrate the potential State-of-the-art utility of models trained on both highly-processed and less processed data. If less computational constraints were present and models were trained for more epochs they may result in a low loss and higher metric score.

Streamlit Interface for Flan-T5



Conclusion

BART and T5 have been selected for their superior text generation capabilities and their effectiveness in closed-book question answering. BART's combination of bidirectional encoding and autoregressive decoding, along with T5's unified text-to-text framework, provides a robust foundation for generating accurate and contextually relevant answers. Their pre-training objectives and adaptability in fine-tuning further enhance their performance in closed-book scenarios. These models offer a strong choice for applications requiring sophisticated text generation and precise question answering capabilities, ensuring effective responses across various contexts.