Get unlimited access to the best of Medium for less than $1/week. Become a member

How to build a Q&A bot for company website

How we created a Q&A bot that uses the site's data to answer user questions

Quantum · Follow

Published in Geek Culture · 11 min read · Oct 18, 2022

16

Automation of the process of interaction between a client and an information resource, as well as retrieving the necessary information from text corpora, is one of the most promising and interesting areas in information technologies, as it helps to significantly reduce the cost and speed up the interaction with various information sources — document archives, books, websites, etc. The IT industry has reached the latest stage in the development of automated searching with the help of artificial intelligence and deep learning. Now we can build complex non-deterministic solutions with great generalization capabilities.

One of the popular scenarios for searching for information in text corpora is the Question Answering Systems (QAS). These systems allow users to quickly answer a question of interest to them on a general topic or in a specialized domain. Here is what researchers in the field of automatic information retrieval have to say about QAS: "For human-computer interaction, natural language is the best information access mechanism for humans. Hence,

Question Answering Systems (QAS) have special significance and advantages over search engines and are considered the ultimate goal of the semantic search for user's information needs".

Quantum has researched this field and created a Q&A bot that can use data from the site to answer user questions. In the study, we used state-of-the-art solutions in the field of NLP, semantic search tested the solution's stability for different use cases and built a pipeline with a web interface through which users can interact with the implemented service.

In this article, you will see a description of the approach to solving the problem, various system' components assessments, ideas for combining these components into a single pipeline, and an analysis of the possibilities and limitations of this type of system.

Architecture

Open in app ↗

Search Write P

bridge network. You can see the application's scheme in Figure 1.

The scope of this solution:

1. Creating a corpus of texts from the site content.
2. Selection of appropriate contexts for the received question.
3. Predicting the answers to a question in each context.
4. Selection and returning the best answer.

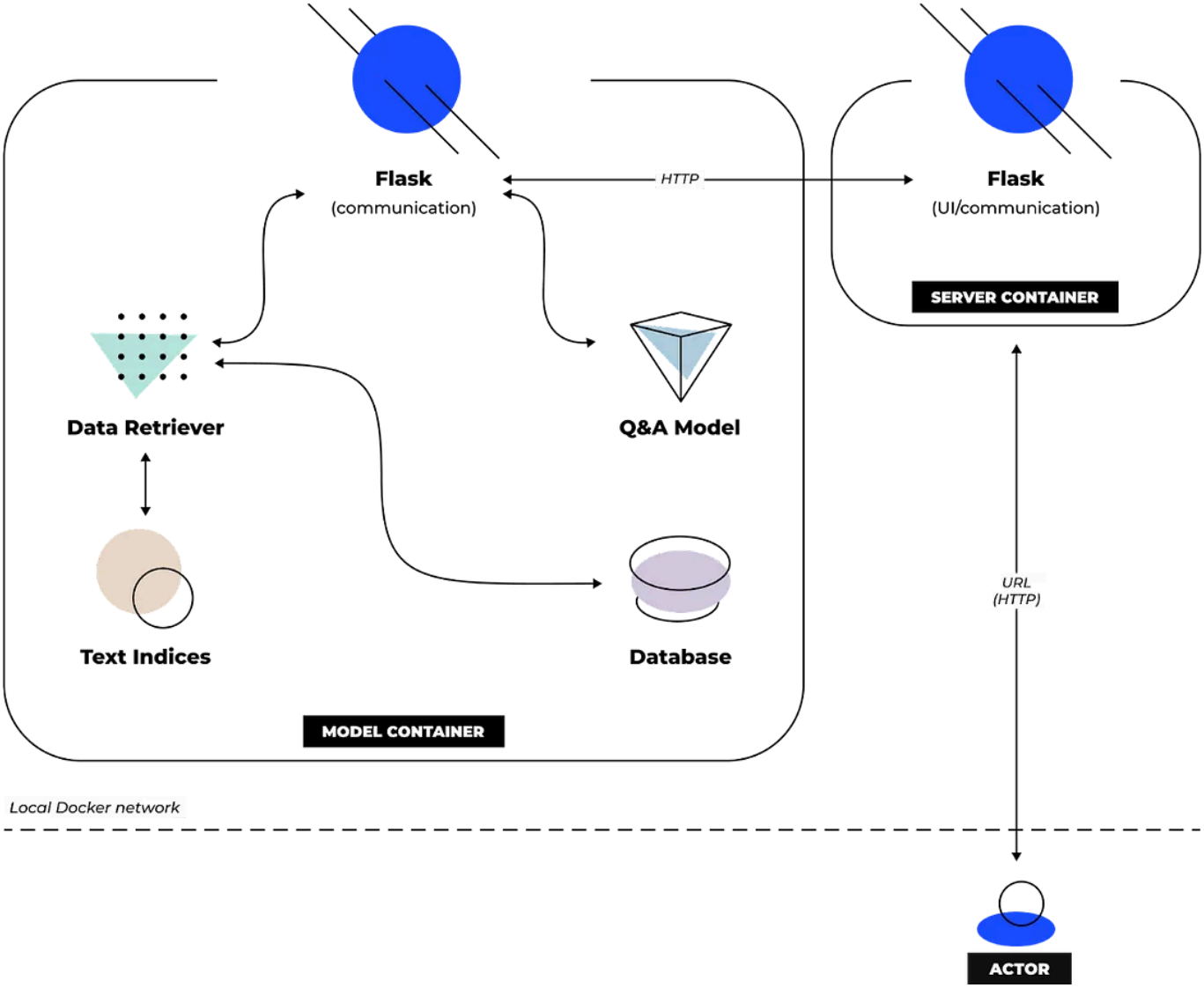


Figure 1. Application scheme

The user communicates with the server container using the GUI. GUI is written in Flask/Dash. When the user enters a question and clicks the "Submit" button, a request to the model is generated on the server container.

The model container receives a request using Flask and forms a query into a retriever — a model for searching for documents that are similar in meaning to the input text. Retriever reads indexes of texts and texts from local storage and returns the top K most suitable contexts, as well as retriever scores for each context, where K is a hyperparameter of the retriever.

With the contexts from the retriever, the service creates K question-context pairs and sends each pair to the Q&A model inference. The model returns answers and confidence scores for each answer.

Next, the model container selects one answer from the proposed. It sends a response with this answer to the server container via HTTP, and the service in the server container displays information on the web interface.

Datasets

To evaluate the performance of our solution, we used several open-source datasets and collected a small sample of questions and contexts from the Quantum website. Here is a description of open-source datasets:

SQuAD2.0

Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset comprising questions posed by crowdworkers on a set of Wikipedia articles. The answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable. SQuAD2.0 combines the 100,000 questions in SQuAD1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones.

NewsQA

NewsQA is a challenging machine comprehension dataset of over 100,000 human-generated question-answer pairs. Crowdworkers supply questions and answers based on a set of over 10,000 news articles from CNN, with answers consisting of spans of text from the corresponding articles.

Quantum site dataset

To form our own questions and context datasets, we took some pages from the Quantum website and extracted meaningful text from them.

We decided to use contexts from pages with more specific meanings — generalized information on the site's first pages (main page, team, expertise, etc.) is rather weak for our purposes.

Therefore, the following pages were chosen:

- Some projects from the Case studies page.
- Pieces of some project descriptions from the R&D page.
- Some articles from the News page.
- Vacancies descriptions from the Careers page.

As a result, 30 contexts were collected. There are 1–4 questions for each context, 100 questions in total. The main requirement for the questions was the brevity and unambiguity of the answer so we could accurately formulate the ground truth for the bot evaluation.

Tools

In this chapter, the modeling tools that we used are described. The solution for getting an answer to a question on a corpus of texts consists of two parts:

- Contexts-based Q&A model — the NLP model which takes a question-context pair in a specific format and produces a meaningful answer to that question based on provided context.
- Retriever — the model that retrieves the most relevant contexts from the entire corpus of texts.

Context-based Q&A models

Q&A model — a natural language model, a neural network trained to predict a word or sequence based on an input sequence of words. Such models today solve some of the most important tasks of the field in NLP — such as machine translation and Q&A systems building. There are several types of models for the Q&A problem, but we settled on context-based models. These models receive both the question itself and the context as input — a piece of text from which you can get an answer. These models are much easier to

infer and can be used either on a general topic or in a specialized domain — it's just a matter of the context you give the model.

At the stage of the initial search for solutions, we selected two Q&A models. Both of them had pre-trained weights and good scores on common benchmarks. A brief description of each model is given below. Initially, we also considered other open-source models, but these two models had pre-trained weights, and their codebases were the most convenient to implement in our pipeline.

UnifiedQA

A Q&A model based on T5 and trained on many open-source datasets. It has several options that vary in complexity and size — from 242 MB to 42 GB. UnifiedQA is based on the T5 generative model with a recursive classification head. One of the advantages of this model is that it was trained on questions of different formats: extractive — when the answer is unequivocally and exactly found in the context), abstractive — when the answer should be self-formulated from information in the context and not taken from a piece of context, Multiple-choice — when an answer can be selected from a list of answers and Yes/No — when the user expects a 'yes' or 'no' answer as the response.

Fusion-in-Decoder (FiD)

A state-of-the-art solution with a pre-trained model used in a pipeline similar to ours. The model also has several options — base and large, each of them is trained on a certain dataset that is popular in this task. The main feature of the model is that it was created to search for information in various contexts. The model receives several question-context pairs, generates an encoded vector in the latent space for each, concatenates these vectors into one, and answers the question.

To begin with, we decided to determine which of the Q&A models answers better to questions from open-source datasets, as well as our questions. To do this, we discarded the problem of context retrieving and provided the models with the correct question-context pairs. We determined that on open-source datasets, the lightest version of the UnifiedQA has the same metric values as FiD. At the same time, FiD performs much worse on our dataset — its answers less often match the ground truth, although, in terms of the meaning, the difference is not very large with UnifiedQA. In this way, to build our pipeline, we used the UnifiedQA model.

Metrics	Models	
	UnifiedQA	FiD
F1	54.8%	24.1%
Rouge	67.9%	38.1%
Cosine	55.3%	52.9%

You can find metrics on our dataset in Table 1.

Retriever

Retrieving is the part of the pipeline responsible for finding appropriate contexts for the question, from which the QA model will then generate answers. Initially, we found several solutions for this part of the pipeline. There are ready-to-use solutions and approaches to text encoding that can be used in implementing your own retriever from scratch.

We decided to use the Pyserini — Python toolkit for reproducible information retrieval research with sparse and dense representations. Dense retrieving (using Faiss searcher) in this toolkit is based on nearest-neighbor searching over transformer-encoded representations of texts.

We chose Pyserini because it is more suitable for our task than the other solutions. Its advantages:

- Semantic search, as opposed to match search.
- Selection speed is higher compared to using solutions based on large DL models.
- Availability of a trained model.
- The ability to concatenate newly built indexes with old ones when expanding the corpus of texts.

Building a Pipeline

In the first revision of the bot, we faced the problem that if there is no correct answer in the context, the model gives a meaningless answer, and the model confidence of this answer can be pretty high. Periodically, this score turned out to be higher than the score of an answer taken from the correct context.

This significantly degraded accuracy when the multi-context prediction was used. Even though the retriever correctly assigned one of the highest ranks to the proper context, the model in this context also answered correctly. We identified this issue by looking at the quality of the Q&A model and retriever separately. We found that answers obtained from the context-based pipeline (without retriever) are much more meaningful than those obtained from the context-free pipeline. In figure 2, you can see which rank the retriever gave to the correct context for each question from the dataset. For our dataset, we see that 97% of the contexts are in the top 5 results, and 91% are in the top 2.

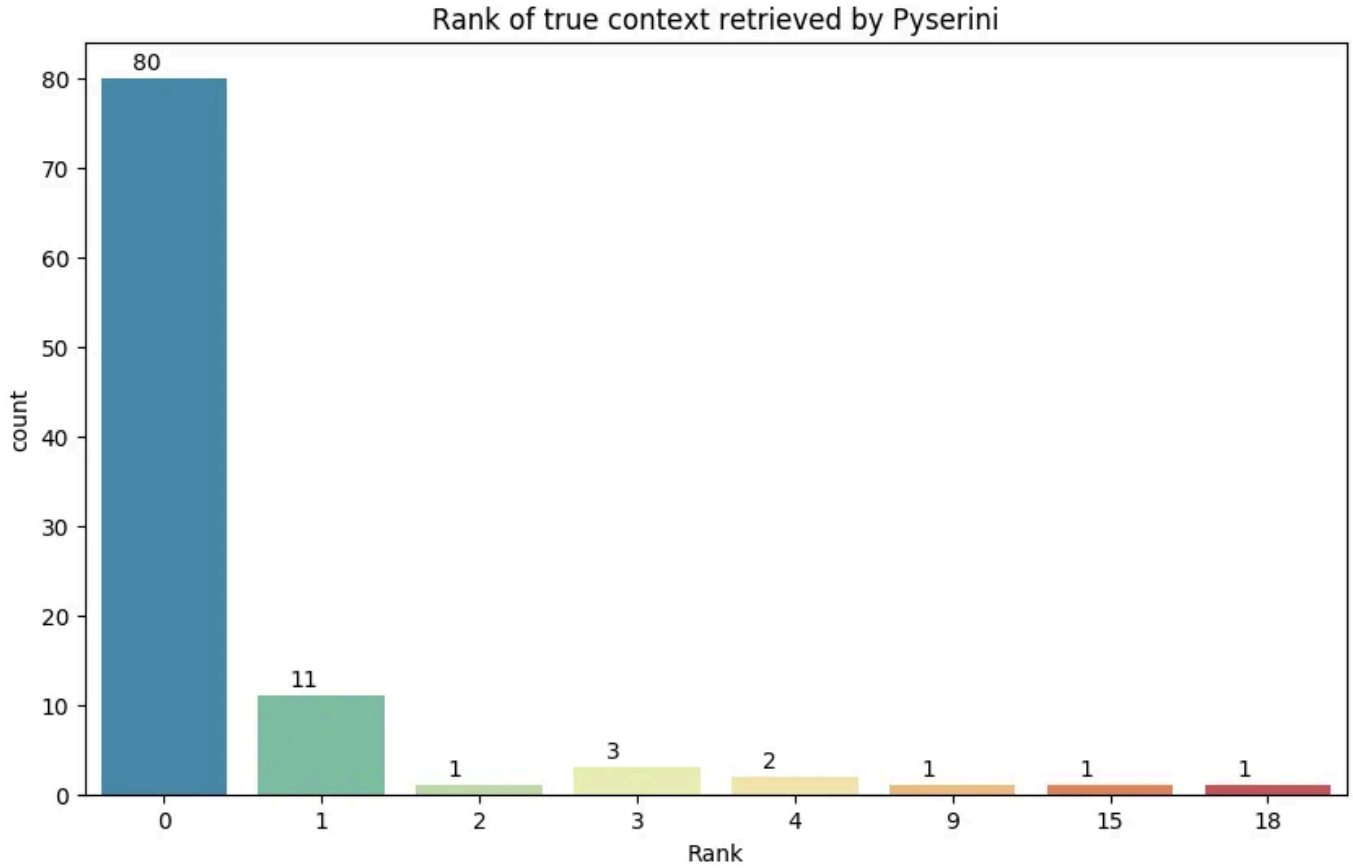


Fig. 2. What rank is given by the retriever to the correct context

In Figure 3, you can see how much the model's quality drops with the number of contexts increase. Although it would be logical to expect that the quality should increase until it hits the limit.

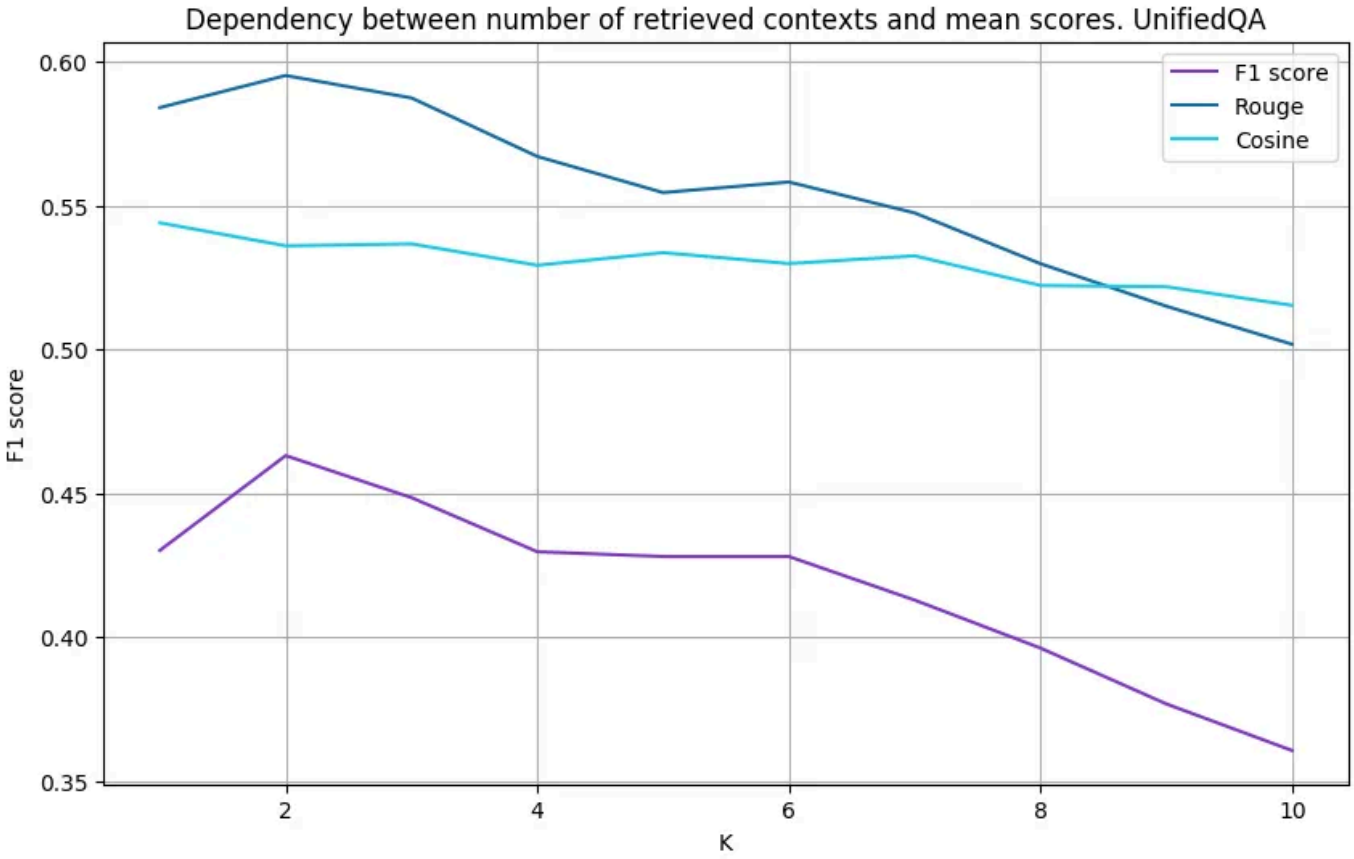


Fig. 3. Mean metrics for each count of contexts.

To solve this problem, we considered the following approaches:

- Combinations of contexts — using random sampling, we created overlapping groups from a small number of selected contexts, concatenated all the contexts from the group into one large text, and used this text as a new context. We did this several times, getting a new list of large contexts.
- Another way to calculate answer confidence scores. It is the score by which we select one answer from the list of answers corresponding to the list of question-context pairs. We have considered various options:

— Tokens' confidence averaging:

$$C_{answer} = 1/k \sum_{i=0}^{k} C_{i-token};$$

— Weighting answers confidence with retriever scores for contexts:

$$C_{answer} = S_{context} \times \prod_{i=0}^{k} C_{i-token};$$

— Combination of approaches 1 and 2:

$$C_{answer} = S_{context} \times 1/k \sum_{i=0}^{k} C_{i-token};$$

— Weighting answers confidences with softmax of retriever scores for contexts:

$$[C_{answer}] = [Softmax(S_{context})] \times [\prod_{i=0}^{k} C_{i-token}].$$

We stopped on the last variant in the second approach, as it gave a noticeable increase in metrics. The intuition of this method is simple — we choose the option, the joint probability of receiving it by the retriever, and the AMC model is the highest among all answers. Now we can look at the metrics' dependence on the increase in the number of contexts in figure 4. With an increase in the number of contexts selected by the retriever, the quality rises to a certain threshold and does not change. Obviously, the difference between graphs 3 and 4 will be even greater for large text corpora.

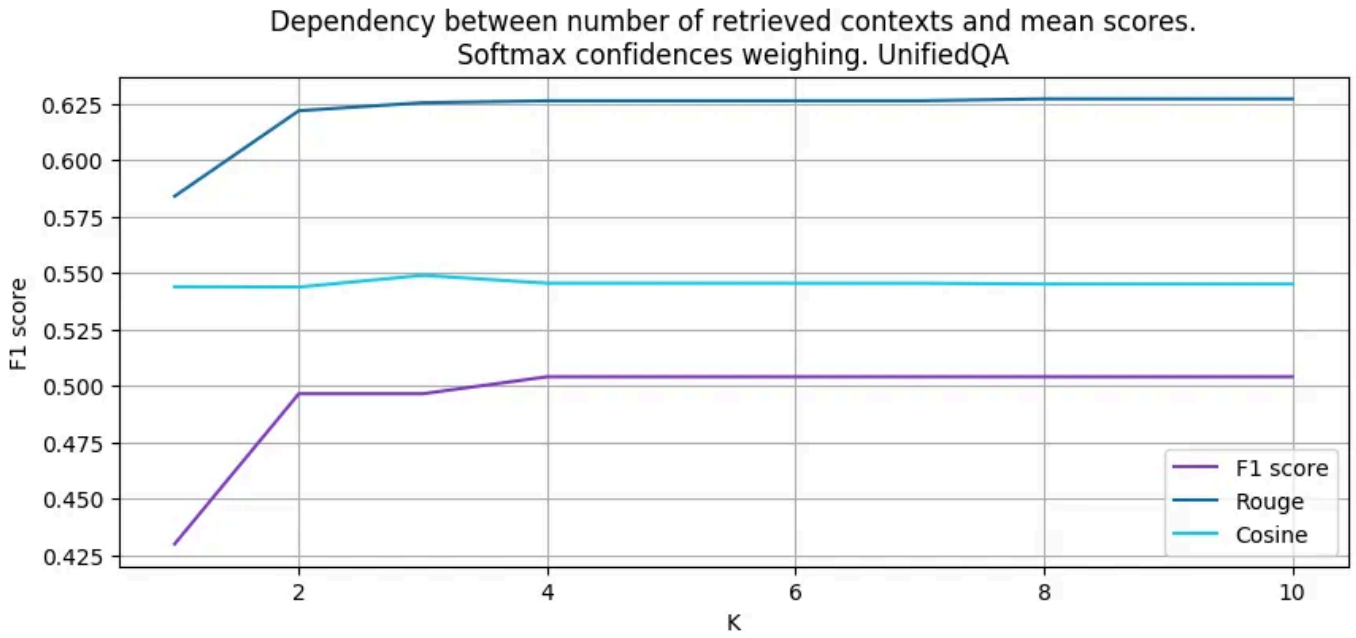


Fig. 4. Mean metrics for each count of contexts. Using Softmax weighting

We also explored the impact of question wording on the quality of bot responses. The study was carried out in two steps — for extractive and abstractive questions.

In the first step, we prepared 2 datasets — the questions in the first dataset had words and sentence structures that were exactly encountered in contexts. We replaced these words with synonyms in the second and changed the question's wording.

First, we put the retriever aside and looked at how the UnifiedQA model behaves if it has to look for an answer by synonyms. You can find mean metrics for context-based inference of different model versions in table 2. The metrics values dropped by about the same amount as if we were using a context-free pipeline.

	UnifiedQA small			UnifiedQA V2 large		
Metrics	Dataset V1	Dataset V2	Change	Dataset V1	Dataset V2	Change
F1	54.8%	45.1%	-9.7%	60.0%	51.5%	-8.5%
Rouge	67.9%	62.1%	-5.8%	74.3%	67.2%	-7.1%
Cosine	55.3%	54.6%	-0.7%	56.0%	54.0%	-2%

Table 2. Models' evaluation for context-based inferences.

Figure 5 shows the rank the retriever gave for the correct context for each question from datasets 1 and 2. We can see that the retriever is not affected as much by replacing words with synonyms.

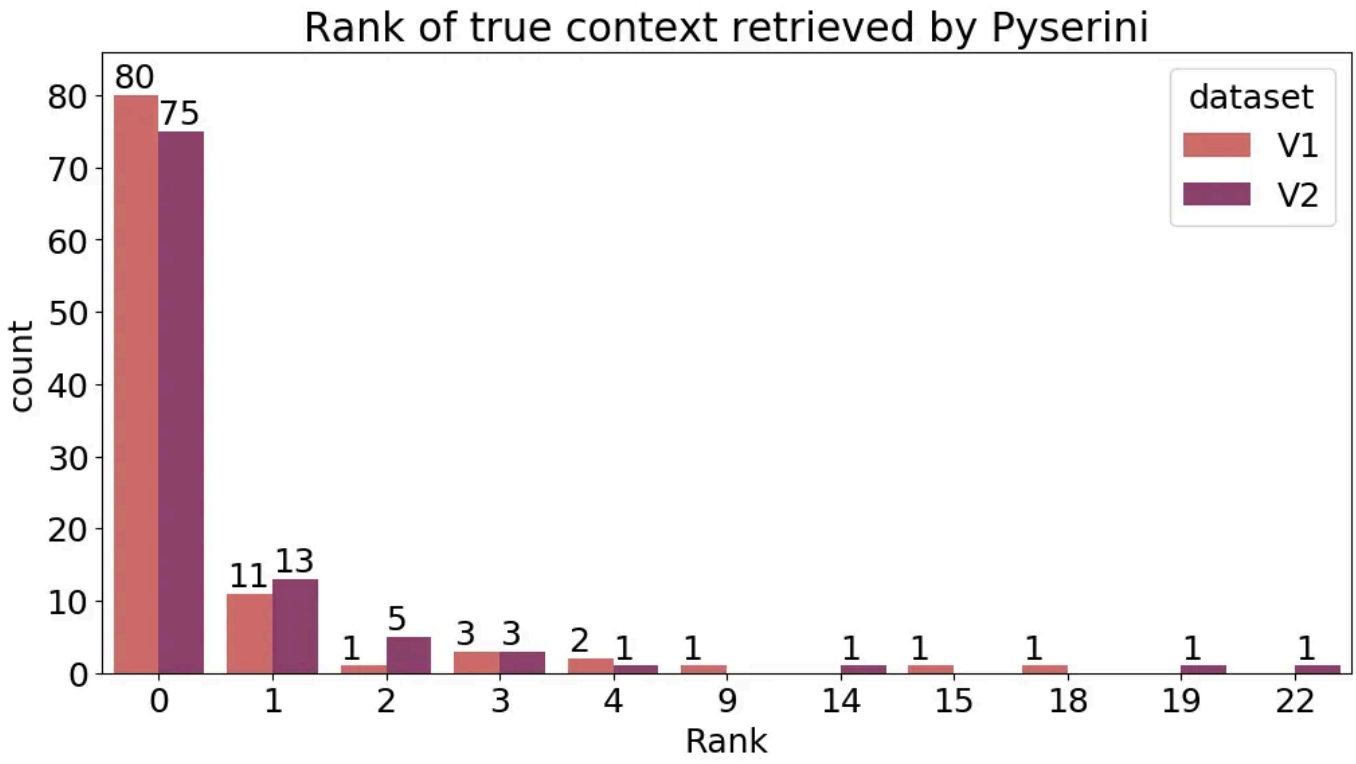


Fig. 5. Count of ranks given by the retriever to the correct contexts for two versions of questions

In Table 3, you can see the comparison for context-free inferences of UnifiedQA.

	UnifiedQA small			UnifiedQA V2 large		
Metrics	Dataset V1	Dataset V2	Change	Dataset V1	Dataset V2	Change
F1	50.4%	40.0%	-10.4%	53.7%	43.8%	-9.9%
Rouge	62.6%	54.3%	-8.3%	68.1%	60.6%	-7.5%
Cosine	54.6%	52.1%	-2.5%	55.2%	52.7%	-2.5%

Table 3. Models' evaluation for context-free inferences.

Thus, our Q&A bot suffers partly from using synonyms and reformulations, although it does not break the whole service.

Results

After building indices of contexts collected on the Quantum site using Pyserini and implementing the application scheme, we got a Q&A system with a web interface that can answer questions about our site. You can see an example in figure 6.

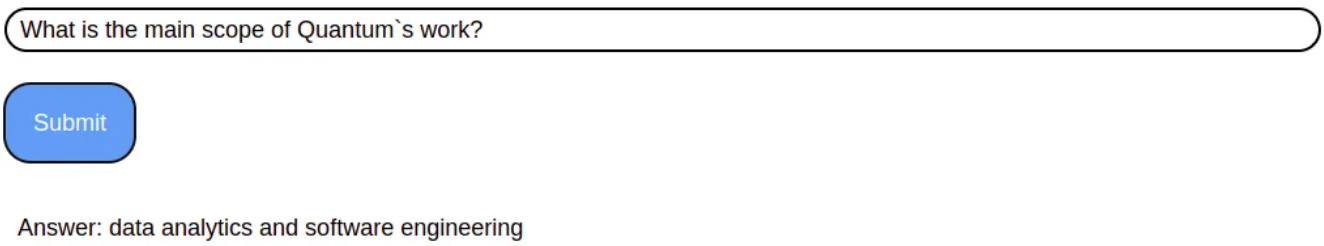


Figure 6. Web GUI and usage example.

Speaking about the restrictions of the use of this bot, we can highlight several important remarks:

- The wording of the question affects the quality of the bot. The presence in the question of words and speech constructions that exactly coincide with the necessary part of the context increases quality. Mainly this applies to the work of the Q&A model but also to the retriever. At the same time, with an increase in the model's complexity, replacing words with synonyms still affects the quality.
- Contexts should be clear and make distinct sense. Contexts that are too abstract cause the model to produce illogical responses.
- The bot cannot collect responses from multiple contexts into one logical response. Also, the QA model does not respond well to questions if parts of the answer are placed in different parts of the context.
- A more complex model answers abstract question more logically, although not always correctly.
- Text formatting is critical to the quality of responses. All contexts and questions must be converted to the correct register in production. No

uppercase is allowed if it's not an acronym. It is also desirable to clean the texts of garbage characters.

Written by Ruslan Babudzhan, proofread by Kostyantin Isaienkov

Bots Q And A NLP Naturallanguageprocessing Semantic Search

Written by Quantum

894 Followers · Writer for Geek Culture

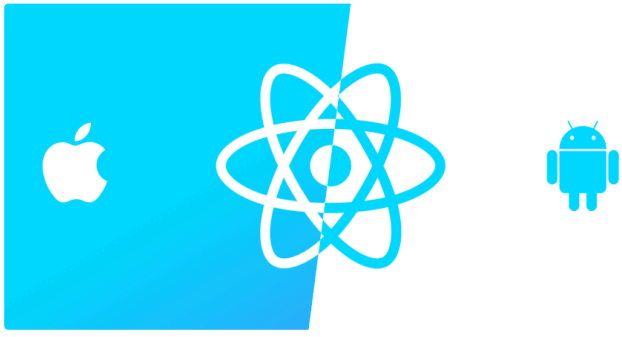
We help companies solve their data challenges.

Follow

More from Quantum and Geek Culture

Quantum in Geek Culture

Crop field boundary detection: approaches and main challenges



Anshul Borawake in Geek Culture

React Native Generate APK— Debug and Release APK

Today, we have powerful tools to improve agriculture management and performance...

9 min read · Jun 22, 2022

72

Generate Debug and Release APK in React Native; Windows, iOS and Linux

3 min read · Apr 3, 2021

1.7K 11

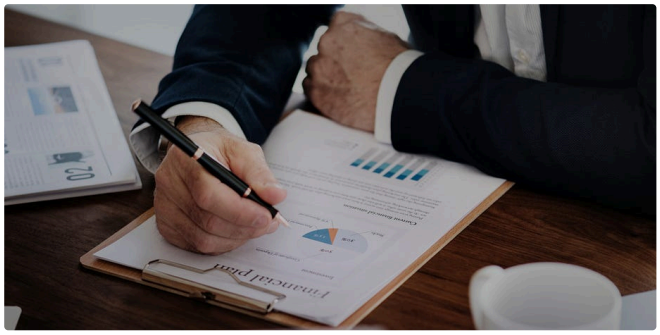
Pooya Amini in Geek Culture

Left Amazon after 7.5+ years; Here is my honest review.

Reasons to like and hate the most about working at Amazon.

9 min read · Nov 11, 2022

4.3K 67



Quantum in DataDrivenInvestor

Four Types of Data Analysis that Improve Decision-Making

This article outlines four main types of data analysis and explains how they help you...

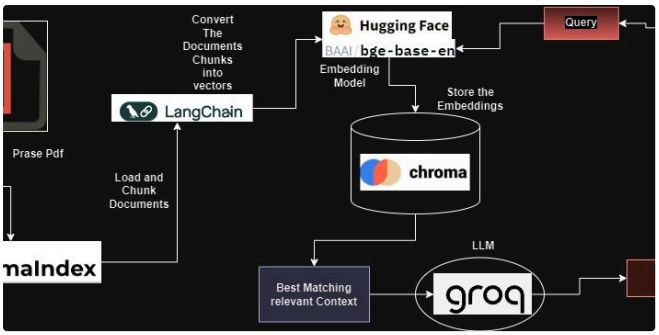
7 min read · Apr 27, 2021

151 1

See all from Quantum

See all from Geek Culture

Recommended from Medium



Plaban Nayak in The AI Forum

RAG on Complex PDF using LlamaParse, Langchain and Groq

Retrieval-Augmented Generation (RAG) is a new approach that leverages Large Languag...

13 min read · Apr 7, 2024

436 3

Rahul Nayak in Towards Data Science

Text to Knowledge Graph Made Easy with Graph Maker

An open-source library for building knowledge graphs from text corpus using...

10 min read · 5 days ago

739 8

Lists

Natural Language Processing

1444 stories · 942 saves

The New Chatbots: ChatGPT, Bard, and Beyond

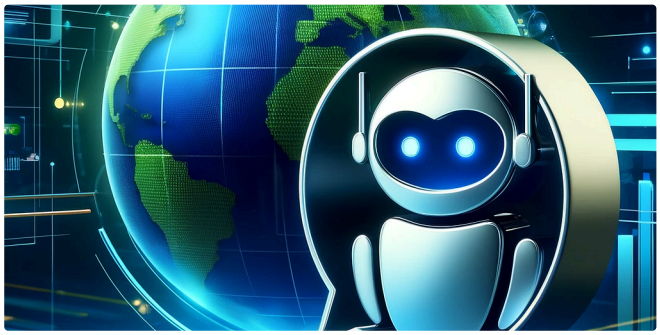
12 stories · 373 saves

data science and AI

40 stories · 152 saves

Generative AI Recommended Reading

52 stories · 1032 saves



Paul Iusztin in Decoding ML

The 4 Advanced RAG Algorithms You Must Know to Implement

Nuno Bispo in Django Unleashed

How to Build an AI Chatbot for Q&A on Any Website

Implement from scratch 4 advanced RAG methods to optimize your retrieval and post-...

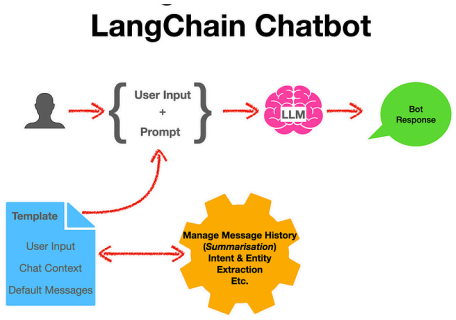
15 min read · May 4, 2024

1.2K 6

This article delves into the development of a chatbot, designed to read a website's conten...

· 11 min read · Apr 5, 2024

106



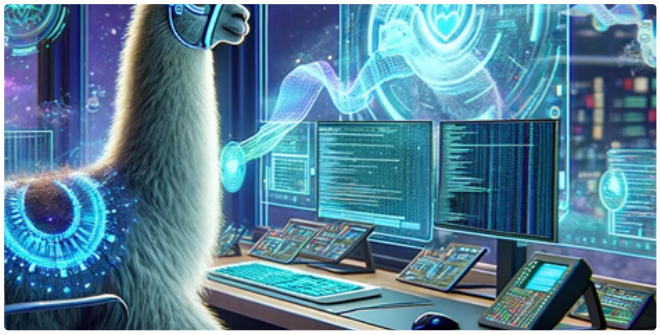
Cobus Greyling

Building The Most Basic LangChain Chatbot

In this article I consider what the basic building blocks are of a LLM-based chatbot,...

6 min read · 6 days ago

110



Ryan Siegler in KX Systems

RAG + LlamaParse: Advanced PDF Parsing for Retrieval

The core focus of Retrieval Augmented Generation (RAG) is connecting your data of...

7 min read · May 3, 2024

107

See more recommendations