Building a Chatbot using Frequently Asked Questions with Microsoft QnA Maker

Catherine Samuel Rajkumar
0888987
csamuel1@lakeheadu.ca
Lakehead University

Disha Sanaboina
0858900
dsanaboi@lakeheadu.ca
Lakehead Univeristy

Harika Kudarvalli
08807033
hkudarva@lakeheadu.ca
Lakehead University

Abstract*— Customer service is an important benefactor in the success of businesses around the globe. Consumers often tend to confuse on the utilization of the product and tend to ask some random questions frequently. As much as the services given by call centers, it takes an irrational amount of time to solve simple queries. Chat bots have successfully led users to clarify any queries at any hour of the day, increasing the efficiency of the companies. These chat bots collect data based on the FAQ's asked by users on subjects related to the product or services provided by the enterprise. The chat bots respond within no time to the user's query and store any new data collected for future analytics. The aim is to research the backend functionality and logistics that goes behind the application of chat bots and apply the same in a user specified system.

***Keywords*— Chatbot, FAQ, Question Answering, LUIS, Cloud, Active Learning**

I. Introduction

A bot in general terms is a software (group of applications) that execute automated tasks over the internet. In a certain way, a bot is used to carry out tasks which are simple and repetitive tasks and are both time consuming and impossible for humans to execute. As bots are specifically designed for productive tasks, frequently they are prone to malicious attacks. The term "bot" is derived from robot. When a bot functions through the internet, it is called www.robot or web robot. Problems faced when developing a bot are:

1. Sending/Receiving messages from a source.
2. Understanding user's (other party's) perspective.
3. Respond accordingly.

Sending/receiving messages as HTTP requests, performing NLP or regex on user input, identifying the state of conversation and to trigger a code as a set of response are some of the functions the bot performs unsupervised. There are many types of bots developed by various organizations. One such example is the Microsoft Bot Builder.

1. It supports Programming languages such as .NET, Node and C#.
2. Establishes a Conversation medium i.e., send/receive messages.
3. Consolidates platform events.
4. Can connect to various platforms.
5. Uses "Write Once and run anywhere" as its backbone.
6. Well suitable for platform independent products which are also called as "platform agnostic" products.

The Microsoft Bot framework includes a set of services, tools and SDKs that are powerful enough to build a bot of user's choice and to connect intelligent bots on a rich platform or "framework". Developers can create a bot, integrate and manage them at a large scale by using Microsoft Bot Framework as their foundation. It gives developers an additional advantage of dramatically optimizing the amount of code and coordination that are required to develop "enterprise-ready" bot experiences.

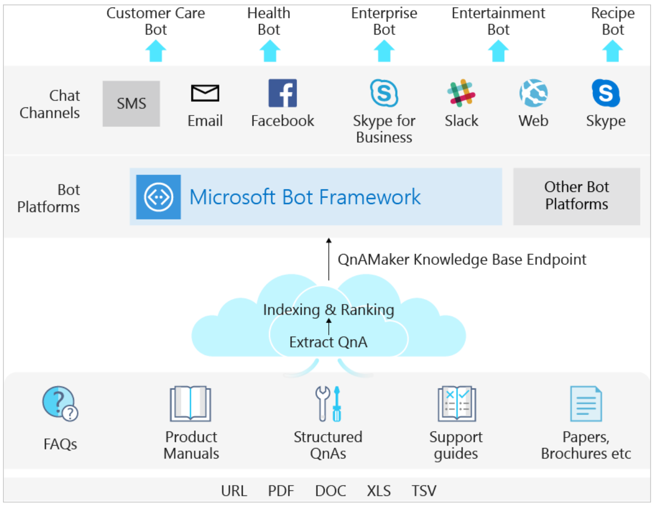

Fig. 1. Different types of bots and layers of Architecture.

QnA Maker is a product of Microsoft which allows us to build, train and publish a simple question and answer bot based on FAQ URLs, structured documents, product manuals or editorial content in minutes. This ready to use platform speeds the development process of a Bot as it is not started from the scratch. In order to understand human conversation, it leverages the abilities of LUIS (Language Understanding Intelligent Service) and combines it with a particular service that enables us to give a rundown of questions and answers, or just transfer an item manual or give a link to a website, and the questions and answers will be parsed for you consequently. Fig. 1 depicts the different types of bots that are available on QnA Maker.

FAQ pages on any website are considered as trustworthy and fast approach to discover the data accessible on the site instead of exploring through all the website pages. Commonly, to discover explicit data rapidly on a lengthy and grouped FAQs is tedious and dull. To make it increasingly explicit and energizing, organizations are continually rearranging and grouping questions in a superior manner yet at the same time it doesn't give an engaging client experience.

II. RELATED WORK

Auto-FAQ-Gen: Automatic Frequently Asked Questions Generation by Fatemeh Raazaghi;

In his research, he states that Frequently Asked Questions (FAQs) is a well-known method for documenting the rundown of common questions on specific topics or settings. Most FAQ pages on the Internet are static and can rapidly wind up obsolete. He proposed to expand the current work on question answering frameworks to producing FAQ records. In ordinary Question Answering (QA) frameworks, clients are just permitted to express their questions in a characteristic language position which is natural language. A new technique is proposed to build the questions by consolidating question extraction and question production strategies. The proposed framework acknowledges, extricates, or produces client questions to create, maintain, and improve the FAQs quality. Additionally, to the basis of QA framework, complementary units are added to direct the FAQ list. The research proposed here will add to the field of Natural Language Processing, Text Mining, QA, especially to give brilliant programmed automated FAQ generation and retrieval.

Design of Question Answering System with Automated Question Generation by Min-Kyoung Kim;

According to Min-Kyuoung Kim, one of the most troublesome issues in creating question-answering (QA) framework is that it is so difficult to produce characteristic language questions and to discover a response to a query question. To maintain a strategic distance from various challenges of creating QA frameworks, he proposed another style of question-answering system architecture that effectively uses sentences inside a document as a source of question/answer. Fundamentally, the proposed QA framework gives users a lot of query question for user data needs, and the applicant questions are consequently produced from huge sentences that are required to contain significant facts or events. The QA framework assembles a total database of (question, answer) combines in the wake of breaking down an entire accumulation of reports. For this, we must follow out the following steps: sentence split, named-entity acknowledgment, question generation, question separating, question/answer ordering. The significant things in the process are question generation and question filtering. Primarily, it can produce questions that ask the entities implemented from a given sentence. The question filtering is to identify critical sentences that have significant data that clients need.

Survey on Chatbot Design Techniques in Speech Conversation Systems by Sameera A. Abdul-Kader;

Human-Computer Speech technology is growing rapidly as a procedure of computer interaction. Since six to seven years, there has been an ongoing upsurge in speech-based search engines and assistants, for example, Alexa, Siri, Google Chrome, Google assistant and Cortana. Natural Language Processing (NLP) methods, for example, NLTK for Python can be connected to examine speech, and intelligent and accurate responses can be generated by designing an engine to convey human like responses. These kinds of applications are known as chatbots, which is the significant point of this study. This paper introduces a review on the strategies used to structure Chatbots and a comparison is made between various plan procedures from nine deliberately chosen papers as per the fundamental techniques received. These papers are illustrative of the noteworthy enhancements in Chatbots in the most recent decade. The paper talks about the similarities and distinctions in the procedures and analyzes specifically about the Loebner prize winning Chatbots. One of the most popular articles on Machine Learning was “Disease Prediction by Machine Learning over Big Data from Healthcare Communities” [9]. The authors used Convolutional Neural Network to build a risk prediction model. The incomplete data was reconstructed using latent factor model.

III. CHARACTERICTICS

The propose model consists of many phases from gathering information from the caller to deducing the symptoms to a predicted disease. Multiple platforms and IDEs have been used to implement this system. Methodology can be divided into four phases.

1. Dialog Management: This involves managing simple Q&A and enabling the chatbot to have complex conversations that are meaningful to the user.
2. Task Automation Capability: This helps in performing tasks for users and ensures the framework has enough dialog capabilities and that it can connect to your back-end systems.
3. Humanization: Users are more engaged in conversation if a chatbot acts more human-like. Few chatbots can detect and show emotions.
4. Reporting and Monitoring: The framework also reports and monitors the performance of the chatbots and how helpful they are to the end users/ customers.
5. Ease of Implementation: This framework allows business users to configure the chatbot themselves without the requirement of custom software development.
6. Security and Compliance: Verifies if the user needs extra security requirements. do you need to be compliant with audit regulations? These security and logging capabilities vary amongst platforms.

7. Interaction Channels: Provides various methods through which the users interact with the chatbot. Helps in choosing a platform that connects easily with your app, webchat, social media platform or voice interface.
8. QnA Maker platform allows users to upload a FAQ document or a link to a FAQ page thus creating an NLP model while exposing an endpoint to a query.
9. Intent Recognition: This gives the framework the ability to "guess" what the user is requesting, even if phrased unexpectedly. Good intent recognition is vital for user satisfaction.

IV. ARCHITECTURE

One of the major changes that happened in QnA Maker was the Architecture of the bot framework which was in GA initially. Recently, The QnA Maker architecture has been implemented in Microsoft Azure Bot framework. The two major parts of the QnA Maker stack are as follows:

1. A Control Plane that consists of major management services of QnA Maker.
2. A Data Plane as a QnA Maker runtime.

QnA Maker consists of cognitive offerings such as an amazing graphical user interface. The basic services of QnA Maker are to extract various contents from a given public URL or any documents and this helps in building a concrete and a comprehensive database with interpretable question and answer framework. And this is commonly known as the KnowledgeBase. This Knowledgebase can be made available to the outer world using a RESTful API endpoint. This can be referred as Knowledge Endpoints. The QnA Maker Portal from the architecture helps in creating a KnowledgeBase and manage the extracted content and using the different sequences of the questions to train the given data. This does not require any developer experience to create a KnowledgeBase. Management of the question and answers that includes the control panel together are referred to as KnowledgeBase. This includes both the creation and the modifying of the KnowledgeBase and so this needs to be trained after every activity in-order to present it as a cognitive service.

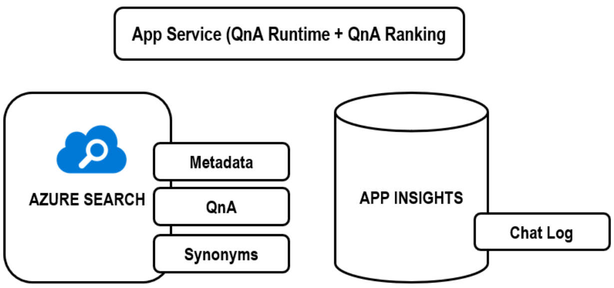

Fig. 2. Architecture of QnA Maker.

The control plane takes care of all the activities along with KnowledgeBase such as creation, training, modifying and publishing activities. These can be categorized as the QnA Management services. The management of such services can be achieved in two different ways using the QnA Maker Portal and the Management APIs. The GUI interface of QnA Maker is one of the amazing features of this service. This just requires minimal or zero developer experience. The KnowledgeBase endpoints can be integrated into any bot frameworks using different technology stacks. Few of the different chat channels that the KnowledgeBase can be integrated with are Skype, Telegram, Facebook etc. Management APIs are a set of RESTful APIs generally provided by Microsoft, with a detailed documentation and a graphical user interface. APIs can be used in .NET, Node.js, or any other application.

The data plane is a runtime component of QnA Maker. Each question and answer of the knowledgebase is stored in the Azure search. This Azure search is an AI based cloud search service for both Web and mobile app development provided by Microsoft Azure. The database also consists of every synonym, antonym and the metadata related to your question database that is stored in Azure search. There is a cloud platform as a service provided by Microsoft Azure to deploy the runtime to the Azure App service. This can be used to host, deploy, and scale several web and mobile applications into the cloud. This adds the application Insights to the architecture of QnA Maker framework that stores all the chat logs as depicted in Fig. 2. that are generated using the knowledgebase. This acts as bot or to its endpoints through any given application. This chat log helps in updating the knowledgebase thus making it more accurate and intelligent with the responses. Overall, the architecture of the QnA Maker is a reliable base built on Azure.

The required QnA Maker service and knowledgebase created provides a RESTful endpoint to be connected with the application and facilitates the integration with a bot framework.

V. METHODOLOGY

The goal is to have a FAQ with lots of questions and their answer, so the bot has a large enough knowledgebase to answer the customer's questions. Thus, the customer receives a very fast and accurate answer. Creating a bot is not an easy task, but Microsoft has developed a new service on Azure for this specific need: QnA Maker. Some of the basic steps in creating a bot involve:

1. Planning: The first step is to consider the reason for creating a bot and its behavior around the users while communicating. QnA Maker extracts FAQs from a public website and the bot will function in a request-response fashion.
2. Building: Building is a crucial step as it consists of development environment, the technology stack, and the language used for bot development.
3. Testing: All complex applications have multiple modules that work collectively to give output hence testing phase is vital for proper functioning and accuracy of the bot.
4. Publishing: The bot needs to be published to a local site or to a live environment by implementing continuous integration and deployment.
5. Connecting: This is not a mandatory phase but connecting the bot to social channels leads to increased interactions with the bot users and the target audience. As the interactions are more it results into a more accurate and efficient system.

6. Evaluating: Log data and interactions can be collected from bot for analytics purpose.

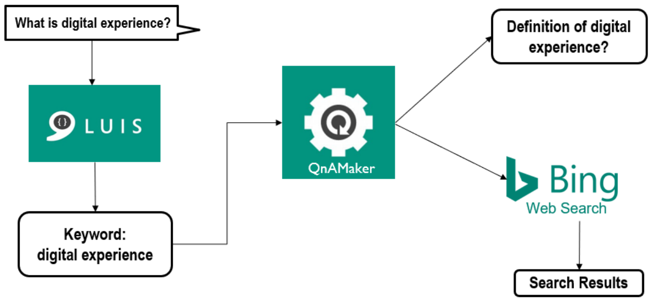

Fig. 3. Flow Chart of Methodology.

As we are discussing about creating a bot using QnA Maker, let's discuss its methodology steps briefly.

1. Create a new QnA service.
2. Indicate Question and Answers (Q&A) source using the URLs, or by uploading files, or by entering manually.
3. Update your knowledge base as per the need for correct Q&A mappings.
4. Train your knowledge base: The train feature of QnA Maker service allows you to check the correctness and relevance of the responses. With this feature, you can quickly correct the responses and re-train the knowledge base as per need.
5. Finally, when you are satisfied with the knowledge base, you need to publish the knowledge base as service.
6. QnA Maker also allows you to share your knowledge base with other users, so they can also contribute to it as well.

A. *Language Understanding Intelligent Service (LUIS)*

LUIS is a service created by Microsoft. LUIS is a cloud-based API service that applies custom machine learning intelligence to a client's conversation. It has complex algorithms to comprehend human language and extracts intents with a confidence score that can be utilized to execute explicit code instructions in the customer application. Language Understanding Intelligent Service (LUIS) enables developers to build smart applications that can understand natural language and respond accordingly to user requests.

As shown in the example above the client application takes user input as the Utterance and sends to the LUIS endpoint as an HTTP request. LUIS has certain variables to extract from the user input in order to understand the language.

- Utterance: The text given by the user are generally termed as utterances and are sent from client application to the LUIS.
- Intent: The main objective of the sentence is defined as Intent.
- Entity: An entity represents an instance of a class of object that is relevant to a user’s intent.
- Label: Label is like entity but tagging the entity with a separate class is called label.

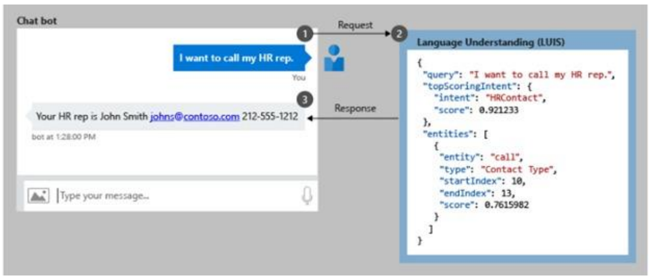

Fig. 4. Example of LUIS.

Taking the example given in Fig. 4, LUIS applies the model that is already learned, to the human language content to give intelligent understanding about the client input. LUIS returns JSON formatted response, with a top intent, "HRContact". The base JSON endpoint reaction contains the question utterance, and the top scoring intent. It can likewise extract information, for example, the Contact Type entity. After a LUIS model is distributed and gets genuine client utterances, LUIS gives several methods to improve accuracy prediction such as Active learning of endpoint utterances, phrase lists to include domain words, and examples to diminish the quantity of utterances required.

B. *Active Learning*

One of the important aspects of enabling Active Learning in QnA maker helps us to improve our knowledge base by suggesting alternative/multiple questions to the question and answer pairs. This is purely based on user-submissions that we receive from the Knowledge Base. The developer or the user can easily review those suggestions and can edit the knowledge base on-the-go by either adding them to existing questions or even rejecting the alternative questions.

In order to make changes in the knowledge base, the suggestions must be accepted. The knowledge base doesn't change automatically. These suggestions do not affect the existing question and answer pairs in the base. The user can easily edit or delete the queries and accept new suggestions as well. Active learning can be triggered based on the confidence scores of the top answers returned by QnA Maker framework. The query can be a possible suggestion if the range of the confidence scores of the answers is within a small range, for each of the possible QnA pairs. Once this suggested query is accepted for a specific QnA pair, it is rejected for all the other pairs. After accepting suggestions, we must save and train the knowledge base. When the endpoints are getting a reasonable quantity and variety of usage queries, Active learning mechanism gives the best possible suggestions. Every 30 minutes, when 5 or more similar queries are clustered, the QnA Maker suggests the user-based questions to the knowledge base designer to either accept or reject the suggestions. All these automatic suggestions are clustered together by similarity.

The top suggestions for alternate questions are displayed based on the frequency and the confidence scores of the queries by end users. Once questions are suggested in the QnA Maker portal, the user can either review and accept or reject those suggestions. This works without any back-end API to manage suggestions.

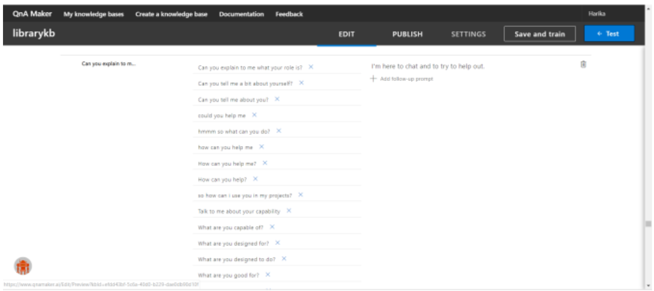

Fig. 5. Active Learning.

In the above figure, we see the screenshot of our knowledge base along with the question and answer pairs. When active learning is enabled in the KB, it automatically suggests alternative questions based on the confidence scores. Here, we see the different suggestions for a single query. The user can accept the suggestions or simply reject them by tapping the X mark. And by tapping the delete icon, the user can easily delete the query answer pair as well from the knowledge base. And on taping save and train button, the suggested questions taken into consideration are fed into the QnA maker chatbot.

By default, active learning will be switched off and the user must manually enable active learning to incorporate it to their knowledge base and to see suggested questions. Once active learning is turned on, information must be sent from the client app to the QnA Maker.

1. Select the Publish button to publish the knowledge base into the QnA Maker. Queries from the Active Learning are collected from the GenerateAnswer API prediction endpoint only
2. Select 'Service Settings' that displays while tapping on the user's name from the top right corner of the QnA maker portal.
3. Find the QnA Maker service then toggle Active Learning. This feature can be disabled by toggling this setting again.
4. Select View Options on the Edit Knowledge base page, in order to see the suggested questions and then select Show active learning suggestions.
5. The user can also filter the knowledge base with question and answer pairs by selecting Filter by Suggestions, to show only suggestions.
6. The alternate questions are displayed with a check mark, ✔ to Each QnA pair with to accept the question or an x to reject the suggestions. Select the check mark to add the question.
7. Select 'Save and Train' button to save the changes to the knowledge base.
8. Select 'Publish' button to allow the changes to be available from the GenerateAnswer API.

Active Learning can used in two ways:

i. Implicit Feedback

The developer understands when a user question has multiple answers with confidence scores that are similar and considers this as feedback. No action is required to enable this feedback. The implicit feedback uses an algorithm that determines the confidence score proximity and makes active learning suggestions. This algorithm is a complex calculation.

For example, When the user's question has a high confidence score, such as 80%, where the range of scores are wide, approximately within 10%. As the confidence score decreases, the range of scores decreases that is considered for Active Learning.

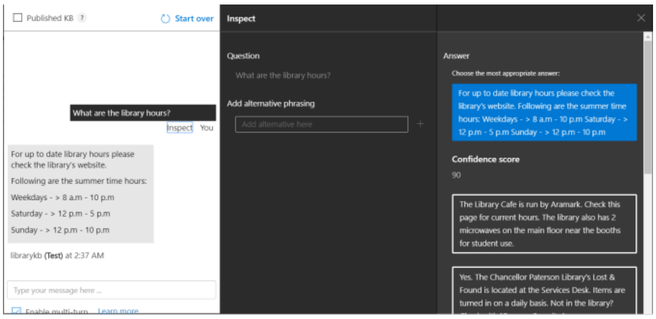

Fig. 6. Implicit Feedback.

From Fig. 6, we see the chatbot with the user query and the relevant answer for the query. On tapping the Inspect button, the knowledge base developer will be taken to the Inspect page where the user query is displayed along with a confidence score beneath the question. Along with the confidence score, we can also select alternative answer for the same question that has a similar confidence score.

The user can also add alternative questions from the inspect page. When the confidence scores for multiple answers are similar for the same question, the implicit feedback comes into play. Inspecting the question plays an important role in understanding the confidence score.

ii. Explicit Feedback

When multiple answers with little variation in confidence scores are retrieved from the knowledge base, the client application asks the user to select the correct question from the list of suggested questions. The explicit feedback from the queries asked by user is fed to the QnA Maker with the Train API.

Fig. 7 depicts the chatbot wherein the user asks a query to the chatbot, with just a keyword, for example, 'Password', for which the knowledge base has multiple questions for the same keyword. During such situations, the QnA Maker along with the active learning feedback prompts the user to choose any one of the relevant questions from the list provided. And on tapping either one of the suggested questions, the user will be provided with the relevant answer to the suggested question immediately.

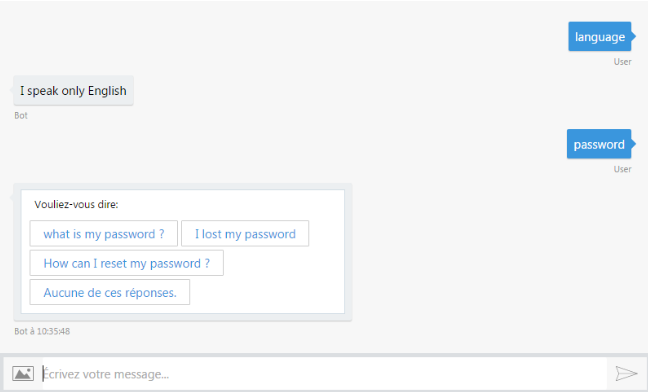

Fig. 7. Explicit Feedback.

VI. Dataset

As QnA Maker works completely on question and answering system, the data being fed should be in that format. Different FAQs have multiple format like html, doc or pdf. Our dataset has been retrieved from Lakehead University Library webpage as shown in Fig. 8. The FAQs were converted into a Microsoft document. At this stage we can do multiple changes to the document, but QnA Maker has an option of doing itself.

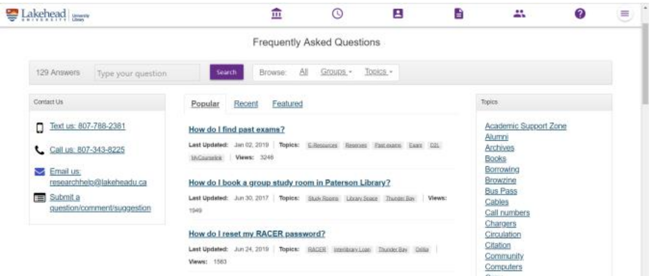

Fig. 8. LU Library FAQ Website.

Once the data is fed to QnA Maker, information divided into question and answer pairs as given in Fig. 9. Questions can be edited manually, and paraphrased questions can be added for improving the accuracy of Knowledgebase. Answers on the right side can also be edited and henceforth add multiple answers.

According to our research, QnA Maker cannot process complex documents like webpages (buttons, nested structures). So, feeding a document or webpage with simple textual format is hassle free for QnA Maker.

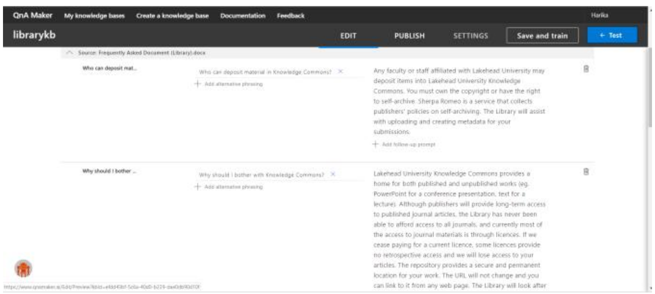

Fig. 9. Library Knowledgebase.

VII. Implementation

Once the customer sends a query to the chatbot, some of the internal functions get activated. This process is clearly depicted Fig. 10. Following are some of the steps involved in query processing.

1. The client application sends the user query to the GenerateAnswerAPI.
2. Qna Maker preprocessing the user query with language detection, spellers, and word breakers.
3. This preprocessing is taken to alter user query for best search results.
4. This altered query is sent to Azure Search Index, receiving the top number of results.
5. QnA Maker applies advanced featurization to determine the correctness of the fetched Azure Search results for user query.
6. The trained ranker model uses the feature score, from step 5, to rank the Azure Search results.
7. The new results are returned to the client application in ranked order.

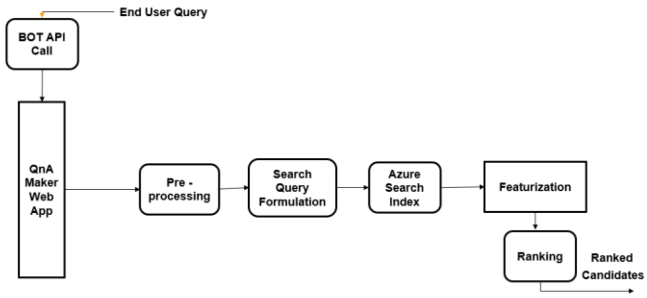

Fig. 10. Query Processing

Training and testing are an iterative process. Training involves editing both question and answers in respect to the content of the chatbot. After training LUIS app, we test it with sample utterances to see if the intents and entities are recognized correctly. If not, make updates to the LUIS app, train, and test again.

Testing in QnA Maker can be performed in two ways:

1. Interactive testing enables you to test both the current and published versions of client's app and compare their results on one screen. Interactive testing runs by default on the current trained model only. For a published model, interactive testing is disabled and needs client's action to enable it because it is counted in hits and will be deducted from your key balance.
2. Batch testing allows you to validate the active, trained model's state with a known set of labeled utterances and entities. In the JSON-formatted batch file, add the utterances and set the entity labels you need predicted inside the utterance

VIII. Comparison Study

A study based on the top 25 of the best-known platforms for building chatbots were made; and the results have been mentioned below:

1. IBM Watson: It is built on a neural network and has intents, dialog and entities as three main components. It performs on various integrating programs such as Node SDK, Python SDK, iOS SDK, Unity SDK. It is widely

used in healthcare, legal, finance and retail market for its ease of use.

2. Pandorabots: It is based on AIML (Artificial Intelligence Markup Language) which includes A.L.I.C.E as the supporting framework. It functions on SDKs, JAVA, Node.js, Ruby and PHP. Padorabots appear in messaging and Native apps, Games, social networking sites, virtual assistance, e-learning and Academies use this platform extensively for research and teaching.

Extraction of QnA's from content and expanding the list of supported file and HTML formats can be easily done by the QnA Maker service which continually improves the underlying algorithms based on which it's been created. Configuring multi-turn extraction can be enabled with the user's knowledgebase. If there is a need for the hierarchy for the question pattern, this hierarchy can be extracted from the document created after extraction. Collaboration is much easier with the QnA platform with a knowledgebase. In order to access the knowledgebases, users need access to the Azure QnA Maker resource group. QnA suggests alternative questions when it has a wide range of quality and quantity of user-based queries. With QnA Maker portal, the user can filter by suggestions and then the user can review and accept. The user can also reject those suggestions. The ranking algorithm of QnA platform matches a user query with a question in the knowledge base. This also avoids the repetition of the same word set between questions thus reducing the likelihood that the right answer is chosen for a given user query with those words. QnA Maker can add synonyms at the service-level, and this is shared by all knowledge bases in the service. Chit-Chat are the conversational pairs provided by Microsoft that are already pre-written. This makes it a lot easier to get a certain bot running quickly without having to prepare it with user induced small talk procedure. Chit-chat, thus, ends up in saving a lot of time for the user.

IX. Results

Various journals have been referred to in this research to study the backend functionality of chat bots. These chat bots operate on numerous FAQ's asked by the user around the globe in order to find a suitable answer to the user query. In recent times, big platforms such as Microsoft, Amazon, Apple have initiated their very own chat bots naming Cortana, Alexa and Siri which have been a hit among the users, proving the efficiency of chat bots. LUIS [Language Understanding] is a machine learning-based service which enables developers to build natural language understanding in bots. Microsoft Azure provides another platform to develop bots in a one stop shop in its featured QnA Maker. It enables the user to plan, program, develop and initiate a bot, also enabling the bot to constantly evolve based on the user feed data [FAQ].

In this experiment we have designed a library management website for our university. Sample screenshot of the website is shown in Fig.11. The website has been built on HyperText Markup Language.

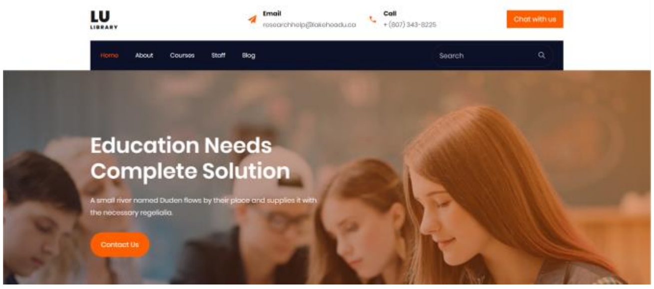

Fig. 11. LU Library Website.

The main idea behind this is to include a chatbot in the web page. In Fig. 12. A screenshot of our chatbot has been included. Initially chatbot is responding with a formal greeting to the user. When asked about "library hours", the reply is very clear and precise. This model can be trained and tested many times o the QnA Maker portal to get better accuracy as discussed above in different sections.

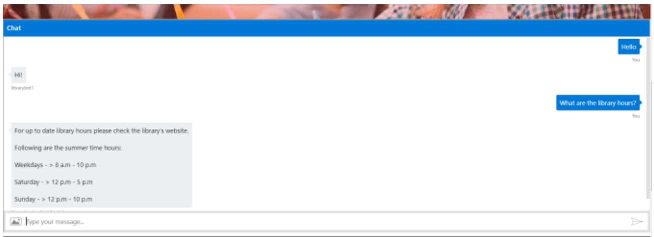

Fig. 12. Library Chatbot.

X. Future Work

We would like to research on other chatbots by multiple companies and compare their results. In this project we have integrated our chatbot with a library web application. We would like to integrate this same chatbot to a Facebook page and connect it to different social channels. This will allow the page itself to answer questions to the viewers without any human intervention.

XI. Conclusion

Our research study has proved that QnA Maker is one of the leading chatbot frameworks which is available free of charge. Using QnA Maker, any user can easily develop and deploy their own chat bot for their enterprise. Though this platform is widely recognized for its efficiency, it lacks in storage space of knowledgebase limiting it to 20 MB. QnA maker is widely capable of processing data from a public URL, but in order to process complex unstructured data such as a video or an image file, more storage space is required. There is also a limitation to the number of transactions per minute. Integrating a FAQ chatbot into a web application is much easier because it does not involve an extensive knowledge of the platform and licensing issues are less compared to integrating the chatbot into any social channels like Facebook or Twitter. Although, with all its limitations, QnA Maker presented by Microsoft Azure is an easy and user-friendly approach in the development of chat bots. Our main aim through this project research was to integrate the Microsoft Bot framework with QnA maker into the university library website that we've created. We have created the knowledge base in the QnA maker portal by retrieving the questions and answer pairs from the Frequently Asked Questions page of the University Library website. With modifications and updates on the way, the future for the development of chat bots through QnA Maker looks impeccable.

ACKNOWLEDGEMENT

We would like to thank Dr. T. Akilan for guiding us through this research study and enlightening us with in depth knowledge. We are also thankful to Mr. Shiqi Wang for his supervision as Teaching Assistant.

REFERENCES

[1] "Create a QnA Bot with Azure Bot Service v4" web site: www.docs.microsoft.com

[2] Min-Kyoung Kim and Han-Joon Kim, "Design of Question Answering System with Automated Question Generation", 2008 Fourth International Conference on Networked Computing and Advanced Information Management \newline

[3] Sameera A. Abdul-Kader and Dr. John Woods, "Survey on Chatbot Design Techniques in Speech Conversation Systems", IJACSA, Vol. 6, No. 7, 2015.\newline

[4] "LearnAI-Designing and Architecting Intelligent Agents" web site: www.azure.github.io/

[5] Raazaghi F. (2015) Auto-FAQ-Gen: Automatic Frequently Asked Questions Generation. In: Barbosa D., Milios E. (eds) Advances in Artificial Intelligence. Canadian AI 2015. Lecture Notes in Computer Science, vol 9091. Springer, Cham

[6] Matt Wade, "QnA Maker best practices" web site: www.blog.getbizzy.io \newline

[7] Kamal Kanth, "Everything about Microsoft QnA Maker" web site: www.advaiya.com

[8] "Hidden Gem in Microsoft Bot Framework - QnA Maker" web site: www.ankitbko.github.io