The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Agent Assist: Automating Enterprise IT Support Help Desks

Senthil Mani,[1] **Neelamadhav Gantayat,**[1] **Rahul Aralikatte,**[1] **Monika Gupta**[1]
{sentmani, neelamadhav, rahul.a.r, monikgup}@in.ibm.com

Sampath Dechu,[1] **Anush Sankaran,**[1] **Shreya Khare**[1]
{sampath.dechu, anussank, shkahre4}@in.ibm.com

Barry Mitchell,[2] **Hemamalini Subramanian,**[2] **Hema Venkatarangan**[2]
bcm@us.ibm.com, {hesubram, hema.sudarshan}@in.ibm.com
[1]IBM Reseach AI [2]IBM Global Business Services

Abstract

In this paper, we present Agent Assist, a virtual assistant which helps IT support staff to resolve tickets faster. It is essentially a conversation system which provides procedural and often complex answers to queries. This system can ingest knowledge from various sources like application documentation, ticket management systems and knowledge transfer video recordings. It uses an ensemble of techniques like question classification, knowledge graph based disambiguation, information retrieval, etc., to provide quick and relevant solutions to problems from various technical domains and is currently being used in more than 650 projects within IBM.

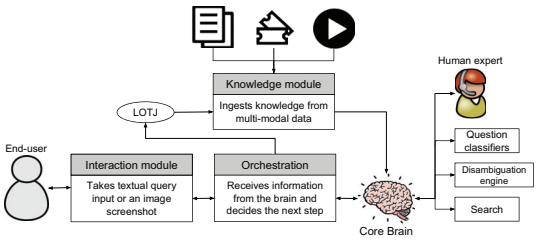

Figure 1: The architecture of Agent Assist explaining the key modules involved in building the system.

1 Introduction

Answering complex questions has been one of the long standing challenges of artificial intelligence (Clark et al. 2016). It is, in general, considered hard for even human experts as it warrants understanding large volumes of multimodal data and also resolution of the input query context. Automating such a complex question answering system has been a grand challenge for decades (Oh et al. 2016). In this paper we present a novel system, Agent Assist, which is a cognitive natural language question answering engine automating enterprise IT support help desks. In the domain of IT support, the end-user (customer) expects an answer irrespective of the method employed to fetch it. Also, the information provided by the customer about the problem may not be complete resulting in a wrong diagnosis by the agent. The use of an automated system enhances the efficiency of support agents and also substantially reduces the cost incurred in the maintenance process. Keeping these issues in mind, we demonstrate three key aspects of our system:

1. a real time knowledge ingestion module, that accepts large corpora of textual documents, ticket logs, and videos
2. an interaction module, that accepts a query from the end-user as textual input or an image screenshot
3. an orchestration module, that controls the brain of the system. The brain is an esemble of QA classifiers, a knowledge graph based disambiguation engine, a search system and a human expert in the loop as a last resort to ensure a relevant answer is given to the end-user.

This system is one instantiation of the Cognitive Automation platform which encompasses the previously mentioned components and others. The main aim of this system is to automate the tasks of human help desk agents and fallback on them only when absolutely necessary.

2 Technical Architecture

Agent Assist has five major components as shown in Figure 1: i) The **core brain** of the system learns from vast corpora of technical question-answer (QA) pairs ii) The **knowledge ingestion** module reads data from various sources and converts each chunk of information into a set of QA pairs iii) The **interaction module** either takes a textual query as input or a screenshot of an error and extracts relevant information from the image iv) The **orchestrator** controls the brain and decides how information flows within the system v) The Learning on the Job (**LOTJ**) component continuously learns from user feedback and the conversations between end-user and human agents.

2.1 Knowledge Ingestion

Agent Assist can currently ingest knowledge from four different data sources, namely: (i) manually curated QA pairs from human experts, (ii) history of tickets logs, (iii) FAQ, user manuals in HTML, *.docx* and *.pdf* formats, (iv) audio-video documents explaining technical processes and knowledge transfers in *.mp4* and *.avi* formats.

2.2 Interaction Module

The user can interact with our platform in two ways: text and images. Users can ask a question or can upload a screenshot of the error they are facing. The output of the system is always rich text. This module contains the following components:

iSight If the user chooses to upload an error screenshot, this component extracts information about the error and certain UI artifacts which may help the brain in resolving the error

Dialog This is a standard dialog component which takes in user's natural language inputs and converts them into a structured and annotated format understandable by the orchestrator. It also takes in output from the orchestrator and generates natural language text for the user to read

2.3 Core Brain

This module uses the knowledge ingested in the previous stages to provide answers. There are three components in this module, each of which takes a different approach to tackle the user's questions.

QA classifiers We can look at QA as a question classification problem. One or more questions in the ingested QA pairs can have the same answer. The first question for which a new answer is written/ingested becomes a 'primary question' and all other questions which have the same answer become it's 'alternate questions'. Each primary question and it's corresponding alternate questions form a class. The purpose of these classifiers is to classify incoming questions into one of these classes. A standard SVM and an advanced CNN classifier are built to suit the needs of different users.

Disambiguation engine Sometimes the users' questions are not complete. Additional information/context is required before the classifiers can classify the question correctly with a high confidence score. A sophisticated disambiguation module is built to deal with such cases of incomplete data. A knowledge graph is built using the ingested data and is used by the disambiguation engine to identify the gaps in the user's question and ask for relevant information.

Search If a question cannot be classified with confidence even after getting missing information from the user, a simple tf-idf and bag of words based search system is used to find documents which may contain relevant information.

2.4 Orchestration

The orchestrator acts as a middleman between the user and Agent Assist. It communicates with all the components in the brain to get relevant answers. It also decides what to show to the user and gathers user feedback at every step to identify it's shortcomings. The major steps taken by the orchestrator when the user asks a new question are as follows: i) The user either uploads a screenshot or enters a question which is then pre-processed to a structured format and given to the orchestrator ii) The orchestrator initially contacts the classifiers to get top-k answers for the question asked and their confidence scores iii) If the confidence of an answer is greater than a threshold, the answer is shown. Else, the disambiguation engine is called and the orchestrator asks counter questions to the user to clarify the context iv) After clarification, if the confidence of the classifiers increase above the threshold, the answer is shown to the user. Else, the search module is invoked vi) At any time, if the user is not happy with the answer provided, the orchestrator does a smooth hand-off to a human agent

2.5 Learning on the Job

The purpose of this module is to identify new QA pairs which can be harvested from interactions with the system. It does so in two ways: i) whenever the search module is invoked and the user provides a positive feedback on a document/audio-video chunk, the LOTJ module associates the question asked by the user to this chunk and forms a new QA pair and updates it's associated weight. If this weight becomes greater than a pre-defined threshold, the QA pair is ingested into the system ii) when the user is dissatisfied with the answers given by the system and a hand-off is made to the human agent, the LOTJ component listens to the conversation between the user and the agent to identify potential QA pairs, updates the associated weights and the same process is followed as before.

At any point in time, the support agents can see the potential QA pairs created by the LOTJ module. They can then edit them if required and approve/reject them for immediate ingestion. Thus, this module provides a non-intrusive way of accumulating new knowledge.

3 System Impact

There are more than 650 internal projects using this system within IBM with more than 5800 users. The system has thus far ingested more than 300K QA pairs. The success of this sytem has resulted in other assistance solutions being developed within IBM using the same underlying components. Some of these solutions are:

1. Employee Assist: Assistance to business users; helping them tackle day-to-day issues they face while interacting with various applications (like HR portal, Travel portal, etc) and there by reducing the overall volume of incidents raised
2. Coding Assist: Assistance to SAP ABAP programmers to help improve their productivity in development of custom ABAP modules
3. Dynamic Automation: Automating the execution of typical services orchestrating across robotic process automatons, thereby reducing the need for a human in the loop.

References

Clark, P.; Etzioni, O.; Khot, T.; Sabharwal, A.; Tafjord, O.; Turney, P. D.; and Khashabi, D. 2016. Combining retrieval, statistics, and inference to answer elementary science questions. In *AAAI*, 2580–2586.

Oh, J.-H.; Torisawa, K.; Hashimoto, C.; Iida, R.; Tanaka, M.; and Kloetzer, J. 2016. A semi-supervised learning approach to why-question answering. In *AAAI*.