PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100-ft Ring Road, Bengaluru – 560 085, Karnataka, India

A Project Report
On

Text Summarization of COVID 19 articles using Question and Answering System using BERT & GPT-2

Submitted in fulfillment of the requirements for the Project phase -1

Submitted by

Venkat Krishna Pillai
SRN: PES2202101387

Abstract

On March 11, 2020, the novel Corona virus disease (COVID-19), was described as a pandemic by World Health Organization (WHO). Globally, the COVID-19 has not only affected the public health socially but also has rigorously affected economically. Substantial declines in income, increase in unemployment, and distractions in the transportation, amenities, and industrial sectors are amongst the major concerns of the pandemic disease extenuation. There are humongous amount of information available presented by different medical, research and policy making community , however it is physically challenging and time consuming for front line workers and staff to go through all the literature and help impacted groups. Taking these enormous challenges we propose a retriever-reader dual algorithmic system that answers the complex queries by searching a document of 70K corona virus-related literature made accessible through the Coronavirus Open Research Dataset Challenge (CORD-19). The retriever is composed of a BM25 retreiver capturing the documents with optimal scores. The reader which is pre-trained Bidirectional Encoder Representations from Transformers (BERT) on SQuAD dev dataset built on top of the HuggingFace BERT transformers, refines the sentences from the filtered documents, which are then passed into ranker which compares the logits scores to produce a short answer, title of the paper and source article of extraction.

Both GPT-2 and BERT had almost similar F1 scores , however BERT seems to have performed better .

Contents

1 Introduction

Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus. Most people infected with the virus experienced mild to moderate respiratory illness and recovered without requiring special treatment. However, some became seriously ill and required medical attention. Older people and those with underlying medical conditions like cardiovascular disease, diabetes, chronic respiratory disease, or cancer were impacted to develop serious illness. Anyone could get sick with COVID-19 and become seriously ill or die at any age. The best way to prevent and slow down transmission was to be well informed about the disease and how the virus spreads and help care givers and takers well informed of latest symptoms and measures.

The people on the cutting edges of the battle - medical services experts, strategy makers, clinical scientists, and so on - will require expert and specific understanding to stay aware of this literature for getting proper knowledge of the latest research findings [2]. The worldwide initiatives to find solution, identify infectious routes symptoms were almost immediate. However, disseminating the knowledge to front line workers was not so fast. COVID-19 question-answering abilities could play vital in current roles and any future pandemic situations making this abundance of information both helpful and significant [19]. Considering the amount of incorrect information disseminated in internet resulting public fear and vaccine hesitancy, the need to for to have a well-researched system becomes obvious. Taking these challenges into thought, COVID-19 search system has the option to enormously help the endeavors of scientists to viably help to battle the current and future pandemics.

The proposed system is a retriever-reader dual algorithmic approach that takes queries from the user as input and generates the most accurate responses consisting of a set answer to the query, its probability score based on the context from the scientific literature [9],[10],[11],[12],[13]. The COVID 19 QA system searches all documents corona virus-related literature made accessible through the Coronavirus Open Research Dataset Challenge (CORD-19) [25]. There are different types on QA models or also called chatbots also known as conversational agent, is a computer software capable of taking natural language input and providing a conversational output in real time [11].The human-chatbot interaction is carried over by an graphical user interface based on human chatbot interaction. Chatbots have evolved to interact via human voice as well. Such chatbots are also known as virtual assistants. Voice assistants like Apples' Siri (2010), Microsoft's Cortana (2013), Amazon's Alexa (2014) and Google's Assistant (2016) are using voice recognition powered by AI to recognize words and phrases by user and interact in personalized manner. The below figure 1 shows evolution of QA system over time.

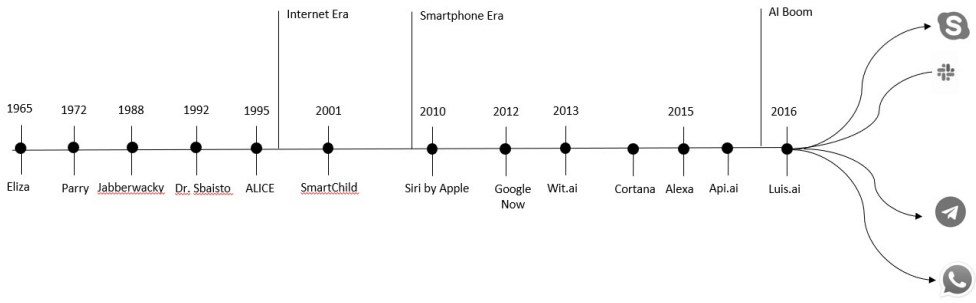


Figure 1: Evolution of ChatBots Timeline

With increased internet usage, proliferation of e-commerce platforms [29] and coupled with need for customer service 24x7 has fueled growth of chatbots over the recent years. Intelligent agents can do a variety of tasks ranging from labor work to sophisticated operations. A chatbot is a typical example of an AI system and one of the most elementary and widespread examples of intelligent Human-Computer Interaction (HCI) [3]. Authors here have discussed different type of HCI approaches.

- Anthropomorphic Approach: This approach involves designing human interface such as to produce human like characteristics.
- Cognitive Approach: These approaches used to develop a user interface that supports the end user and considers the abilities of human brain and sensory recognition.
- Empirical Approach: This approach is used for examining and comparing the usability of multi-conceptual designs
- Predictive Modelling Approach: GOMS method is used for examining and takes into consideration, user's experience in terms of time taken by a user to efficiently and effectively complete a goal. GOMS stands as g stands for goals for operators, and m for methods and s for section rules

1.1 Background

Alan Turing in 1950 proposed the Turing Test ("Can machines think?"), and it was at that time that the idea of a chatbot was popularized [22]. The first known chatbot was Eliza, developed in 1966, whose purpose was to act as a psychotherapist returning the user utterances in a question form [24]. It used simple pattern matching [5] and a template-based response mechanism. Its conversational ability was not good, but it was enough to confuse people at a time when they were not used to interacting with computers and give them the impetus to start developing other chatbots [12]. An improvement over ELIZA was a chatbot with a personality named PARRY developed in 1972 [6]. In 1995, the chatbot ALICE was developed which won the Loebner Prize, an annual Turing Test, in years 2000, 2001, and 2004. It was the first computer to gain the rank of the "most human computer" [22]. ALICE relies on a simple pattern-matching algorithm with the underlying intelligence based on the Artificial Intelligence Markup Language (AIML) [15], which makes it possible for developers to define the building blocks of the chatbot knowledge [22]. Chatbots, like SmarterChild [16] in 2001, were developed and became available through messenger applications. The next step was the creation of virtual personal assistants like Apple Siri, Microsoft Cortana, Amazon Alexa, Google Assistant and IBM Watson. There have been number of factors that have contributed to increased usage and adoption of chatbots recently.

- Growth of Internet: Internet adoption is growing at 49.6 percent, and as more people get online every day, the power of the Internet grows. Not only has the number of people using Internet gone up, but also the time spent on the Internet by everyone is on the rise. Adults spend close to 28 hours a week on average on the Internet gathering information, the Internet is estimated to have generated around 1.2 million terabytes of data (1 terabyte is 1,000 gigabytes). The year 2007 marked the emergence of Big Data, which means there is a lot of data that can be mined for information retrieval, and the tools to do so are still being actively developed by large enterprises around the globe. One of the key components for an intelligent chatbot is (www.internetworldstats.com/emarketing.htm).
- Advancement of Technology: In the past few years there has been a boon for the field of machine learning and artificial intelligence. In the early years of 2000s, the machine learning field evolved with addition of deep learning, which helps computer machines to "see" and understand things in text, images, audio, or videos. The top technology companies pushed the development of AI to leverage the power of cheap computation to solve hard problems. The transition of theoretical machine learning problems to practical implementation has helped Internet companies leverage machine learning to grow their businesses. The top technology companies in the world have all contributed in making the machine learning algorithms available in the open for anyone to use and build exciting applications.

1.2 Types of Question Answering System

QA models have made huge advancements as far as both execution and throughput [9],[10],[11]. Such upgrades can be credited to the presentation of huge scope QA datasets and deep learning models, and the recent focus of the research towards the proficiency of such models. Lately, most of the QA models are extractors and re-rankers since their attention is on a generally small-sized dataset of text such as articles, sentences, or on the other hand entries, and so on [12] -[13]. Whereas, our

proposed framework works straightforwardly on a huge corpus of literature. Our model coordinates best practices from the information retrieval with BERT and GPT-3 to create a framework focusing on an end-to-end closed domain question answering system and analysis on a standard benchmark showing improvement over the past work. Our model fine-tunes pre-trained versions of BERT with SQaUAD adequately accomplishing high scores in recognizing answers.

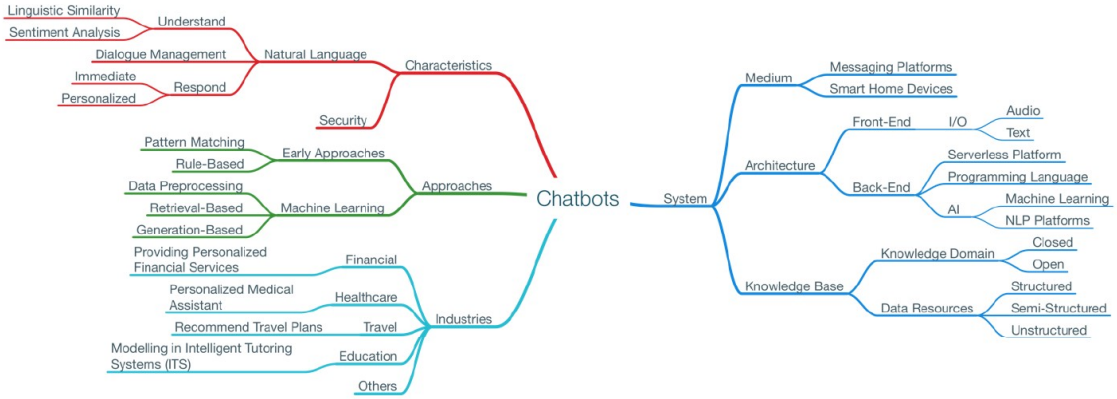


Figure 2: Caption

1.3 Problem Statement

Since the outbreak of COVID 19, there has been attempt to study and publish articles related to COVID-19 that are publicly available, to the medical community and wider society. However, the challenge of browsing through every article to look for specific content is time consuming and time intensive. Our system aims to tackle this challenge of mining numerous articles published by research bodies on COVID-19 and extracting high priority responses and summarize the most featured article. We will attempt to summarize using BERT and GPT transformer models and evaluate the best responses.

1.4 Proposed Solution

The methodology for retrieving the documents based on user query, reading through and extracting the most appropriate document and summarize happens in multiple stages.

#	Column	Dtype
0	paper_id	object
1	title	object
2	authors	object
3	affiliations	object
4	abstract	object
5	text	object
6	bibliography	object
7	raw_authors	object
8	raw_bibliography	object

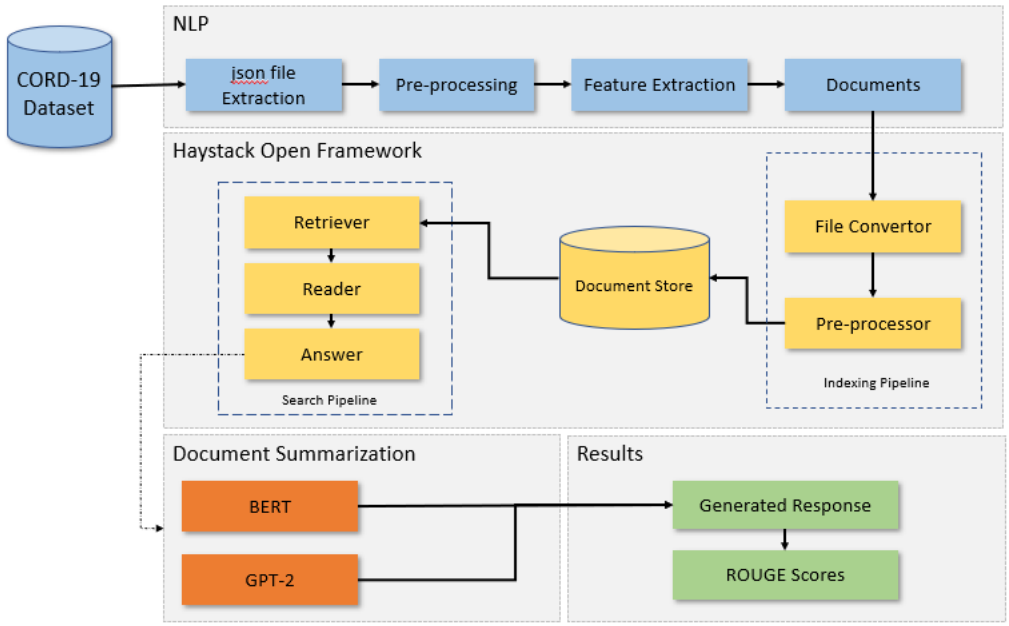


Figure 3: Overall Architecture

Figure 2 illustrates the architecture of COVID 19 QA system. It consists of 3 components.

- NLP pipeline for data preparation.
- Haystack open framework for document retriever and reader.
- Document summarizer

Based on step 1 of data extraction and cleaning using NLP, we have following 9 columns extracted. Out the 9 columns 2 of them will be extensively used in current work.

1. abstract
2. text

1.4.1 Open haystack framework

Each of the text content will be converted to document to be used in next level for open haystack framework in indexing pipeline. Haystack is open-source framework for building search systems, python framework for developing End to End question answering systems. It provides a flexible way to use the latest NLP models to solve several QA tasks in real-world settings with huge data collections. Haystack is useful for providing solutions to diverse QA tasks such as Financial Governance, Knowledge Base Search, Competitive intelligence etc. Haystack is available here as an open-source library with an extended paid enterprise version with all the bells and whistles. (Source:

https://analyticsindiamag.com/what-is-haystack-for-neural-question-answering/).

Large Neural networks, especially ones with transformer-based architectures, perform extremely well not only on Extractive Question Answering but also on Generative Question Answering(QA). But these models are computationally expensive and time-consuming. This makes them unusable in latency-sensitive applications. HayStack solves this problem by prefiltering the documents using faster but less powerful solutions. This allows the Neural Model to complete inference in a small amount of time. Typical Haystack Pipeline consists of retriever a lightweight filter that scans through all the documents in the Document store and identifies a small relevant candidate set of documents. Retrievers can be models using both sparse or dense methods. One very efficient retriever model is ElasticSearch.It is a proper sparse indexing-based search engine, in the current scenario to keep things simple we will be using Inmemoryserarch.

Haystack works are 3 levels

1. Nodes
   Haystack offers different nodes that perform different kinds of text processing. These are often powered by the latest transformer models and code-wise they are Python classes whose methods can be called directly.

2. Pipelines
   Haystack is built on the idea that great systems are more than the sum of their parts. By combining the NLP power of different nodes, users can create powerful and customizable systems. The pipeline is the key to making this modular approach work. When adding nodes to a pipeline, the user can define how data flows through the system and which nodes perform their processing step when. On top of simplifying data flow logic, this also allows for complex routing options such as those involving decision nodes.

   - Readers, also known as Closed-Domain Question Answering systems in Machine Learning speak, are powerful models that do close analysis of documents and perform the core task of question answering. The Readers in Haystack are trained from the latest transformer based language models and can be significantly sped up using GPU acceleration.
   - Retriever assists the Reader by acting as a lightweight filter that reduces the number of documents that the Reader has to process. It does this by scanning through all documents in the database and quickly identifying the relevant and dismissing the irrelevant. In the end, only a small set of candidate documents are passed on to the Reader

3. REST API
   In order to deploy a search system, you will need more than just a Python script. Rather you need a service that can stay on, handle requests as they come in and also be callable by many different applications. For this, Haystack comes with a REST API that is designed to work in production environments

1.4.2 Reader

The Reader takes a question and a set of Documents as input and returns an Answer by selecting a text span within the Documents. Readers use models to perform QA. Within the Haystack question answering (QA) pipeline, the reader object is the interface to a Transformer-based language model. This can be a publicly available model, your own trainable model, or a combination of both — where you fine-tune an existing model on your own dataset. Language models parse both query and answer candidates, and return the most likely answer spans. Because of their ingenious Transformer architecture and the large amounts of data they see during training, they have a good understanding of the syntax and semantics of natural language. But they are also computationally expensive and much slower than the other components of the Haystack pipeline. The reader also handles the details around making documents fit with the hardware. Thereby, we as the system users don't have to worry about the internal representation of our queries — for all we know, we feed a natural language query to the model and it returns answer spans from our document collection, ranked by a matching score. Haystack readers allow you to load any one of over 11,000 trained models from the Hugging Face Model Hub. Performance Speed Trade off There are many different language models based on

1. How fast do I need my model to be?

2. How much computational power do I have?

Usually, these two requirements influence one another. For instance, while most language models are so big that they require at least one GPU to run. there are lightweight models like MiniLM(deepset//minilm-uncased-squad2) that will work on just the CPU of your laptop. However, working with this model will likely require a tradeoff in terms of performance. On the other hand, a SOTA model like the large ALBERT XXL ahotrod//albert xxlargev1 squad2 512) will show excellent results, but at the expense of being slower.

1.4.3 Retriever

The Retriever performs document retrieval by sweeping through a DocumentStore and returning a set of candidate Documents that are relevant to the query. See what Retrievers are available and how to choose the best one for the use case. The Retriever takes a query as input and checks it against the Documents contained in the DocumentStore. It scores each Document for its relevance to the query and returns the top candidates. In current scenario we plan to use, Okapi BM25 (BM is an abbreviation of best matching) is a ranking function used by search engines to estimate the relevance of documents to a given search query. It is based on the probabilistic retrieval framework developed in the 1970s and 1980s by Stephen E. Robertson, Karen Sp arck Jones, and others. The name of the actual ranking function is BM25. The fuller name, Okapi BM25, includes the name of the first system to use it, which was the Okapi information retrieval system, implemented at London's City University in the 1980s and 1990s. BM25 and its newer variants, e.g. BM25F (a version of BM25 that can take document structure and anchor text into account), represent TF-IDF-like retrieval functions used in document retrieval. So far the algorithm have been implemented for Okapi BM25, BM25L, BM25+, BM25-Adpt and BM25T. BM25 is a variant of TF-IDF, commonly used for text retrieval. It improves upon its predecessor in two main aspects:

- It saturates tf after a set number of occurrences of the given term in the document
- It normalises by document length so that short documents are favoured over long documents

If they have the same amount of word overlap with the query BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of their proximity within the document. It is a family of scoring functions with slightly different components and parameters. One of the most prominent instantiating of the function is as follows.

$$score(D,Q) = \sum_{i=1}^{\infty} \Sigma IDF(q_i).\frac{f(q_i,D).(k_i+1)}{f(q_i,D)+k_i.(1-b+b.\frac{|D|}{avgd_i})} \tag{1}$$

where
f(qi,D) is the number of times that qi occurs in document D
D— is the length of documents D in words, and avgdl is the average document length in the text collection from which documents are drawn.
k1 and **b** are free parameters, usually chosen, in absence of advanced optimization. IDF(qi) is the IDF weight of the query term qi. it is usually computed as -

$$IDF(q_i) = \ln(\frac{(N-n(q_i)+0.5}{n(q_i} + 0.5) + 1 \tag{2}$$

where
N is the total number of documents in collection and n(qi) is the number of documents containing qi.
tf is how many times words in the query occur in that document.
idf is the inverse of the fraction of documents containing the word.

The next step in the process is to identify most relevant document from response and summarize the essence of the research material. Manual Summarization methods were predominant since the 19th

century, which consumed significant amount of time in human labour. M.I.T started this trend where they uploaded a huge chunk of lectures, with each lecture containing a summary along with transcripts which will be handy in keyword searching. This approach was useful when the content was limited but in today's world there are terabytes of data, so this approach will not be of much help. In 20th century, we see a huge increase in data generation every day, and the extractive summarization failed due to its poor quality. To overcome this issue, researchers tried to put rhetorical information in the content, and tried to see if summarization context can be improved. The performance gain was better, but the quality of context was sub-par. It proved that the technology had mettle but required more research in this area.

1.4.4 BERT

Bidirectional Encoder Representation of Transformers (BERT) is commonly used nowadays for text summarization. In today's world, huge amount of data gets generated every minute, be it from social media, organizations, log files etc. to properly get the context of terabytes of data, we would need to summarize the data. Till early 2005, manual summarization was common. With increase in volume and variety of data, researchers started looking at automatic summarization. In 2017, Vaswani introduced a model that had attention mechanism as well as feed forward implementation and called it "Transformer" [1]. BERT was developed by Google in 2018 on top of Transformers and is a popular model for text summarization at present. As of today, there are predominantly two summarization types. 1) Extraction of key phrases or sentences from a large text is called extractive summarization, where classification method is performed on phrases or sentences to check whether any meaningful words can be extracted. 2) Paraphrasing of a large sentence, keeping its content meaningful is called abstractive summarization, wherein model tries to encode the input sentence in the first go, followed by decoding it to output text, to generate meaningful context. Abstractive summarization is harder compared to extractive summarization as this model needs to generate human readable content from hidden representations.

Methodology

Being trained as a masked model the output vectors are tokened instead of sentences. Unlike other extractive summarizers it makes use of embeddings for indicating different sentences and it has only two labels namely sentence A and sentence B rather than multiple sentences. These embeddings are modified accordingly to generate required summaries

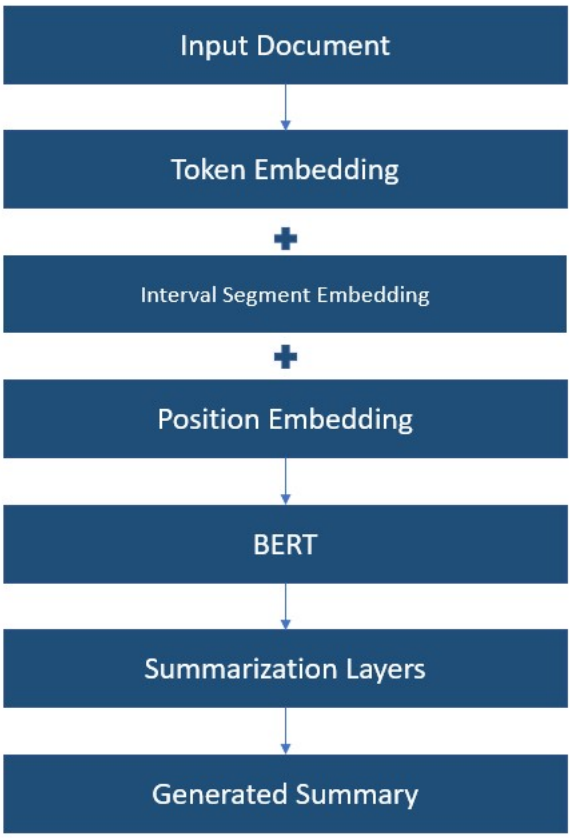


Figure 4: Methodology

The BERT model is modified to generate sentence embeddings for multiple sentences. This is done by inserting [CLS] token before the start of the first sentence. The output is then a sentence vector for each sentence. The sentence vectors are then passed through multiple layers that make it easy to capture document level features. The final summary prediction is compared to ground truth and the loss is used to train both the summarization layers and the BERT model.

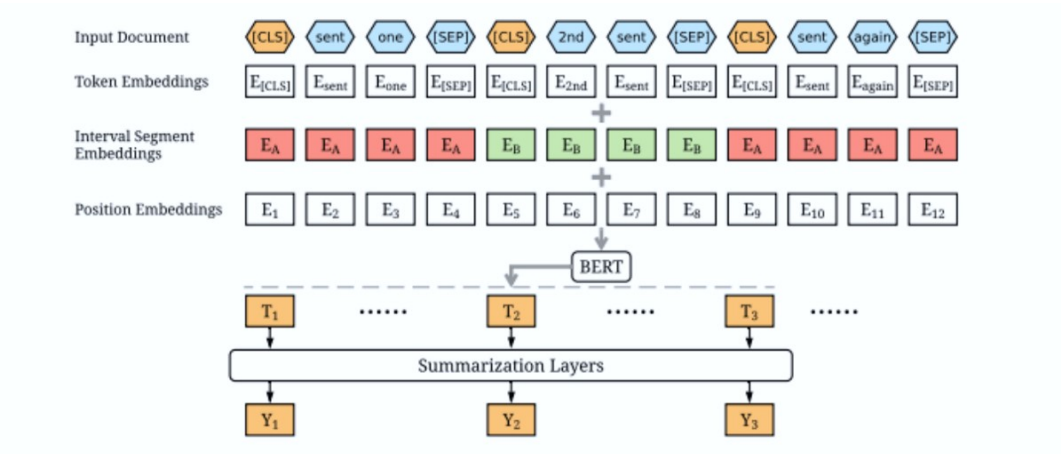


Figure 5: BERT Architecture

1.4.5 GPT-2

Generative Pre-trained Transformer 2 (GPT-2) is an open-source artificial intelligence created by OpenAI .GPT-2 translates text, answers questions, summarizes passages and generates text output on a level that seem like human generated ,It is a general-purpose learner; it was not specifically trained to do any of these tasks, and its ability to perform them is an extension of its general ability to accurately synthesize the next item in an arbitrary sequence.GPT-2 was created as a "direct scale-up" of OpenAI's 2018 GPT model,with a ten-fold increase in both its parameter count and the size of its training dataset.

GPT-2 has a generative pre-trained transformer architecture which implements a deep neural network, specifically a transformer model,[3] which uses attention in place of previous recurrence- and convolution-based architectures. ttention mechanisms allow the model to selectively focus on segments of input text it predicts to be the most relevant. This model allows for greatly increased parallelization, and outperforms previous benchmarks for RNN/CNN/LSTM-based models.

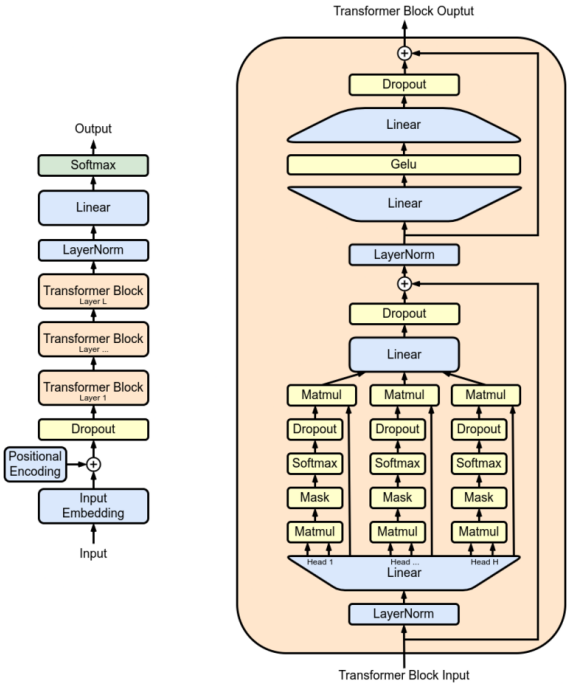


Figure 6: GPT Architecture, , Source : https://en.wikipedia.org/wiki/GPT-2

GPT's architecture itself was a twelve-layer decoder-only transformer, using twelve masked self-attention heads, with 64 dimensional states each (for a total of 768). Rather than simple stochastic gradient descent, the Adam optimization algorithm was used; the learning rate was increased linearly from zero over the first 2,000 updates, to a maximum of 2.5 x 10^{-4}, and annealed to 0 using a cosine schedule.

While GPT's fine-tuning was adapted to specific tasks, its pre-training was not; to perform the various tasks, minimal changes were performed to its underlying task-agnostic model architecture. Despite this, GPT still improved on previous benchmarks in several language processing tasks, outperforming discriminatively-trained models with task-oriented architectures on a number of diverse tasks.

Corpus , The unsupervised pre-training was performed using BooksCorpus,[57] a dataset of over 7,000 unpublished fiction books from various genres; this dataset was chosen in part because its long passages of continuous text conditioned the model to handle long-range information. Other available datasets, while larger, were rejected on the basis that they lacked this long-range structure (being "shuffled" at a sentence level). The ftfy library was used to clean the BooksCorpus text (standardize punctuation and whitespace); it was tokenized using spaCy.

Performance wise , Due to the broadness of its dataset, and the broadness of its approach, GPT-2 became capable of performing a diverse range of tasks beyond simple text generation: answering questions, summarizing, and even translating between languages in a variety of specific domains, without being instructed in anything beyond how to predict the next word in a sequence.

One example of generalized learning is GPT-2's ability to perform machine translation between French and English, for which task GPT-2's performance was assessed using WMT-14 translation tasks. GPT-2's training corpus included virtually no French text; non-English text was deliberately removed while cleaning the dataset prior to training, and as a consequence, only 10MB of French of the remaining 40,000MB was available for the model to learn from (mostly from foreign-language quotations in English posts and articles). Despite this, GPT-2 achieved 5 BLEU on the WMT-14 English-to-French test set (slightly below the score of a translation via word-for-word substitution). It was also able to outperform several contemporary (2017) unsupervised machine translation baselines on the French-to-English test set, where GPT-2 achieved 11.5 BLEU. This remained below the highest-performing contemporary unsupervised approach (2019), which had achieved 33.5 BLEU. However, other models used large amounts of French text to achieve these results; GPT-2 was estimated to have used a monolingual French corpus approximately 1/500 the size of comparable approaches.

While GPT-2's ability to generate plausible passages of natural language text were generally remarked on positively, its shortcomings were noted as well, especially when generating texts longer than a couple paragraphs. GPT-2 deployment is resource-intensive; the full version of the model is larger than five gigabytes, making it difficult to embed locally into applications, and consumes large amounts of RAM. In addition, performing a single prediction "can occupy a CPU at 100% utilization for several minutes", and even with GPU processing, "a single prediction can take seconds". To alleviate these issues, the company Hugging Face created DistilGPT2, using knowledge distillation to produce a smaller model that "scores a few points lower on some quality benchmarks", but is "33% smaller and twice as fast".

#	**architecture**	parameter count	training data
GPT-1	12-level, 12-headed Transformer decoder (no encoder), followed by linear-softmax.	0.12 billion	Book Corpus 4.5 GB of text, from 7000 unpublished books of various genres.
GPT-2	GPT-1, but with modified normalization	1.5 billion	WebText: 40 GB of text, 8 million documents, from 45 million webpages upvoted on Reddit.
GPT-3	GPT-2, but with modification to allow larger scaling.	175 billion	570 GB plaintext, 0.4 trillion tokens. Mostly CommonCrawl, WebText, English Wikipedia, and two books corpora (Books1 and Books2).

2 Literature Survey

2.1 Question Answering System

R Vaishya and Kricka have noted that the emerging technologies are set to play an important role in worlds response to the COVID-19 pandemic and similar situations. From Artificial intelligence-powered search tools and contact tracing based on mobile communication technology. the increasing awareness and use of these tools will be important in future research into the disease [13]. The COVID-19 pandemic has created unprecedented challenges for the medical and clinical diagnostic community. The fight against COVID-19 is being supported by a number of databases and artificial intelligence (AI)-based initiatives aimed at assessing dissemination of the disease [1], aiding in detection and diagnosis, minimizing the spread of the disease, and facilitating and accelerating research globally [23].

Broth Andreas et al. [7] proposed a Question- Answering (Q/A) system over Knowledge Bases(KB), which overwhelms impediments which were endured—namely, porting KBs language-agonistic systems. Yu Wenhao develop TransTQA, which is a novel system that offers automatic responses by retriev-

ing proper answers based on correctly answered similar questions in the past. TransTQA is built upon a Siamese ALBERT network, which enables it to respond quickly and accurately. Furthermore, TransTQA adopts a standard deep transfer learning strategy to improve its capability of supporting multiple technical domains. [26].

Abacha Ben Asma et al. [1] presented a Medical Visual Question Answering task (VQA-Med) focusing primarily on 4 categories of clinical questions: Modality, Plane, Organ System, and Abnormalities. The first three tasks are classified as: classification task and the fourth can be put with answer generation. These categories are designed with different degrees of difficulty leveraging both classification and text generation approaches. All of these questions are solved through images only, and do not require any additional domain-based context for generating answers, and were evaluated on Accuracy and BLEU score.

Nogueira Rodrigo et al. [17] re-implemented the query-based passage re-ranking. Trained on state of the art TRECCAR dataset, it beats out the previous top entry by 27MSMARCO passage retrieval task. The researcher's pipeline can be divided into 3 stages: first being retrieving all conceivable documents pertinent to the given query, using archetype operations such as BM25. The second stage implicates passage re-ranking by computationally-accelerated technique. The final stage being top tenor fifty of these documents being responses to the user's question. Using the BERT LARGE Model as binary classification model and [CLS] vector as input to a single-layer neural network to obtain the probability of being germane.

Rajpurkar, Pranav and team et al. [18] presented the Stanford Question Answering Dataset (SQuAD), a new reading comprehension dataset consisting of 100,000+ questions posed by crowd workers on a set of Wikipedia articles, where the answer to each question is a segment of text from the corresponding reading passage and analyzed the dataset to understand the types of reasoning required to answer the questions, leaning heavily on dependency and constituency trees. We build a strong logistic regression model, which achieves an F1 score of 51.0%, a significant improvement over a simple baseline (20%). However, human performance (86.8%) is much higher, indicating that the dataset presents a good challenge problem for future research.

Esteva et al. [8] implemented a retriever-ranker semantic search engine devised to handle complex queries regarding COVID-19. The retriever is composed of a Siamese-BERT (SBERT) as an encoder, with a TF-IDF vectorizer and reciprocal-rank fused with a BM25 vectorizer, as a ranking function to estimate the relevance of documents in a given search query. A query is given for which each document is provided with a retrieval score. The ranker comprises a multi-hop question- answering (QA) module that, alongside a multi-paragraph abstractive summarizer, adjusts retriever scores. The retrieved set of documents are run through a question answering module and an abstractive summarizer in the ranking. Based on this ranking, the documents are ranked by a weighted combination of their answer match, retrieval scores, and summarization match.

Qiao Jin, Bhuwan Dhingra [10] introduce PubMedQA, a novel biomedical question answering (QA) dataset collected from PubMed abstracts. The task of PubMedQA is to answer research questions with yes/no/maybe (e.g.: Do pre operative statins reduce atrial fibrillation after coronary artery bypass grafting?) using the corresponding abstracts. PubMedQA has 1k expert-annotated, 61.2k unlabeled and 211.3k artificially generated QA instances. Each PubMedQA instance is composed of (1) a question which is either an existing research article title or derived from one, (2) a context which is the corresponding abstract without its conclusion, (3) a long answer, which is the conclusion of the abstract and, presumably, answers the research question, and (4) a yes/no/maybe answer which summarizes the conclusion. PubMedQA is the first QA dataset where reasoning over biomedical research texts, especially their quantitative contents, is required to answer the questions. Our best performing model, multi-phase fine-tuning of BioBERT with long answer bag-of-word statistics as additional supervision, achieves 68.1% accuracy, compared to single human performance of 78.0% accuracy and majority-baseline of 55.2% accuracy, leaving much room for improvement.

In Abacha [4] he tried to solve one of the challenges in large-scale information retrieval (IR) is

developing fine-grained and domain-specific methods to answer natural language questions. Despite the availability of numerous sources and datasets for answer retrieval, Question Answering (QA) remains a challenging problem due to the difficulty of the question understanding and answer extraction tasks. One of the promising tracks investigated in QA is mapping new questions to formerly answered questions that are "similar".

Raphel Tang present CovidQA, the beginnings of a question answering dataset specifically designed for COVID-19, built by hand from knowledge gathered from Kaggle's COVID-19 Open Research Dataset Challenge. To our knowledge, this is the first publicly available resource of its type, and intended as a stopgap measure for guiding research until more substantial evaluation resources become available. While this dataset, comprising 124 question–article pairs as of the present version 0.1 release, does not have sufficient examples for supervised machine learning, we believe that it can be helpful for evaluating the zero-shot or transfer capabilities of existing models on topics specifically related to COVID-19. This paper describes our methodology for constructing the dataset and presents the effectiveness of a number of baselines, including term-based techniques and various transformer-based models [21].

Jimmy Lim and team [14] conducted a user study to investigate this area and discovered that, overall, users prefer paragraph-sized chunks of text over just an exact phrase as the answer to their questions. Furthermore, users generally prefer answers embedded in context, regardless of the perceived reliability of the source documents. When researching a topic, increasing the amount of text returned to users significantly decreases the number of queries that they pose to the system, suggesting that users utilize supporting text to answer related questions. We believe that these results can serve to guide future developments in question answering interfaces. Despite significant advances in the underlying technology of question answering systems, the problem of designing effective user interfaces has largely been unexplored.

The architecture proposed for the question-answering (QA) module, CAiRE-COVID by Su et al. [20], consists of three different modules:

1. Document Retriever: It preprocesses the user's query the most relevant n number of publications. It paraphrases long, complex queries to simple queries that are easier to comprehend by the system. These queries are run through the IR module, which returns paragraphs from the highest matching score.

2. Relevant Snipper Selector: The most relevant parts of paragraphs are highlighted and re-ranked based on the scores.

3. Multi- Document Summarizer: It returns an abstractive summary by summarizing the top relevant paragraphs. This QA module is the perfect example of the extractor and re-ranked since their attention is on a small-sized dataset such as sentences or articles.

Many models have been proposed over time to solve this problem of question-answering from a large corpus many of such have been focused domain but the final capturing capabilities of models are had proven ineffective when comes to a large corpus whereas our proposed model phase by phase increases the answering capability thus increasing the chances of the final answer being closer to ground answer as our model employs the retriever-reader dual algorithmic system. Whereas, coming to the deep learning models their performance is limited when capturing the embedding representations of question-answer pairs which generalize the model with one neural network which captures one-side features. Therefore, it's necessary for the model to capture complex all-side features of QA-pair. Existing approaches deploys the heuristics on coming up with the possible answer whereas the proposed model is assessing its answering capabilities against the ground truth sentence in the ranker phase. The proposed architectures thus try to deal with such shortcomings which are discussed in the further sections.

2.2 Document Summarization

2.3 Classic summarization approaches

Automatic text summarization is a complex set of procedures that focuses on reducing large phrases to smaller size while maintaining the output context. The two major types are: extractive summarization, which involves selecting and combining meaningful context from the input, while abstractive summarization involves generating abstract to convey useful context from input phrase. They have their own relevance as well as cons, and the selection depends on the specific problem and the level of context relevance required during summarization. For abstractive summarization, language processing models are majorly used with the combined contribution leveraging models of machine learning and deep learning.

Research in automatic-text-summarization has shown tremendous improvement recently along with newer discoveries. This is due to the improvements in language processing and prior-trained models such as BERT and Open AI GPT-2 being readily available [61]. These models are trained on vast text datasets such as the Wikipedia corpus, and can perform better when tested with large set of NLP tasks, including machine translation, question and answering, text classification, etc. One example of a summarization model that leverages the pre-trained BERT model is BERTSUM,[60] (Lapata, 2019) that has been trained on a news dataset and thus forms part of extractive BERT variant.

However, since BERT has not been designed for language generation tasks, it has limited scope in abstractive summarization. We noticed that the most popular models for abstractive summarization are those that leverages the transformer-encoder-decoder architecture like sequence-to-sequence models. Nowadays, unified text-to-text is majorly used by researchers to train a single huge language model on parallel NLP processes at a single go, [59] (Colin Raffel, 2019) allowing it to map input text to output text for different tasks. Due to this, we suggest fine-tuning pre-trained Open AI GPT-2 model to extract mapping from input keywords to an output text, thus abstractly generating a summary.

Based upon size of input source documents that are used to generate a summary, summarization can be divided in two types:

1. **Single Document**: Single document text summarization is automatic summarization of information a single document (Garner, 1982).
2. **Multiple Document**: Multi-document text summarization is an automatic summarization of information from multiple document (Ferreira et al., 2014).

Multi-document summarization is important where we must put different types of opinions together, and each idea is written with multiple perspectives within a single document. Single document text summarization is easy to implement, but multi-document summarization is a complex task. Redundancy is one of the biggest problems in summarizing multiple documents. Carbonell Goldstein (1998) has given MMR (Maximal Marginal Relevance) approach, which helps to reduce redundancy. Another main problem for multi-document summarization is heterogeneity within a large set of documents.

It is very complex to summarize multiple documents with extractive methods where there are so many conflicts and biases in the real world. Here for multiple documents, abstractive summarization performs far better. However, multi-document summarization also brings issues like redundancy in output summary while working with a huge number of documents. Single document text summarization is used in a limited field like reading the given comprehension and giving an appropriate title or summary. In contrast, multi-document text summarization can be used in the field of news summarization from different sites, customer's product reviews from different vendors, QA systems and many more.

Or once can bifurcate based on summarization outcome

1. **Abstractive Summarization**: Abstractive summaries need to identify the key points and then add a generative element. Finally, mixed strategies need to combine these elements and provide

a mechanism to decide when each mode should be used. Abstractive summarization, on the other hand, tries to guess the meaning of the whole text and presents the meaning to you. It creates words and phrases, puts them together in a meaningful way, and along with that, adds the most important facts found in the text. This way, abstractive summarization techniques are more complex than extractive summarization techniques and are also computationally more expensive.

2. **Extractive Summarization**: Extractive strategies are set up as binary classification problems where the goal is to identify the article sentences belonging to the summary. Extractive summarization methods work just like that. It takes the text, ranks all the sentences according to the understanding and relevance of the text, and presents you with the most important sentences. This method does not create new words or phrases, it just takes the already existing words and phrases and presents only that. You can imagine this as taking a page of text and marking the most important sentences using a highlighter.

There can be multiple levels of classification as seen below.

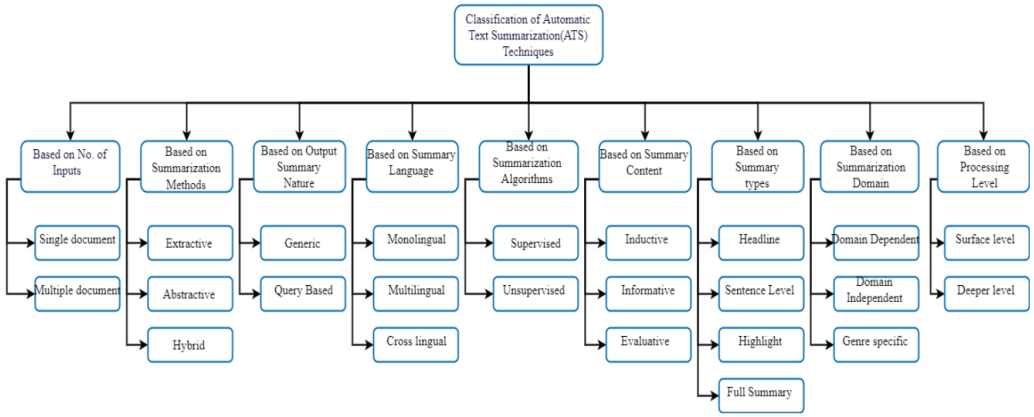


Figure 7: ATS Classification : Source: Divakar Yadav1 ,Jalpa Desai, Arun Kumar Yadav2 : Automatic Text Summarization Methods: A Comprehensive Review

2.4 Classic Question Answering approaches

Question Answering (QA) has been a challenging task in natural language understanding. The key components in QA require the capability of understanding the question and the context in which the question is generated. QA has been deemed challenging due to dynamic nature of natural languages commonly used This has resulted in the application of data driven methods in question answering. The idea is to allow the data, instead of the methods, do most of the work in question answering. This is due to a large number of text repositories that is available.

Rule-based approach was one of the initially used most prominent methods for QA systems. These systems utilized rules devised from grammatical semantics to determine the correct answer for a given question and guided accordingly. These rules are usually handcrafted and heuristic, relying on lexical and semantic hints on context. These rules exploit predefined patterns that classify questions based on the answer type. These grammatical rules represent the context in the form of decision trees and this was used to find the path that leads to the correct answer. A major drawback of rule-based question answering systems was that the heuristic rules needed to be manually created and managed. To devise these rules an in-depth knowledge of the semantics of a language was a requirement.

With the rapid growth of text material available online the importance of statistical approaches for QA has also increased. These approaches lean on predicting answers based on data. As these methods are capable of addressing the heterogeneity of data and free from structured query languages they have been adapted to various stages of QA. Statistical approaches require the formation of a hypothesis before proceeding to build the model. This hypothesis sets the tone for the creation of the model. With

the advancements in machine learning systems gained the capability to navigate the direction that the data dictates [7]. These inducted the self-learning capability to the QA systems. These systems are capable of building a knowledge base (taxonomy) from the training data it is provided and then use it to answer the actual questions. This brought a level of independence to the systems that were not quite there in rule-based or statistical approaches. Furthermore, as these systems are capable of optimizing itself over time it became one of the most lucrative approaches for QA [8].

Induction of Neural Networks for QA systems opened up a plethora of possibilities. Conventional machine-learning techniques were limited in their ability to process natural data in their raw form. For decades, constructing a machine learning system required careful engineering and considerable domain expertise to design a feature extractor that transformed the raw data into a suitable internal representation from which the learning subsystem, often a classifier, could detect or classify patterns in the input [38]. Deep learning methods are representation learning methods that allow a machine to be fed with raw data and to automatically discover the representations needed for detection, prediction or classification [38]. These are with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level [38]. With the composition of enough such transformations, very complex functions can be learned. For classification tasks, higher layers of representation amplify aspects of the input that are important for discrimination and suppress irrelevant variations [38].

Networks such as Dynamic Memory Networks [9], Reinforced-Memory Networks currently provide the state-of-the-art results in Question Answering bringing Artificial Intelligence closer to human perception.

- **RULE-BASED APPROACH**
  The initial question answering systems were logical representations of decision trees. The decision trees were linguistic structures that mirrored the way humans understand text based on grammatical rules. At the very beginning, all these rules were custom made for QA. As these systems relied on the constant extension of functionality by the addition of rules that opened up new paths in the decision tree. QA systems based on such decision trees used lexical and semantic heuristics to see find evidence whether a sentence contains an answer to a question. These systems require rule sets that define paths for questions based on the question type. The path taken by the answer extraction process for "where is Chikpet" would be different from "Who signature are on the one-Rupee note". To improve the answer matching syntactic analysis, morphological analysis, Part-Of-Speech tagging and Named Entity Recognition were later incorporated to these systems. Though these systems were successful initiatives to the Question Answering domain they were not without flaws. Extensions to these systems required new rules to be introduced to the system which is cumbersome and makes these systems not suited for domains with highly volatile data.
- **STATISTICAL APPROACH**
  In the present research state, quick evolution in available online text repositories and web data has amplified the prominence of statistical approaches. These approaches put forward such techniques, which cannot only deal with the very large amount of data but their diverseness as well [6]. Additionally, statistical approaches are also independent of structured query languages and can formulate queries in natural language form. These approaches basically require a satisfactory amount of data for precise statistical learning but once properly learned, produce better results than other state-of-art approaches. Furthermore, the learned statistical program or method can be easily customized to a new domain being independent of any language form. In general, statistical techniques have been so far successfully applied to the different stages of a QA system. Support vector machine (SVM) classifiers, Bayesian classifiers, Maximum entropy models are some techniques that have been used for question classification purpose. These statistical measures analyze questions for making a prediction about users' expected answer type. These models are trained on a corpus of questions or documents that have been annotated with the particular mentioned categories in the system [6]. One of the pioneer works based on the statistical model was IBM's statistical QA [10] system. This system utilized maximum entropy model

for question/ answer classification based on various N-gram or bag of words features. And in IBM's statistical QA TREC-11 [11] they incorporated a novel extension of statistical machine translation. In this paper they have represented their model as p(c—a, q) which attempts to measure the c, 'correctness', of the answer and question And the paper introduced another hidden variable or the model e which represents the class of the answer. the new model is represented as

$$p(c|q,a) = P.e.p(c,e|q,a) = P.e.p(C|e,q,a)p(e|q,a) \tag{3}$$

Stochastic approaches have brought about a significant improvement in POS tagging. This approach suggested a stochastic model that will be used in a statistical POS tagger to decide on the best suited tag for a particular word. This system utilizes the statistical model for handling disambiguation, decoding and smoothing of unknown words. In Order to handle these three properties, they used following techniques disambiguation - Hidden Markov Model (HMM), decoding - Viterbi Algorithm and for Smoothing - Linear Interpolation [32]. The tag which provides maximum of transition probabilities in combination with the previous two tags is selected as the tag for previously unseen tags.

- **MACHINE LEARNING APPROACH**
  With the introduction of machine learning to the QA domain algorithms that can learn to understand linguistic features without explicitly being told to. Statistical methods paved the path for this approach that enables the system to analyze an annotated corpus (training set) and then build a knowledge base. Context is processed usually by Named Entity Recognition techniques that act as the classifier to build a taxonomy [40]. This taxonomy then acts as the knowledge base. The questions would also be subjected to the same process. The major advantage machine learning brings to the table is its learnability. This makes the system highly scalable as long as there is enough training data. As most of the fields with a requirement for QA systems such as Journalism and Legal have a penchant for having large data machine learning approach has become quite popular in Question Answering. These systems regularly rake in state of the art results because of the speed and coverage factors that benefit because of the extensibility of this approach over the rule-based techniques. Machine learning is combined with statistical approaches often in linguistic and sentiment related fields [34]. Successful systems have been developed using basic machine learning classifiers and strong underpinning feature sets [34]. with several word dictations. Systems developed with the strongest features and sentiment lexicons usually utilize linear support vector machines to train their underlying systems [34]. Similarly, an ensemble meta-classifier-based approach, was able to successfully reproduce previous implementations, where it calculates the average confidence score of individual classifiers for positive, negative and neutral classes [34]. Ensemble approach has proven to be powerful in combining several methods

- **DEEP LEARNING APPROACH**
  Deep learning fundamentally differs from machine learning because of the ability it has to learn underlying features in data using neural networks. A standard neural network (NN) consists of many simple, connected units called neurons, each producing a sequence of real-valued activations [23]. Input neurons get activated through sensors perceiving the environment, other neurons get activated through weighted connections from previously active neurons [23]. Some neurons may influence the environment by triggering actions. Each input has a weight associated with it that depicts the importance of it compared to other inputs [23]. Recently, deep learning models have obtained a significant success on various natural language processing tasks, such as semantic analysis, machine translation and text summarization [24]. Neural Network architectures map textual context into logical representations that are then used for answer prediction. These neural networks utilize bidirectional Long Short-Term Memory (LSTM) units for question processing and answer classification [24].
  Recurrent Neural Networks have shown promise in natural language processing. These networks differ from traditional neural networks because of the relationships the hidden layers maintain with the previous values [35]. This recurrent property gives RNN the potential to model long span dependencies [35] [39]. This allows the network to cluster similar histories that allows efficient

representation of patterns with variable length [36]. However recurrent networks are not without its flaws. The major issue with Recurrent Neural Networks is that gradient computation becomes increasingly ill-behaved the farther back in time an error signal must be propagated, and that therefore learning arbitrarily long span phenomena will not be possible [36]. To address this issue several techniques have been introduced. Out of these Backpropagation through time (BPTT) is the most probable [37]. With BPTT, the error is propagated through recurrent connections back in time for a specific number of time steps [37]. Thus, the network learns to remember information for several time steps in the hidden layer when it is learned by the BPTT.

- **NEURAL NETWORK MODELS** RNNs have been able to make a significant improvement in contrast to other machine learning techniques due to their ability to learn and carry out complicated transformations of data its ability maintaining long-term as well as short-term dependencies. They are said to be 'Turing-Complete' [16], therefore having the capacity to simulate arbitrary procedures. RNN contains an interplay of Reasoning Attention, and Memory, commonly referred to as the 'RAM model' in Deep Neural Networks (DNN). In order to obtain state-of-the-art results different models have been introduced overtime in order to improve the performance of DNNs. Researchers have been keen on building models of computation with various forms of explicit storage. Google's DeepMind project released Neural Turing Machines that extend the capabilities of neural networks by coupling them to external memory resources, which they can interact with by attentional processes [16]. NTM is similar to a regular computer with Von Neumann architecture that reads and writes to a memory. The significance is that these operations are differentiable and the controller that handles reads and writes is a neural network. It is shown that NTMs can infer simple algorithms such as copying, sorting, and associative recall from input and output examples. One of the main reasons why NTMs were so popular is that they are derived from psychology, cognitive science, neuroscience, and linguistics. Short-Term memory in NTMs is handled by resembling a working memory system, designed to solve tasks that require the application of approximate rules to data that are quickly bound to memory slots known as "rapidly created variables". Additionally, NTMs use an attentional process to read from and write to memory selectively. Fig. 6 shows the architecture of a Neural Turing Machine.

  End-to-end Memory Networks introduces a recurrent attention model over a possibly large external memory [20]. Since they are trained end-to-end it requires significantly less supervision during training. This technique is applicable to synthetic question answering and to language modelling. An RNN architecture is presented where it reads from a long term memory multiple times before producing the output hence, improving its ability to reason. An RNN has a single chance to look at the inputs that are being fed one by one in order. But End-to-end memory network's architecture places all its inputs in the memory and the model decides which part to read next. After NTMs, Facebook AI research introduced Stack Augmented Recurrent Nets. They have attempted to show that some basic algorithms can be learned from sequential data using a recurrent network associated with a trainable memory [17]. Although machine learning has progressed over the years, with the scaling up of learning algorithms, alternative hardware such as GPUs or large clusters have been necessary. It is not practical with real-world applications. This approach has increased the learning capabilities of recurrent nets by allowing them to learn how to control an infinite structured memory. Neural Training Machines were more expensive than previously considered due to the utilization of an external memory. Thus, Reinforcement Learning Neural Turing Machines (RLNTM) was introduced [18]. RLNTMs use a Reinforcement Learning Algorithm to train Neural Network that interacts with interfaces such as memory tapes, input tapes, and output tapes, to solve simple algorithmic tasks. RLNTMs use Reinforce algorithm to learn where to access the discrete interfaces and to use the backpropagation algorithm to determine what to write to the memory and to the output. RLNTMs have succeeded at problems such as copying an input several times to the output tape, reversing a sequence, and a few more tasks of comparable difficulty

3 Materials and Method

3.1 Data-set

COVID-19 Open Research Dataset (CORD-19). CORD-19 is a resource of over 1,000,000 scholarly articles, including over 400,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses. This freely available dataset is provided to the global research community to apply various AI techniques for support COVID pandemic.

CORD-19 integrates papers and preprints from several sources, where a paper is defined as the base unit of published knowledge, and a preprint as an unpublished but publicly available counterpart of a paper. Each paper is associated with bibliographic metadata, like title, authors, publication venue, etc, as well as unique identifiers such as a DOI, PubMed Central ID, PubMed ID, the WHO Covidence ,4 MAG identifier , and others. Some papers are associated with documents, the physical artifacts containing paper content; these are the familiar PDFs, XMLs, or physical print-outs.

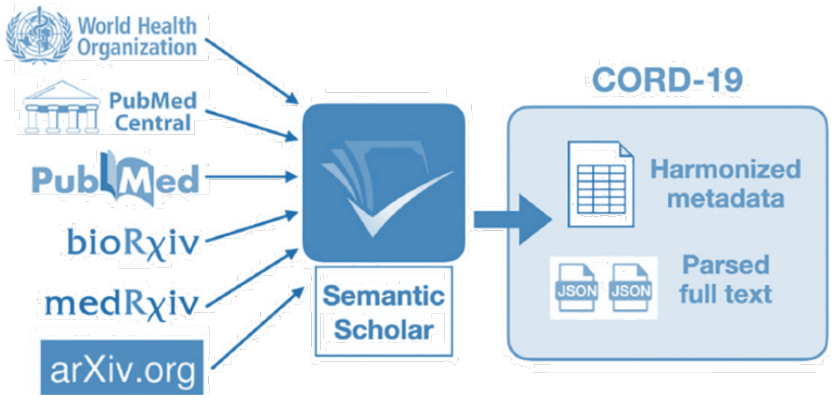


Figure 8: CORD-19 Dataset Sources

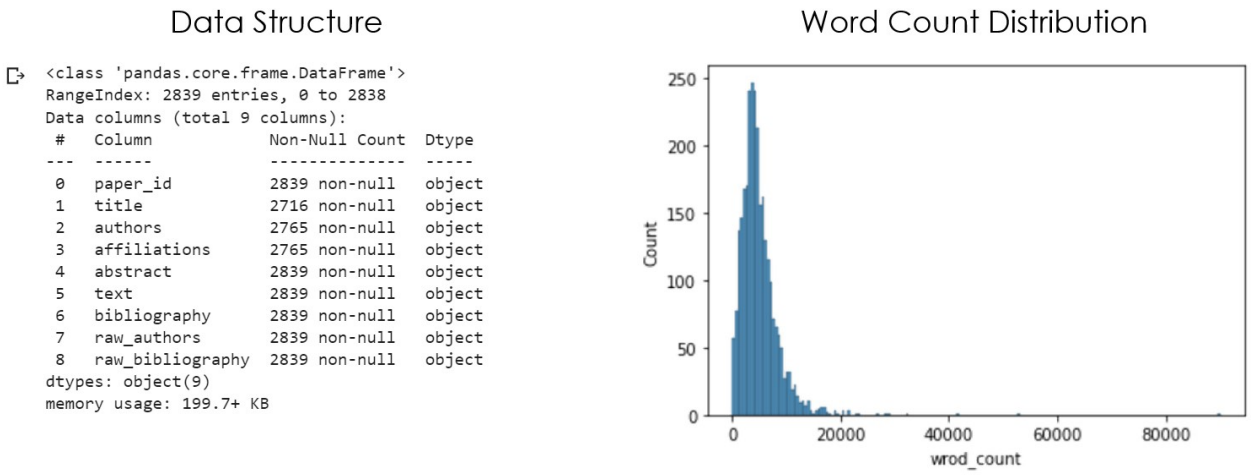


Figure 9: CORD-19 Dataset Metadata

3.2 Design Approach

3.2.1 NLP

1. Json file ingestion from Kaggle CORD 19 dataset
2. Data clean-up and pre-processing

3. Convert to DataFrame

4. onvert DataFrame to documents for Haystack indexing pipeline

3.2.2 Open Haystack Framework

1. Extract top 5 response from retriever-reader pipeline and Identify source document of response

3.2.3 Text Summarization

1. Summarize text using BERT and GPT-2

3.3 Evaluation Metrics

Evaluation is crucial in the research and development of automatic summarization applications, in order to determine the appropriateness of a summary based on different criteria, such as the content it contains, and the way it is presented. To perform an adequate evaluation is of great relevance to ensure that automatic summaries can be useful for the context and/or application they are generated for. The main challenge to be addressed in evaluation is the definition and use of a good metric, able to capture whether the summary is good enough. But the concept of good is very subjective and depends on a great number of issues, so every metrics may be not suitable for all types of summaries.

3.3.1 ROUGE

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is essentially a set of metrics for evaluating automatic summarization of texts as well as machine translations.

It works by comparing an automatically produced summary or translation against a set of reference summaries (typically human-produced). Let's say that we have the following system and reference summaries:

System Summary (what the machine produced):

```
the cat was found under the bed
```

Reference Summary (gold standard — usually by humans):

```
the cat was under the bed
```

If we consider just the individual words, the number of overlapping words between the system summary and reference summary is 6. This, however, does not tell you much as a metric. To get a good quantitative value, we can actually compute the precision and recall using the overlap.

Simply put, recall (in the context of ROUGE) refers to how much of the reference summary the system summary is recovering or capturing. If we are just considering the individual words, it can be computed as

$$\frac{NumberofWordsOverlapping}{TotalWordsinReference} \tag{4}$$

in this example recall would be

$$Recall = \frac{6}{6} = 1.0 \tag{5}$$

This means that all the words in the reference summary have been captured by the system summary.

$$Precision = \frac{6}{7} = 0.86 \tag{6}$$

summaries are in some way forced to be concise through some constraints, then you could consider using just the recall, since precision is of less concern in this scenario.

ROUGE-N, ROUGE-S, and ROUGE-L can be thought of as the granularity of texts being compared between the system summaries and reference summaries.

- ROUGE-N — measures unigram, bigram, trigram and higher order n-gram overlap
- ROUGE-L — measures longest matching sequence of words using LCS. An advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order. Since it automatically includes longest in-sequence common n-grams, you don't need a predefined n-gram length.
- ROUGE-S — Is any pair of words in a sentence in order, allowing for arbitrary gaps. This can also be called skip-gram concurrence. For example, skip-bigram measures the overlap of word pairs that can have a maximum of two gaps in between words. As an example, for the phrase "cat in the hat" the skip-bigrams would be "cat in, cat the, cat hat, in the, in hat, the hat".

ROUGE-1 refers to overlap of unigrams between the system summary and reference summary. ROUGE-2 refers to the overlap of bigrams between the system and reference summaries.

ROUGE-2 precision and recall scores.

System Summary:

```
the cat was found under the bed
```

Reference Summary:

```
the cat was under the bed
```

System Summary Bigrams:

```
the cat, cat was, was found, found under, under the, the bed
```

Reference Summary Bigrams:

```
the cat, cat was, was under, under the, the bed
```

Based on above :

$$Rouge2\text{–}Recall = \frac{4}{5} = 0.8 \tag{7}$$

$$Rouge2\text{–}Precision = \frac{4}{6} = 0.67 \tag{8}$$

ROUGE-L is based on the longest common subsequence (LCS) between our model output and reference, i.e. the longest sequence of words (not necessarily consecutive, but still in order) that is shared between both. A longer shared sequence should indicate more similarity between the two sequences.

ROUGE-S allows us to add a degree of leniency to the n-gram matching performed with ROUGE-N and ROUGE-L. ROUGE-S is a skip-gram concurrence metric: this allows to search for consecutive words from the reference text that appear in the model output but are separated by one-or-more other words.

Consider the new reference R and candidate summary C:

- R: The cat is on the mat
- C: The gray cat and the dog

If we consider the 2-gram "the cat", the ROUGE-2 metric would match it only if it appears in C exactly, but this is not the case since C contains "the gray cat". However, using ROUGE-S with unigram skipping, "the cat" would match "the gray cat" too.

We can compute ROUGE-S precision, recall, and F1-score in the same way as the other ROUGE metrics.

Pros and Cons of ROUGE
This is the tradeoff to take into account when using ROUGE.

- Pros: it correlates positively with human evaluation, it's inexpensive to compute and language-independent
- Cons: ROUGE does not manage different words that have the same meaning, as it measures syntactical matches rather than semantics

3.3.2 BELU

The Bilingual Evaluation Understudy Score, or BLEU for short, is a metric for evaluating a generated sentence to a reference sentence. A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0. BLEU also penalizes sentences shorter than the reference sentence. We can all now agree why BLEU is a widely used metric but it does have some flaws like it does not consider meaning.

The score was developed for evaluating the predictions made by automatic machine translation systems. It is not perfect, but does offer 5 compelling benefits:

- It is quick and inexpensive to calculate.
- It is easy to understand.
- It is language independent.
- It correlates highly with human evaluation.
- It has been widely adopted.

$$logBleu = min(1 - \frac{r}{c}, 0) + 4\sum_{i=1}^{4} \frac{logp_n}{4} \tag{9}$$

$$= min(1 - \frac{r}{c}, 0) + \frac{logp_1 + logp_2 + logp_3 + logp_4}{4} \tag{10}$$

Bleu Score is calculated by considering the text of the entire predicted corpus as a whole. Therefore you cannot compute Bleu Score separately on each sentence in the corpus, and then average those scores in some way.

Strengths of Bleu Score

The reason that Bleu Score is so popular is that it has several strengths:

- It is quick to calculate and easy to understand.
- It corresponds with the way a human would evaluate the same text.
- Importantly, it is language-independent making it straightforward to apply to your NLP models.
- It can be used when you have more than one ground truth sentence.
- It is used very widely, which makes it easier to compare your results with other work.

Weaknesses of Bleu Score

In spite of its popularity, Bleu Score has been criticized for its weaknesses:

- It does not consider the meaning of words. It is perfectly acceptable to a human to use a different word with the same meaning eg. Use “watchman” instead of “guard”. But Bleu Score considers that an incorrect word.
- It looks only for exact word matches. Sometimes a variant of the same word can be used eg. “rain” and “raining”, but Bleu Score counts that as an error.
- It ignores the importance of words. With Bleu Score an incorrect word like “to” or “an” that is less relevant to the sentence is penalized just as heavily as a word that contributes significantly to the meaning of the sentence.
- It does not consider the order of words eg. The sentence “The guard arrived late because of the rain” and “The rain arrived late because of the guard” would get the same (unigram) Bleu Score even though the latter is quite different.

ROUGE vs BLEU

In case you don't know the BLEU metric already, I suggest that you read the companion article Learn the BLEU metric by examples to get a grasp on it.

- BLEU focuses on precision: how much the words (and/or n-grams) in the candidate model outputs appear in the human reference.
- ROUGE focuses on recall: how much the words (and/or n-grams) in the human references appear in the candidate model outputs.
- These results are complementing, as is often the case in the precision-recall tradeoff.

3.3.3 METOR

metric was primarily used in the Machine Translation literature. Checkout our article giving an overview of techniques for machine translation to get a better understanding of the translation application.

The steps to compute the METEOR metric. Computing an Alignment

An alignment between the generated text and the reference can be done by matching word for word, or by using tools for similarity such as word embeddings, dictionary and so on.

A chunk in an alignment is an adjacent set of words that map to adjacent set of words in the reference. The above alignment has three chunks. While there are multiple alignments possible we want to pick that alignment where the number of chunks is the lowest. For instance, take two sentences, "the cat likes the bone" (generated text) and "the cat loves the bone" (Reference). The word "the" in the generated text can be mapped to either of the two "the"s in the reference, but mapping it to the first makes more sense since it leads to just one chunk since all words in generated text map to reference consecutively: The – The, cat – cat, likes – loves, the – the, bone -bone. METEOR takes into account both the precision and recall while evaluating a match.

$$Precision = \frac{m}{w_t} \tag{11}$$

$$Recall = \frac{m}{w_r} \tag{12}$$

$$F_mean = \frac{10PR}{R + 9P} \tag{13}$$

where

m: Number of uni-grams in the candidate translation also found in reference

wt: Number of uni-grams in candidate translation

wr: Number of uni-grams in reference translation

Computing Chunk Penalty A chunk is a set of consecutive words. Typically we note that chunks of words in the source map to chunks of words in the target. The chunk penalty gives a penalty based

on the number of chunks in candidate that map to chunks in target or the reference. In an ideal case, if the candidate and the reference are the same, all words in candidate map to all words in reference consecutively and we just have one chunk. But in reality, we have more chunks in a less than ideal match.

The chunk penalty is computed as follows.

$$p = 0.5(\frac{c}{u_m})^2 \tag{14}$$

C: Number of chunks in candidate Um: Unigrams in candidate

Computing the overall METEOR score

The final meteor score combines the F-score computed from precision and recall with the chunk penalty.

$$M = F_{mean}(1 - p) \tag{15}$$

Hence, while the set of words in the reference and candidate are the same, in this example, we see a chunk penalty. This penalty ensures that "on the cat sat the mat" which matches perfectly with the reference (just 1 chunk) gets a higher score than "the cat sat on the mat".

4 Systems Requirement Specification

4.1 System Visualization

For the QA and summarization presentation, I have used streamlit for visualization. Streamlit is an open-source python framework for building web apps for Machine Learning and Data Science. We can instantly develop web apps and deploy them easily using Streamlit. Streamlit allows you to write an app the same way you write a python code. Streamlit makes it seamless to work on the interactive loop of coding and viewing results in the web app. Streamlit allows you to write an app the same way you write a python code. The streamlit has a distinctive data flow, any time something changes in your code or anything needs to be updated on the screen, streamlit reruns your python script entirely from the top to the bottom. This happens when the user interacts with the widgets like a select box or drop-down box or when the source code is changed.

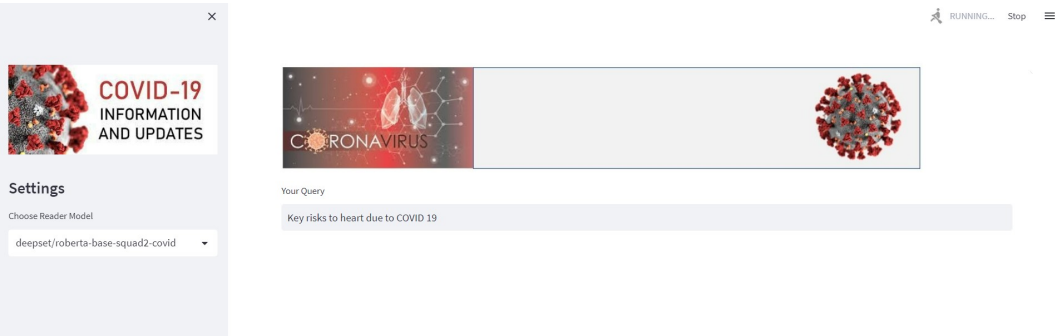


Figure 10: Streamlit QA Header

Responses..

Patients with pre-existing cardiac conditions and those with elevated serum troponin during their disease course have worse outcomes, including higher mortality, longer length of hospital stay and need for mechanical ventilation	myocarditis, acute heart failure and cardiac arrest	pre-existing cardiovascular disease increases the risk of COVID-19 infection and is indeed present in a high number of cases	amaged endothelial and epithelial cells, which increases the expression of chemokines. Geneticall	lymphopenia, thrombocytopenia, elevated serum troponin, severely elevated inflammatory markers, high d-dimer, and abnormal renal and liver function tests
73.0%	72.0%	67.0%	61.0%	59.0%

Figure 11: Responses Probability Match

Score %

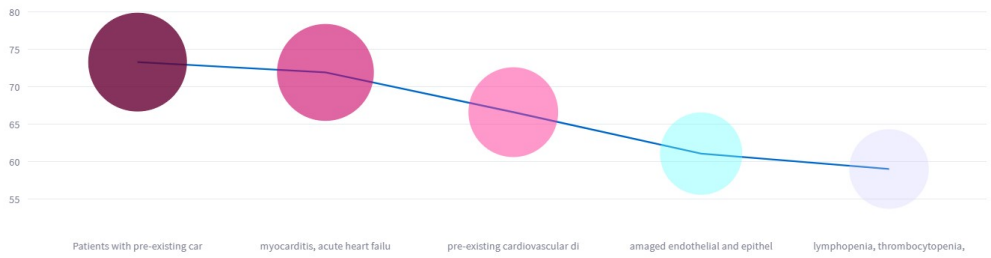
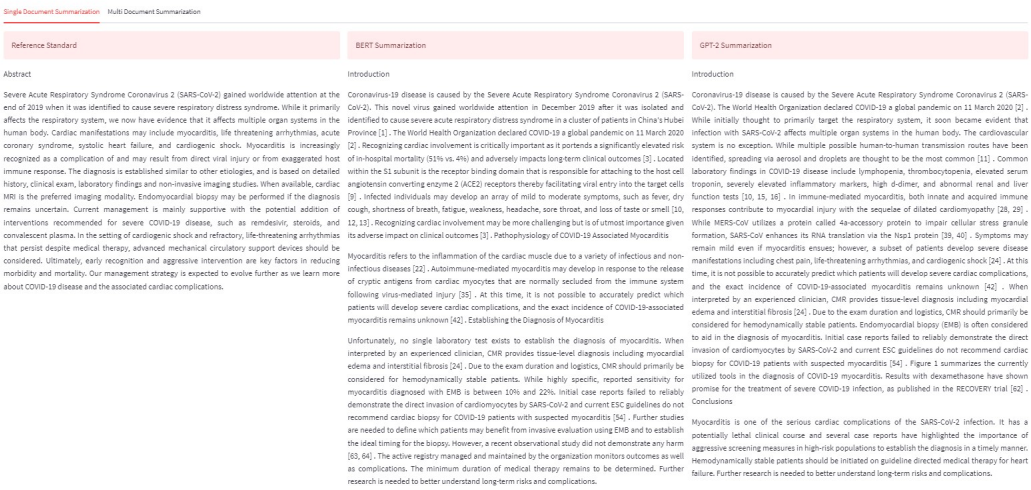


Figure 12: Probability Match


Single Document Summarization | Multi Document Summarization

Reference Standard

Abstract

Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) gained worldwide attention at the end of 2019 when it was identified to cause severe respiratory distress syndrome. While it primarily affects the respiratory system, we now have evidence that it affects multiple organ systems in the human body. Cardiac manifestations may include myocarditis, life threatening arrhythmias, acute coronary syndrome, systolic heart failure, and cardiogenic shock. Myocarditis is increasingly recognized as a complication of and may result from direct viral injury or from exaggerated host immune response. The diagnosis is established similar to other etiologies, and is based on detailed history, clinical exam, laboratory findings and non-invasive imaging studies. When available, cardiac MRI is the preferred imaging modality. Endomyocardial biopsy may be performed if the diagnosis remains uncertain. Current management is mainly supportive with the potential addition of interventions recommended for severe COVID-19 disease, such as remdesivir, steroids, and convalescent plasma. In the setting of cardiogenic shock and refractory, life-threatening arrhythmias that persist despite medical therapy, advanced mechanical circulatory support devices should be considered. Ultimately, early recognition and aggressive intervention are key factors in reducing morbidity and mortality. Our management strategy is expected to evolve further as we learn more about COVID-19 disease and the associated cardiac complications.

BERT Summarization

Introduction

Coronavirus-19 disease is caused by the Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2). This novel virus gained worldwide attention in December 2019 after it was isolated and identified to cause severe acute respiratory distress syndrome in a cluster of patients in China's Hubei Province [1] . The World Health Organization declared COVID-19 a global pandemic on 11 March 2020 [2] . Recognizing cardiac involvement is critically important as it portends a significantly elevated risk of in-hospital mortality (51% vs. 4%) and adversely impacts long-term clinical outcomes [3] . Located within the S1 subunit is the receptor binding domain that is responsible for attaching to the host cell angiotensin converting enzyme 2 (ACE2) receptors thereby facilitating viral entry into the target cells [9] . Infected individuals may develop an array of mild to moderate symptoms, such as fever, dry cough, shortness of breath, fatigue, weakness, headache, sore throat, and loss of taste or smell [10, 12, 13] . Recognizing cardiac involvement may be more challenging but is of utmost importance given its adverse impact on clinical outcomes [3] . Pathophysiology of COVID-19 Associated Myocarditis

Myocarditis refers to the inflammation of the cardiac muscle due to a variety of infectious and non-infectious diseases [22] . Autoimmune-mediated myocarditis may develop in response to the release of cryptic antigens from cardiac myocytes that are normally secluded from the immune system following virus-mediated injury [35] . At this time, it is not possible to accurately predict which patients will develop severe cardiac complications, and the exact incidence of COVID-19-associated myocarditis remains unknown [42] . Establishing the Diagnosis of Myocarditis

Unfortunately, no single laboratory test exists to establish the diagnosis of myocarditis. When interpreted by an experienced clinician, CMR provides tissue-level diagnosis including myocardial edema and interstitial fibrosis [24] . Due to the exam duration and logistics, CMR should primarily be considered for hemodynamically stable patients. While highly specific, reported sensitivity for myocarditis diagnosed with EMB is between 10% and 22%. Initial case reports failed to reliably demonstrate the direct invasion of cardiomyocytes by SARS-CoV-2 and current ESC guidelines do not recommend cardiac biopsy for COVID-19 patients with suspected myocarditis [34] . Further studies are needed to define which patients may benefit from invasive evaluation using EMB and to establish the ideal timing for the biopsy. However, a recent observational study did not demonstrate any harm [63, 64] . The active registry managed and maintained by the organization monitors outcomes as well as complications. The minimum duration of medical therapy remains to be determined. Further research is needed to better understand long-term risks and complications.

GPT-2 Summarization

Introduction

Coronavirus-19 disease is caused by the Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2). The World Health Organization declared COVID-19 a global pandemic on 11 March 2020 [2] . While initially thought to primarily target the respiratory system, it soon became evident that infection with SARS-CoV-2 affects multiple organ systems in the human body. The cardiovascular system is no exception. While multiple possible human-to-human transmission routes have been identified, spreading via aerosol and droplets are thought to be the most common [11] . Common laboratory findings in COVID-19 disease include lymphopenia, thrombocytopenia, elevated serum troponin, severely elevated inflammatory markers, high d-dimer, and abnormal renal and liver function tests [10, 15, 16] . In immune-mediated myocarditis, both innate and acquired immune responses contribute to myocardial injury with the sequelae of dilated cardiomyopathy [28, 29] . While MERS-CoV utilizes a protein called 4a-accessory protein to impair cellular stress granule formation, SARS-CoV enhances its RNA translation via the Nsp1 protein [39, 40] . Symptoms may remain mild even if myocarditis ensues; however, a subset of patients develop severe disease manifestations including chest pain, life-threatening arrhythmias, and cardiogenic shock [24] . At this time, it is not possible to accurately predict which patients will develop severe cardiac complications, and the exact incidence of COVID-19-associated myocarditis remains unknown [42] . When interpreted by an experienced clinician, CMR provides tissue-level diagnosis including myocardial edema and interstitial fibrosis [24] . Due to the exam duration and logistics, CMR should primarily be considered for hemodynamically stable patients. Endomyocardial biopsy (EMB) is often considered to aid in the diagnosis of myocarditis. Initial case reports failed to reliably demonstrate the direct invasion of cardiomyocytes by SARS-CoV-2 and current ESC guidelines do not recommend cardiac biopsy for COVID-19 patients with suspected myocarditis [34] . Figure 1 summarizes the currently utilized tools in the diagnosis of COVID-19 myocarditis. Results with dexamethasone have shown promise for the treatment of severe COVID-19 infection, as published in the RECOVERY trial [62] . Conclusions

Myocarditis is one of the serious cardiac complications of the SARS-CoV-2 infection. It has a potentially lethal clinical course and several case reports have highlighted the importance of aggressive screening measures in high-risk populations to establish the diagnosis in a timely manner. Hemodynamically stable patients should be initiated on guideline directed medical therapy for heart failure. Further research is needed to better understand long-term risks and complications.

Figure 13: Text Summarization Result

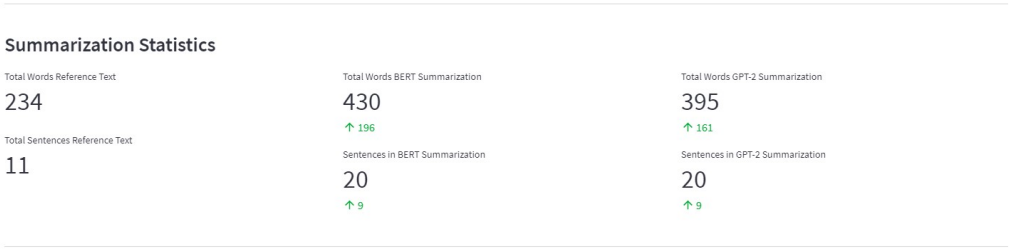


Figure 14: Text Summarization Statistics

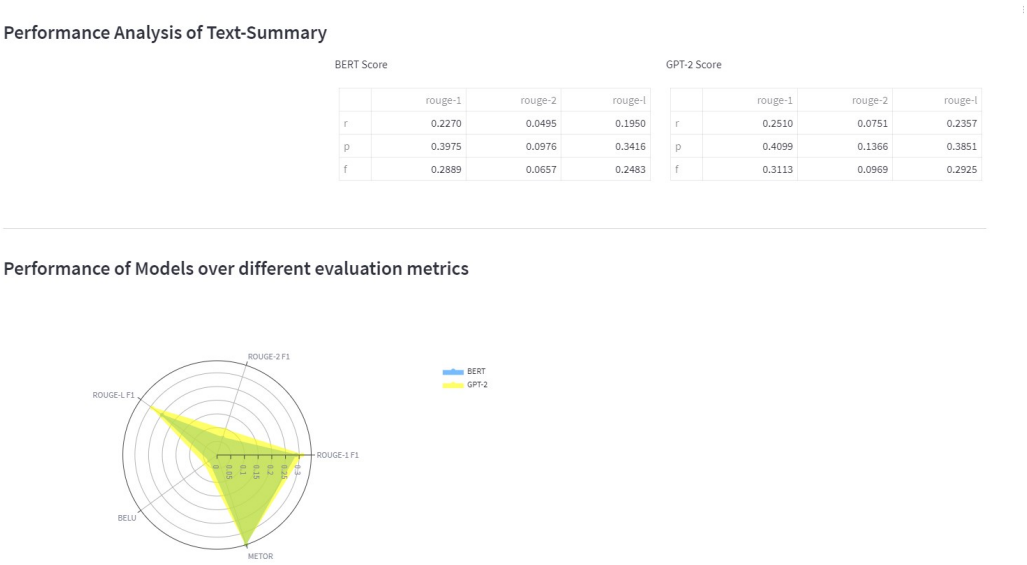


Figure 15: Summarization Performance

Visual Studio - configuration Version: 1.76.2 (user setup)Commit: ee2b180d582a7f601fa6ecfdad8d9fd269ab1884
Date: 2023-03-14T17:55:54.936Z
Electron: 19.1.11
Chromium: 102.0.5005.196
Node.js: 16.14.2
V8: 10.2.154.26-electron.0
OS: Windows$_NTx$6410.0.22621
$Sandboxed : No$

4.2 Functional and Non-Functional requirements

Functional requirements define a function that a system or system element must be qualified to perform and must be documented in different forms. The functional requirements describe the behavior of the system as it correlates to the system's functionality. Functional requirements are written in a simple language, so that it is easily understandable. The examples of functional requirements are authentication, business rules, audit tracking, certification requirements, transaction corrections, etc. These requirements allow us to verify whether the application provides all functionalities mentioned in the application's functional requirements. They support tasks, activities, user goals for easier project management.

Non-functional requirements are not related to the software's functional aspect. They can be the necessities that specify the criteria that can be used to decide the operation instead of specific behav-

iors of the system. Basic non-functional requirements are - usability, reliability, security, storage, cost, flexibility, configuration, performance, legal or regulatory requirements, etc.

- Execution qualities like security and usability, which are observable at run time.
- Evolution qualities like testability, maintainability, extensibility, and scalability that embodied in the static structure of the software system.

Non-functional requirements specify the software's quality attribute. These requirements define the general characteristics, behavior of the system, and features that affect the experience of the user. They ensure a better user experience, minimizes the cost factor. Non-functional requirements ensure that the software system must follow the legal and adherence rules. The impact of the non-functional requirements is not on the functionality of the system, but they impact how it will perform. For a well-performing product, atleast some of the non-functional requirements should be met.

4.2.1 Functional requirements

- Ability to accept queries from users
- Ability to identify top document from repositories
- Ability to summarize top document and provide statistics to user to choose which Summarization technique to be used

4.2.2 Nonfunctional requirements

- Reliability - Technology that is highly reliable functions with the same or similar efficiency after extensive use
- Speed - Since we use open source free cloud version , speed may not be the best available in market.
- Security - The access to site is controlled and managed by Streamlit cloud and Github. The data is CORD 19 , freely available , no sensitive data.
- Portability - Portability means how effectively a system performs in one environment compared to another. For example, a user might purchase a new cell phone model and download a mobile application they had on their last device. If the application runs as efficiently on the new phone as it did on the old phone, then it's highly portable. As a developer, you can design your applications to function properly on multiple devices to improve portability.
- Compatibility - Highly compatible systems typically function well when other applications are running on a device. Compatibility also allows people who have different operating systems to use the same applications. Since the application is launced on clud and accessable through browser , there are no dependencies on local environments.
- Capacity - The capacity of a system refers to the amount of storage it offers. When using some applications. In our case we are using open source and free cloud versions, capacity is limited. Overall scores are low due to limited amount storage and processing abilities.
- Usability - Simple navigation features

4.3 Constraints and Assumptions

- Proper and complete question text performs better than short texts. Example - **"What is the most advisable quarantine period for COVID 19"**, instead of just **"quarantine period"**
- Both the readers are tuned for English language only. Reading non english reasearch articles is not effective.
- On prem version coding and that of cloud differs, has to be taken care.

- Streamlit reads from top to bottom every time an stremlit element is chosen, this causes page to render every time.
- Due to open source and free cloud version chosen the space and processing capabilities are limited and can cause system to crash.

5 Proposed Methodology

6 Conclusion and Future work

6.1 Results

6.1.1 Reader -Performance Analysis

Figure 16: Question

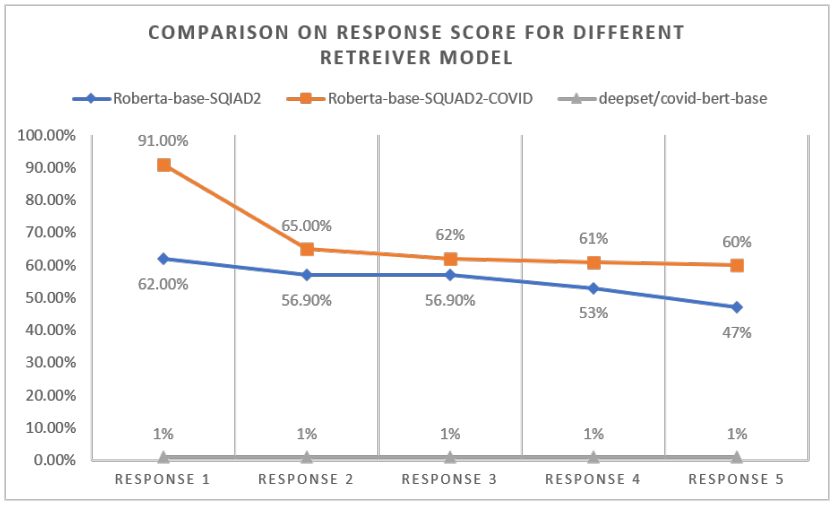
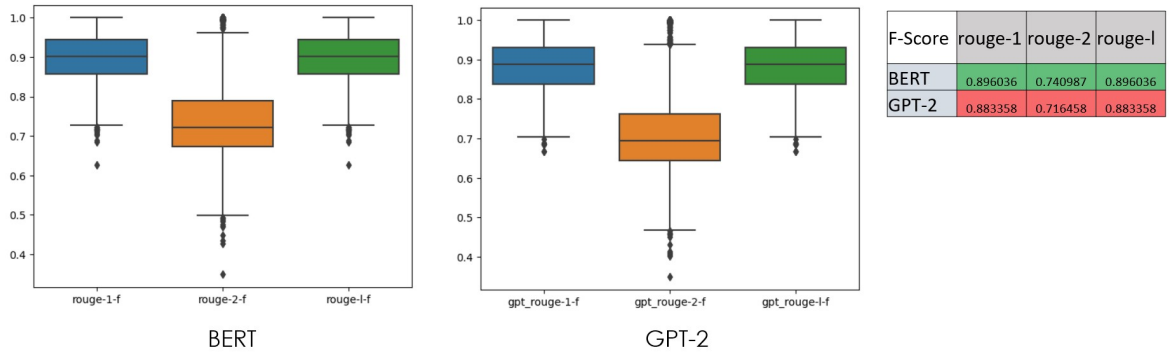


Figure 17: Retriever Model Outcome F1 Score

	Roberta-base-SQIAD2	Roberta-base-SQUAD2-COVID	deepset/covid-bert-base
Response1	Feb 24 until	Quarantine longer than 30 days	Climatically temperate regions ...
Response2	29.45 Days	quarantine length longer than 30 days	raisons and soft tissue njuries
Response3	14-day	14-day	88
Response4	30 days	Restrictions of asymptomatic healthy people...	respiratory capacity , primarily in patients with chronic
Response5	nation wide quarantine	Average qurantine days was 29.45 days with standard deviation of 7.34 days	Mandatory variable ...

Figure 18: Responses

6.1.2 Text Summarization Performance



F-Score	rouge-1	rouge-2	rouge-l
BERT	0.896036	0.740987	0.896036
GPT-2	0.883358	0.716458	0.883358

Figure 19: Text Summarization Outcomes F1 Score

In overall Roberta-base-SQUAD2-COVID retriever performed better in all search cases, which were closer to human language outcomes. After trying multiple questions and input the response and probable scores for the Roberta-base-SQUAD2-COVID were always high.

In terms of text summarization , BERT and GPT-2 has almost similar performance. However BERT had better performance , in most of cases METOR score for GPT-2 always seemed to be better. METOR score reflect more human like responses.

BERT and GPT-2 perform quite differently on the token prediction task depending on the position of the token being predicted. For a fixed sequence length of 100 tokens, BERT performs best when the masked token is between positions 5 and 95, while GPT-2 tends to continually improve as context length increases. Interestingly, when the final token in the sequence is to be predicted, BERT's performance falls off dramatically, while GPT-2 performance remains stable.

6.2 Future Work

In current work I have attempted to summarize one top article, and compared the same with standard reference. For future work , one can attempt multi document summarization based on top 5 documents identified by retriever. The challenge would be to see if they have similar scope and merge them to single text.

References

[1] A. B. Abacha, S. A. Hasan, V. V. Datla, J. Liu, D. Demner-Fushman, and H. Müller. Vqa-med: Overview of the medical visual question answering task at imageclef 2019. *CLEF (working notes)*, 2(6), 2019.

[2] T. Acter, N. Uddin, J. Das, A. Akhter, T. R. Choudhury, and S. Kim. Evolution of severe acute respiratory syndrome coronavirus 2 (sars-cov-2) as coronavirus disease 2019 (covid-19) pandemic: A global health emergency. *Science of the Total Environment*, 730:138996, 2020.

[3] N. A. Akbar, I. Darmayanti, S. M. Fati, and A. Muneer. Deep learning of a pre-trained language model's joke classifier using gpt-2. *Journal of Hunan University Natural Sciences*, 48(8), 2021.

[4] A. Ben Abacha and D. Demner-Fushman. A question-entailment approach to question answering. *BMC bioinformatics*, 20(1):1–23, 2019.

[5] P. B. Brandtzaeg and A. Følstad. Why people use chatbots. In *Internet Science: 4th International Conference, INSCI 2017, Thessaloniki, Greece, November 22-24, 2017, Proceedings 4*, pages 377–392. Springer, 2017.

[6] K. M. Colby, S. Weber, and F. D. Hilf. Artificial paranoia. *Artificial intelligence*, 2(1):1–25, 1971.

[7] D. Diefenbach, A. Both, K. Singh, and P. Maret. Towards a question answering system over the semantic web. *Semantic Web*, 11(3):421–439, 2020.

[8] A. Esteva, A. Kale, R. Paulus, K. Hashimoto, W. Yin, D. Radev, and R. Socher. Co-search: Covid-19 information retrieval with semantic search, question answering, and abstractive summarization. *arXiv preprint arXiv:2006.09595*, 2020.

[9] S. Hofstätter, M. Zlabinger, M. Sertkan, M. Schröder, and A. Hanbury. Fine-grained relevance annotations for multi-task document ranking and question answering. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 3031–3038, 2020.

[10] Q. Jin, B. Dhingra, Z. Liu, W. W. Cohen, and X. Lu. Pubmedqa: A dataset for biomedical research question answering. *arXiv preprint arXiv:1909.06146*, 2019.

[11] R. Khan, A. Das, R. Khan, and A. Das. Introduction to chatbots. *Build better chatbots: A complete guide to getting started with chatbots*, pages 1–11, 2018.

[12] L. C. Klopfenstein, S. Delpriori, S. Malatini, and A. Bogliolo. The rise of bots: A survey of conversational interfaces, patterns, and paradigms. In *Proceedings of the 2017 conference on designing interactive systems*, pages 555–565, 2017.

[13] L. J. Kricka, S. Polevikov, J. Y. Park, P. Fortina, S. Bernardini, D. Satchkov, V. Kolesov, and M. Grishkov. Artificial intelligence-powered search tools and resources in the fight against covid-19. *Ejifcc*, 31(2):106, 2020.

[14] J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. R. Karger. What makes a good answer? the role of context in question answering. In *INTERACT*, 2003.

[15] M. d. G. B. Marietto, R. V. de Aguiar, G. d. O. Barbosa, W. T. Botelho, E. Pimentel, R. d. S. França, and V. L. da Silva. Artificial intelligence markup language: a brief tutorial. *arXiv preprint arXiv:1307.3091*, 2013.

[16] G. Molnár and Z. Szüts. The role of chatbots in formal education. In *2018 IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 000197–000202. IEEE, 2018.

[17] R. Nogueira and K. Cho. Passage re-ranking with bert. corr abs/1901.04085 (2019). *arXiv preprint arXiv:1901.04085*, 2019.

[18] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[19] I. Shen, L. Zhang, J. Lian, C.-H. Wu, M. G. Fierro, A. Argyriou, and T. Wu. In search for a cure: recommendation with knowledge graph on cord-19. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3519–3520, 2020.

[20] D. Su, Y. Xu, T. Yu, F. B. Siddique, E. J. Barezi, and P. Fung. Caire-covid: A question answering and query-focused multi-document summarization system for covid-19 scholarly information management. *arXiv preprint arXiv:2005.03975*, 2020.

[21] R. Tang, R. Nogueira, E. Zhang, N. Gupta, P. Cam, K. Cho, and J. Lin. Rapidly bootstrapping a question answering dataset for covid-19. *arXiv preprint arXiv:2004.11339*, 2020.

[22] A. M. Turing. *Computing machinery and intelligence*. Springer, 2009.

[23] R. Vaishya, M. Javaid, I. H. Khan, and A. Haleem. Artificial intelligence (ai) applications for covid-19 pandemic. *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, 14(4):337–339, 2020.

[24] J. Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[25] C. Wise, V. N. Ioannidis, M. R. Calvo, X. Song, G. Price, N. Kulkarni, R. Brand, P. Bhatia, and G. Karypis. Covid-19 knowledge graph: accelerating information retrieval and discovery for scientific literature. *arXiv preprint arXiv:2007.12731*, 2020.

[26] W. Yu, L. Wu, Y. Deng, R. Mahindru, Q. Zeng, S. Guven, and M. Jiang. A technical question answering system with transfer learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 92–99, 2020.