Using Question Generation/Answering Models in a Real-Time system

Carol Sebastian
Computer Engineering
Fr.Conceicao Rodrigues College of Engineering, Bandra
Mumbai, India
carolseb80@gmail.com

Princeton Baretto
Computer Engineering
Fr.Conceicao Rodrigues College of Engineering, Bandra
Mumbai, India
princebaretto@gmail.com

Sherwin Pillai
Computer Engineering
Fr.Conceicao Rodrigues College of Engineering, Bandra
Mumbai, India
sherwinpillae@gmail.com

Supriya Kamoji
Computer Engineering
Fr.Conceicao Rodrigues College of Engineering, Bandra
Mumbai, India
supriyas@frcrce.ac.in

***Abstract*—Question Generation and Answering, being a challenging task, has gained considerable attention in the past years. Even though significant milestones are achieved, when used in a real-time system, it needs indispensable optimization. This paper proposes an approach to developing an online platform that facilitates traditional processes by introducing a virtual assistant to support educational programs by asking questions in natural language and getting an answer without reading the internal documents relevant to the problem. The system put forward is a cloud-based solution that automatically generates questions and provides sample answers from a given document(s). The entire architecture integrated into the WhatsApp interface with Twilio API's help offers a user-friendly experience.**

I. Introduction

Querying a given document or understanding a considerable record is always a tedious task. Whether it be a research paper based on any topic or a textbook in pdf format, most of the content is becoming online, which opens up a huge opportunity to use the online content for better understanding and easily accessible information. Assume we have a multi-page pdf, and we need to refer to it to answer a specific question, but we do not know which part of the pdf the answer exists, here we can leverage Deep Learning and N.L.P. techniques. Question answering is becoming an exciting part of the N.L.P. due to the advancements in deep learning models such as transformers, neural networks, Etc.; question answering is getting increasingly popular.

Open-domain question answering (Q.A.) is a task in natural language understanding that can be immensely useful for the users to understand better and find answers to their queries as soon as possible.

Another emerging topic in this field is question generation. Assume that a teacher wants to set a question paper for the students, he/she sets the questions from a particular domain, but for this, he/she needs to refer to the whole topic first and then frame questions out of it. Here, question generation using deep learning techniques can reduce the time spent reading the content and completes the task faster and efficiently.

Question generation is a task that takes the context and an answer phase, then generates a question from these inputs. This field has become tremendously popular in the academic as well as industrial sectors.

We aim to combine both Question answering and question generation techniques in one place, which would help the user easily upload the content they need for processing, and based on the models present in the system, it will give the respective output to the user.

A user interface plays a vital role for the user. For easy accessibility of the system, the whole system is based on "WhatsApp" to be used through a chat-based system that is easy to understand and easy to use.

We propose a system capable of interacting with the user via "WhatsApp" as a chat application and, based on the user's inputs, providing tasks such as question generation and question answering using the proposed models further in this paper.

The rest of this paper is organized as follows. In Section 2, we discuss the related work done on question generation and answering topics. Section 3 describes the datasets used for model training. Section 4 introduces our approach and the models used for question generation and answering, providing the system's methodology. Furthermore, Section 5 provides the results, and we conclude the paper and discuss future work.

II. Related Work

Development in the field of Automatic Question Answering Systems is excellent in recent years. Many approaches have been used to implement the following requirement. A. Agarwal[1] divides the whole system processing into three subprocesses, Entity Recognition using D.C.N. (Dynamic Concept Network), Question Analyzer, which filters relevant features using tokens, and Answer Retrieval, for extracting the answer based on the above two processes. Darshana [2], in his paper, provided a Versatile global T-max pooling and DeepLSTM for quality answer prediction. They have also used Efficient D.F.M. to forecast amicable solutions, especially D.F.M. used for the ranking cause. Monisha Devi[3] provided a Closed Domain Answering System based on predefined knowledge domain resources. Here, the query is processed based on the keywords generated by the S.E.R.(System Entity Recognizer). Emanuele Damiano[4] also used Local Proximity Prioritizer for a high-density answer matching from the query and Keywords Overlapping Module for better answer selection. Marco Pota[5] proposed a closed domain Q.A. system based on fuzzy logic and uses various components like Named entity matching, Keywords Overlapping, Proximity with Gap, Proximity Distance, Highlighted Text to L.A.T., Multiple L.A.T. discourager for answer selection. Ajitkumar Meshram Pudge gave a factoid-based Question Answering System in his paper [7]. The system uses TF-IDF Vectorizer and Cosine Similarity Finder as tools for improving system accuracy. In his paper[9], Akshay Upadhyay proposes an L.S.T.M.-based model that classifies a question into an opinionated or a polar category.

The system also uses soft computing for identifying duplicate questions for faster processing and better accuracy.

Like Question Answering, fields of Question Generating systems have significantly improved over the last few years. Yua Sun [10] papers propose a QuGan model based on Q.R.N.N. and Reinforcement Learning(R.L.) for a Tibetian Question generation. Q.R. is used as Generator to derive maximum probability text sequence and the degree of correctness of the generated data checked using L.S.T.M. The system uses Reinforcement Learning for faster sequence identifiers B.E.R.T. for system grammatical error reduction.

III. Dataset Description

Dataset used by our proposed system is SQuAD 2.0 Dataset. Stanford released it for Natural Language Processing tasks. SQuAD (Stanford Question Answering Dataset) is an open-source dataset used to train and test a model for generating and answering questions. The whole dataset consists of Wikipedia articles with crowdsourcing, where crowds were to raise questions on Wikipedia articles for which answers were also available in the same article.

The dataset also includes more than fifty thousand unanswerable questions for a better model training process and answerable questions.

IV. Methodology

The architecture of all the former Q.A. models like B.E.R.T., A.L.B.E.R.T., E.L.E.C.T.R.A., Etc. is an amplified transformer. Transformer proffers the model to take longer texts, sentences, or even paragraphs into consideration while generating the response and giving an accurate and precise answer.

Nevertheless, in a real-time system, the model's input can not be limited to a few sentences or paragraphs. Passing long, unprocessed raw data directly into any Q.A. system could result in a StackOverflowError and crash the system if the hardware requirements do not satisfy. Thus, the Longer the text, the higher is the computation power required.

Even if the hardware requirements meet, the whole operation could be pretty expensive considering the cost and time taken to carry out the process. Hence, using Question Answering/Generation models for a real-time system is not appreciative unless the input text's length is minimized.

A. Question Answering (Q.A.) Models

The architecture employed here is an efficient way of preprocessing the raw data before passing it to the Q.A. model, where the input data can be of any form (pdf, word, docs, images, txt, ppt, Xls, Etc.), but the goal remains the same, i.e., to work efficiently with almost any kind of data received from the user.

The more metadata available, the more accurate the analysis of the file's content will be. Let it be a scanned pdf or an image; architecture should still work well with the documents by carrying out Optical Character Recognition(O.C.R.), performing content analysis, and further translating the metadata. Therefore, to support maximum document types, extract the metadata from them, perform content analysis, translation, search engine indexing, Etc., the Apache toolkit, Tika, and a FOSS(free and open-source software), can be helpful.

Almost every Q.A. architecture based on transformers follows the same pipeline of breaking down the input text into paragraphs and tokenizing it. Further removes the stop words and then performs basic N.L.P. tasks (like lemmatization and stemming) and passes them to the transformer.

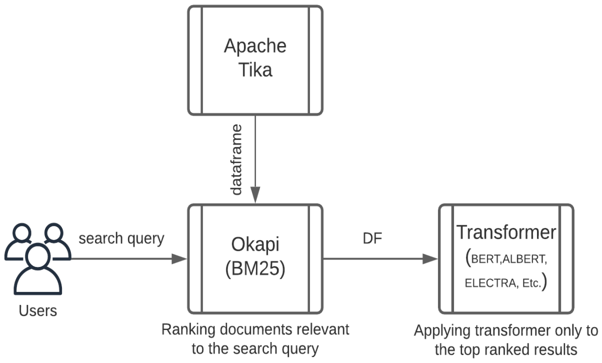

Apache Tika
dataframe
Users
search query
Okapi (BM25)
DF
Transformer (BERT,ALBERT, ELECTRA, Etc.)
Ranking documents relevant to the search query
Applying transformer only to the top ranked results

Fig. 1. *Question Answering Architectur*e

Given a document, assuming that not every sentence or paragraph is relevant to the query, not all metadata needs to follow the Q.A. pipeline. With the help of a probabilistic relevance model, the document can be filtered and eliminate a majority of the text by ranking texts/documents based on the relevance with a given search query. The Okapi(BM25), a weighting scheme framework, the best-known derivative of the probabilistic relevance model, can rank the text and efficiently query only the informative and relevant text.

A neural network-based deep learning model for performing N.L.P. tasks, trained on the S.Q.U.A.D.2.0 dataset, can accurately extract the selected content is required to answer. B.E.R.T. (Bidirectional Encoder Representations from Transformers), A.L.B.E.R.T. (Acoustic and Laryngeal Biofeedback Enhancement Real-Time), E.L.E.C.T.R.A. (Efficiently Learning an Encoder that Classifies Token Replacements Accurately), Etc. are some of the widely accepted architecture designs for the Q.A. task.

The given architecture can be built on top of any existing Q.A. models to increase its efficiency. The question answering pipeline has been outlined in figure 1.

B. Question Generation (Q.G.) Models

Many attempts to generate a question from a given context have been made, but due to the prior models' complexity, it is not considered a mainstream task as Q.A.

The straightforward Q.G. pipeline would be

1. Handling long text**:** Input to the system is not restricted just to sentences or paragraphs, but also document(s) with multiple sections are accepted. With the help of Apache Tika, the raw data can be parsed efficiently even when the input is of the type .doc, .txt, .docx, .pdf, .ppt, .xls, Etc. Even though the Transformer can handle long texts, it often throws out a StackOverFlow Error when provided with many such inputs. Such Errors can be handled by streamlining the input and forwarding them in batches (of sentences or paragraphs) to the model.

2. T5 Tokenizer: It is an encoder-decoder model that converts N.L.P. problems to text-to-text format. It helps to classify the sentence as an answer-aware sentence or not. The Transformer accepts the answer-aware ones, discarding the rest.

3. Generating Question: By providing the answer and the answer-aware text to an enhanced transformer trained on S.Q.U.A.D. 2.0, questions can be generated.

4. Grouping similar questions: When there are different contexts with similar intent, the model might generate similar questions. These edge cases are detected and grouped while compiling.

5. Question Selection: With user interaction, the questions formed can be directed to a particular domain to meet the user's requirements. The answer-aware sentences can be clustered and viewed as various domains by performing various N.L.P. tasks (like tokenizing, lemmatizing, and stemming). A set of questions is ranked and selected by performing a Cosine similarity algorithm with all the available domains and the user-selected domains.

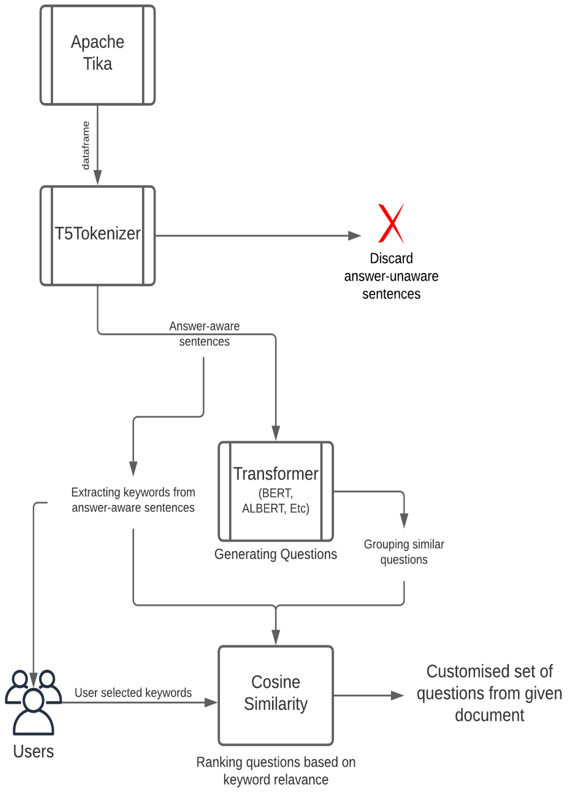

Apache Tika
dataframe
T5Tokenizer
X
Discard answer-unaware sentences
Answer-aware sentences
Transformer (BERT, ALBERT, Etc)
Extracting keywords from answer-aware sentences
Generating Questions
Grouping similar questions
Users
User selected keywords
Cosine Similarity
Customised set of questions from given document
Ranking questions based on keyword relavance

Fig. 2. *Question Generation Architecture*

Every Question generated has a corresponding context and an answer. If the user interacts with the system, these questions can be selected and ranked according to his/ her choices. Given a distinct set of all the keywords (the subject to the generated questions), if the user selects a set of keywords, then, by performing a cosine similarity, the most relevant contexts are found, and questions generated. The entire question generation pipeline has been outlined in figure 2.

C. Q.A. & Q.G. Models in a Real-Time System

To provide high speed, automatic software integration, back-ups, mobility, Etc., The proposed system is a cloud-based solution. DigitalOcean, an Infrastructure as a service (IaaS) provider, is used.

General users using the system interact with a chatbot having a WhatsApp Interface. The chatbot is integrated with a Google-owned framework, DialogFlow, with a trained intent to help the users interact in a natural language. Every text entered by the user is forwarded to the web server via Twilio API services. Twilio API acts as a mediator between the WhatsApp Interface and Q.A. /Q.G. APIs that can handle texts, images, pdfs, docs, and many more.

Reverse proxying helps to use the VMs efficiently, providing a level of abstraction and smooth network traffic control. NGINX, an open-source software, helps load balancing, caching, web serving, and reverse proxying, making the underlying APIs readily available. So, the NGINX Container wraps the WebApp containing Q.A./Q.G APIs. Further dockerizing this container.

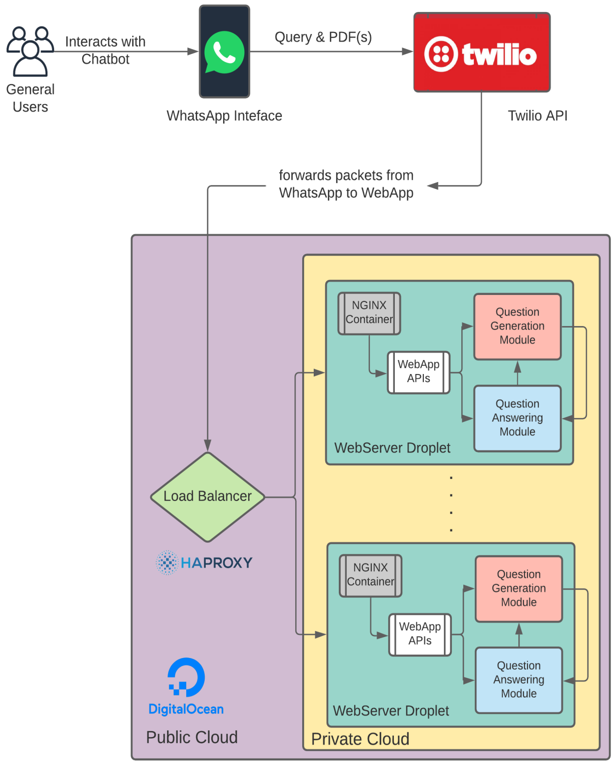


Fig. 3. Cloud Based System Architecture

Considering that not just one user is active on the system, the load at the web servers increases, and this network traffic needs to be regulated. The droplets can be horizontally scaled to increase the availability of the WebApps, and a load balancer can monitor this. HAProxy (High Availability Proxy), a load balancer, helps control the incoming requests and distribute them between the droplets in a Round-Robin fashion.

The entire cloud-based Q.A./Q.G. system architecture has been outlined in figure 3.

V. Results And Conclusion

We have used S.Q.U.A.D. 2.0 dataset for training the Question Generation as well as Question Answering models. We have used the *nlg-eval* package to deduce the results of the model. The results on S.Q.U.A.D. 2.0 dev set is as follows:

Model	BLEU-4	METEOR	ROUGE-L
t5-Question Generation	21.323	27.085	43.596
Question Answering	21.014	26.912	43.248

We have used three evaluation metrics for the overall performance of the system.

BLEU : Bilingual Evaluation Understudy evaluates the text generated for many NLP tasks and computes the scores by comparing one candidate text with it's reference.

METEOR : Metric for evaluation of translation with explicit reordering is based on the harmonic mean of unigram precision and recall.

ROUGE-L : Recall oriented understudy for gisting evaluations using LCS is used to score the text based on the longest co-occurring in sequence n-grams

Below are the comparisons of time taken on a paragraph of text.

Question Answering Model	Mean Time
Proposed Model	1.8 seconds
T-5 base model	3 seconds
T-5 small model	2.5 seconds

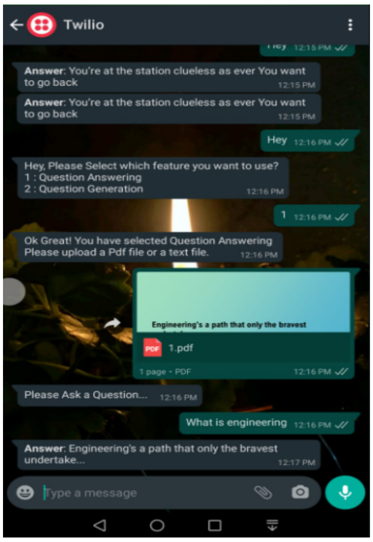
Question Generation Model	Mean Time
Proposed Model	15 seconds
T-5 base model	22 seconds
T-5 small model	18.67 seconds

The proposed models can be easily deployed and can be inferred quickly so that the user does not have to wait for the response for a long time. In addition to this, the Question answering part implements a probabilistic relevance model, which filters the paragraphs based on the relevancy and then accordingly feeds the most relevant paragraphs to the model for inference; due to this, the inference time of the model is reduced significantly.

To make the whole process faster and available for the user, we have also proposed a highly available cloud-based system design. The proposed system works on Digital Ocean Cloud following the principles of microservice architecture, as it reduces the load on a single machine and distributes the tasks among machines. The whole system can be accessed through a WhatsApp Account, making it easy to use and understand.

The Question-answering model follows a decent step-by-step approach, achieving state-of-the-art performance on both sentence-level and paragraph-level contexts. Our unique algorithm speeds up the process to get the output in the minimum time possible and provides a strong base for further research.

Figure 4,5,6 are some snippets from the WhatsApp deployed system.



Hey 12:15 PM

Answer: You're at the station clueless as ever You want to go back 12:15 PM

Answer: You're at the station clueless as ever You want to go back 12:15 PM

Hey 12:16 PM

Hey, Please Select which feature you want to use?
1 : Question Answering
2 : Question Generation 12:16 PM

1 12:16 PM

Ok Great! You have selected Question Answering
Please upload a Pdf file or a text file. 12:16 PM

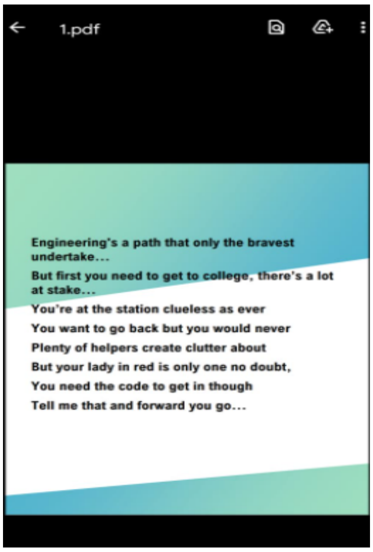
Engineering's a path that only the bravest
1.pdf
1 page • PDF 12:16 PM

Please Ask a Question... 12:16 PM

What is engineering 12:16 PM

Answer: Engineering's a path that only the bravest undertake... 12:17 PM

Fig. 4. WhatsApp Deployed Chatbot



Engineering's a path that only the bravest undertake...

But first you need to get to college, there's a lot at stake...

You're at the station clueless as ever

You want to go back but you would never

Plenty of helpers create clutter about

But your lady in red is only one no doubt,

You need the code to get in though

Tell me that and forward you go...

Fig. 5. Page 1 of the uploaded PDF in FIg. 4.

VI. Acknowledgement

Special thanks to faculty from Fr. Conceicao Rodrigues College of Engineering for the guidance and support throughout the dissertation on the Q.A/Q.G topic. We thank our families and friends who supported us in writing this paper. Their contributions to our efforts are greatly appreciated.

VII. References

[1] A. Agarwal, N. Sachdeva, R.K Yadav, V.Udandarao, V.Mittal, "EDUQA: Educational Domain Question Answering System Using Conceptual Network Mapping", SBILab, Department of ECE, IIIT-Delhi, India, Million Sparks Foundation, ICASSP, 978-1-5386-4658-8 2019 IEEE

[2] Darshana V Vekariya, Nivid R Limbasiya, "A Novel Approach for Semantic Similarity Measurement for High Quality Answer Selection in Question Answering using Deep Learning Methods", 2020 6th International Conference on Advanced Computing & Communication Systems (ICACCS), 978-1-7281-5197-7 2020 IEEE.

[3] Vignesh A, Monisha Devi, Venkateshwaran G K, Hariharan K " Semantics Enhanced Answer Selection in Closed-domain Question Answering System", 2018 International Conference on Power, Information and Communication (ICCPEIC), 978-1-5386-2447-0 2018 IEEE.

[4] Emanuele Damiano, Raffaele Spinelli, Massimo Esposito, Giuseppe De Pietro, "Semantics-Enhanced Answer Selection in Closed-domain Question Answering System", 2016 12th International Conference on Signal-Image Technology & Internet-Based Systems. DOI 10.1109/SITIS.2016 IEEE.

[5] Marco Pota, Massimo Esposito, Giuseppe De Pietro, "Learning to Rank Answers to Closed-Domain Questions by using Fuzzy Logic", the Natural Research Council of Italy. 978-1-5090-6034-4 2017 IEEE.

[6] Sweta P. Lende, Dr.M.M. Raghuwanshi "Question Answering System on Education Acts Using NLP Techniques" in the 2016 IEEE Sponsored World 45 Conference on Futuristic Trends in Research and Innovation for Social Welfare (WCFTR'16).

[7] Ajitkumar Meshram Pundge, C. Namrata Mahender " Evaluating Reasoning in Factoid based Question Answering System by Using Machine Learning Approach in Proceedings of the International Conference on Communication and Electronics Systems (ICCES 2018), IEEE Xplore Part Number:CFP18AWO-ART; ISBN:978-1-5386-4765-3.

[8] Animesh Srivastava, Shantanu Shinde, Naeem Patel, Siddhesh Despande, Anuj Dalvi, Shweta Tripathi " Questionator - Automated Question Generation using Deep Learning" in the 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE).

[9] Akshay Upadhya B, Sowmya Kamath S, Swastik Udupa " Deep Neural Network Models for Question Classification in Community Question-Answering Forums" in 10th ICCCNT 2019 July 6-8, 2019, IIT - Kanpur, IEEE 45670.

[10] Yuan Sun, Chaofan Chen, Tianci Xia and Xiaobing Zhao, "QuGAN: Quasi Generative Adversarial Network for Tibetan Question Answering Corpus Generation", Volume 7, IEEE 2019.

[11] Ying-Hong Chan, Yao-Chung Fan "A Recurrent BERT-based Model for Question Generation" in Proceedings of the Second Workshop on Machine Reading for Question Answering, pages 154–162, 2019