Michael Ambaye

Improving Search Engine Results Using Different Machine Learning Models and Tools

Master's thesis of mathematical information technology

December 10, 2020

University of Jyväskylä

Faculty of Information Technology

Author: Michael Ambaye

Contact information: ambaye.michael@gmail.com

Supervisors: Oleksiy Khriyenko

Title: Improving Search Engine Results Using Different Machine Learning Models and Tools

Työn nimi: Finnish title

Project: Master's thesis

Study line: Web Intelligence and Service Engineering

Page count: 60

Abstract: The aim of this thesis is to provide viable methods that can be used to improve the return position (RP) of a relevant document when a natural language query (NLQ) is applied by a user.

For the purpose of demonstration, we will be using IBM's Watson Discovery Service (WDS) as a search engine that uses supervised machine learning. This feature of WDS enables a user to train the tool so that it can learn to associate the language used in the NLQ to the language used in documents labelled as relevant. Therefore, instead of mapping an NLQ to the relevant document, it will build a model that works in such a way that similar language used in the natural language query will be associated with documents containing similar language as the document labeled as relevant.

The search engine works in such a way that it first searches for the first 100 documents and then ranks the documents based on the training examples provided by the user. In other words, the training example is only applied after the search is complete and the first 100 documents are collected.

The first 100 documents are retrieved based on what has been enabled from options such as: keywords, entities, relations, semantic roles, concept, category classification, sentiment analysis, emotion analysis, and element classification (Watson Discovery Service, 2019).

Bringing 100 documents to be re-ranked for NLQ presents a challenge when the user uses a language that is not present in the documents ingested. For example, the documents ingested could be technical documents using official languages and the user could be using a search word that is commonly used among colleagues. This would mean that even when the training example is present for the type of language used by the user pointing to relevant document, the user will not be able to get the expected documents because they will not have been inside the first 100 documents and therefore will not be re-ranked. Therefore, in this thesis, we will be going through various tools and methods that would enable us to improve the return position of relevant documents that a user expects.

Keywords: Machine learning, AI, Search engine, Watson, Watson Discovery, Natural language processing, Natural language query, NLQ, Natural language classifier, Elastic search, return position

Suomenkielinen tiivistelmä: Abstract in Finnish…

Avainsanat: Keywords in Finnish…

Glossary

AI	Artificial Intelligence
CFD	Customer Failure Description
CV	Cross Validation:
FD	Failure Description
FSU	Field Service Unit
HTML	Hypertext Markup Language
IBM	International Business Machines
JSON	JavaScript Object Notation
ML	Machine Learning
NLQ	Natural Language Query
NLU	Natural Language Understanding
OFD	Own Failure Description
Regex	Regular expression
SME	Subject Mater Expert
SO	Service Order
TD	Technical Documentation
THD	Technical Help Desk
TIP	Technical Information Portal
URL	Uniform Resource Locator
WDS	Watson Discovery Service
WKS	Watson Knowledge Studio

List of Figures

Figure 1. Architecture View – Key Components 3
Figure 2. Big Data Analysis Pipeline (Jagadish et al, 2014). 9
Figure 3: Discovery Pipeline (Discovery, 2020b) 17
Figure 4: Architecture of Watson Discovery Discovery(2020a) 18
Figure 5. Query Structure for JSON String 21
Figure 6. Query aggregate 21
Figure 7. How Watson Assistant Works (Watson Assistant, 2020b) 27
Figure 8. implementation steps 27
Figure 9. Building a machine learning model (Knowledge Studio 2020a) 31
Figure 10. Type system (Knowledge Studio 2020b) 33
Figure 11. Adding a dictionary (Knowledge Studio, 2020b) 33
Figure 12. Content Delivery Platform (Content Delivery Platform 2020). 34
Figure 13. annotations and groud truth (Knowledge Studio 2020d) 38
Figure 14. K-fold cross validation 42
Figure 15. Natural Language Classifier (2020a) 44

List of Tables

Table 1. Untrained THD_FSU_Collection 47
Table 2. Untrained CDP_Collection 48
Table 3. Trained THD_FSU_collection 48
Table 4. Trained CDP_collection 49
Table 5. Trained + WKS model on THD_FSU_collection 50
Table 6. Query expansion on CDP_collection 51
Table 7. Watson Assistant on CDP_collection 52

Contents

1 INTRODUCTION 1
1.1 Objective of the Thesis 3
1.2 Background Work for the Thesis 4
1.3 Thesis Structure 5

2 BIG DATA 6
2.1 Introduction to Big Data 6
2.2 Important Technical Challenges of Big Data 8
2.3 Phases of Data Pipeline 9
2.3.1 Data Acquisition 9
2.3.2 Extraction, Cleaning, and Annotation 10
2.3.3 Integration, Aggregation, and Representation 11
2.3.4 Analysis, Interpretation, and Modelling 12
2.4 Challenges in Big Data Analysis 13
2.4.1 Inconsistency and Incompleteness 13
2.4.2 Scale 14
2.4.3 Timelines 14
2.4.4 Privacy and data ownership 14

3 MACHINE LEARNING MODELS, TOOLS AND METHODS USED 16
3.1 Discovery 16
3.1.1 Ingestion 17
3.1.2 Enrichment 19
3.1.3 Query 20
3.1.4 Query Expansion 22
3.1.5 Relevancy Training 24
3.2 IBM Watson Assistant 25
3.2.1 Intent 27
3.2.2 Entity 28
3.3 Knowledge Studio 30
3.3.1 Rule Based Model 31
3.3.2 Creating a Watson Knowledge Studio Model 32

4 PRACTICAL IMPLEMENTATION 34
4.1 Data Source 34
4.1.1 Content Delivery Platform (CDP). 34
4.1.2 THD and FSU 35
4.1.3 Service Order 36
4.2 Training by Subject Mater Experts 37
4.2.1 Training Watson Knowledge Studio 37
4.2.1 Training Watson Assistant 39
4.2.2 Training Watson Discovery Service 41
4.2.3 Enrichment of Training Set with Negative Samples 43

5 RESULTS ... 45
5.1 Untrained ... 46
5.2 Trained ... 48
5.3 Annotated and Trained ... 49
5.4 Query Expanded and Trained ... 50
5.5 Watson Assistant and Trained ... 52

6 DISCUSSION ... 53

7 CONCLUSION ... 55

REFERENCES ... 57

1 Introduction

In 2016 there was a project that was undertaken by Oy IBM Finland Ab (hereinafter "IBM") for a manufacturing company that provides maintenance work for its clients. For privacy reasons, the name of the manufacturing company, its clients, or the company's products will not be mentioned in this thesis. Instead, the thesis will provide similar examples and use-cases where the application and technologies could be applied.

This manufacturing company manufactures a product that is directly used by its client. It also provides services related to maintenance work for the products. To be able to do this, the company has incorporated many technological tools, among which are data models, IoT and AI. In addition to their own products, the company also provides maintenance work for similar products that is manufactured by their competitors. In some cases, this situation can arise because they purchased the product themselves.

This company has a big number of employees that perform this maintenance services and noticed that a lot of time and resources are being wasted regarding maintenance work due to some inefficiencies caused by technicians going to the site without having much idea about what is the current problem and, in some cases, not being able to identify what was the root cause problem. The immediate problem that was registered is quickly fixed instead, and technicians return to fix similar problem soon as the root cause problem has not been mitigated.

The project's mission was to provide the most relevant information to fix certain problem in the most efficient way by providing the technicians with materials such as technical documentation for the part they may need to fix, previous similar history encountered by their colleagues and what they have done to resolve the issue, and technical help desk information. In addition to this, the project also focused on getting the root cause of the problem by analyzing the information coming out of the IoT devices and previous failures.

The project provided a smart search engine to retrieve the information needed. To accomplish these requirements, there were few technologies that were deployed such as IBM

Watson Discovery Service (WDS), IBM Watson Assistant, IBM Knowledge studio and a custom query builder. Furthermore, the project relied on subject matter experts (SMEs) to train the supervised machine learning model by providing positive examples. These examples are given to the model in such a way that, for certain “types” of natural language queries (NLQs), these “types” of documents are relevant.

The first problem description (NLQ) comes to the system when a customer of the company calls the call center. The agent taking the call, writes the description down, creates a ticket, and assigns a technician to the case. This description is referred to as customer failure description (CFD) and will be used as a query input to the system. If the description is too vague and the system would not be able to generate a good answer, or the technician determines the system cannot get much out of it to know what the potential problem is, the technicians will go to the site and examine the problem themselves. If they are not able to know by themselves what the problem is, then they will put another description as a query input (NLQ) to the system. This description is referred to as own failure description (OFD).

Some of the technologies that are shown in Figure 1are out of scope for this thesis, but IBM Watson services will be discussed. Some of the flow will also be introduced, and references will be made.

The theoretical aspect this thesis is, to provide a cognitive method of generating negative examples to enhance the accuracy of a machine learning tool to generate a reliable answer to a natural language query. IBM Watson Discovery Service will be used to test all the results.

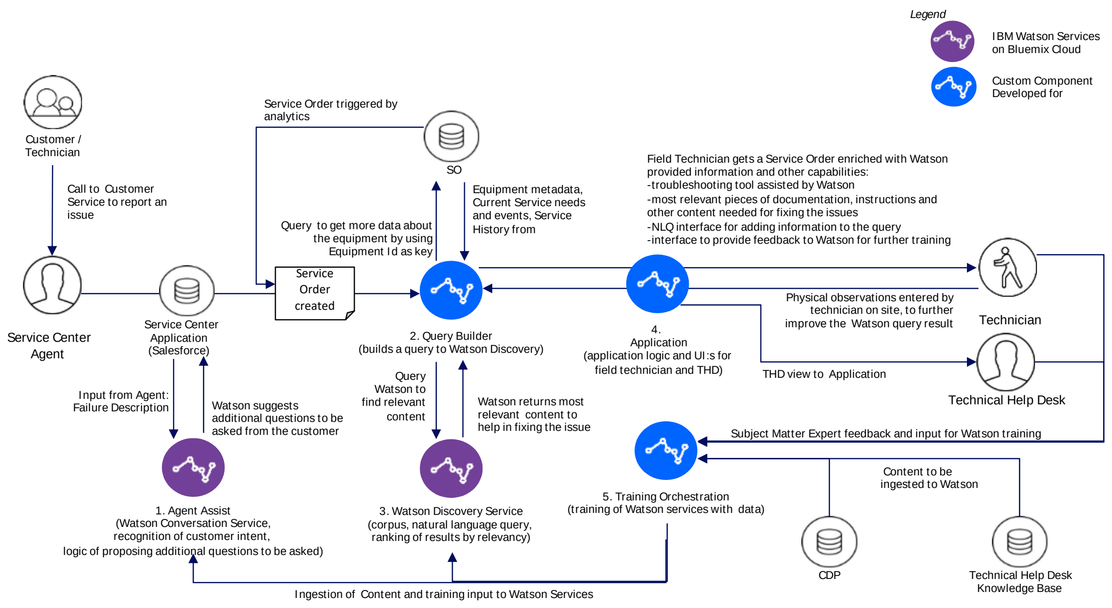
Legend
IBM Watson Services on Bluemix Cloud
Custom Component Developed for
Customer / Technician
Call to Customer Service to report an issue
Service Center Agent
Service Center Application (Salesforce)
Service Order triggered by analytics
SO
Service Order created
Query to get more data about the equipment by using Equipment Id as key
Equipment metadata, Current Service needs and events, Service History from
2. Query Builder (builds a query to Watson Discovery)
Field Technician gets a Service Order enriched with Watson provided information and other capabilities:
-troubleshooting tool assisted by Watson
-most relevant pieces of documentation, instructions and other content needed for fixing the issues
-NLQ interface for adding information to the query
-interface to provide feedback to Watson for further training
Technician
Physical observations entered by technician on site, to further improve the Watson query result
4. Application (application logic and UI:s for field technician and THD)
THD view to Application
Technical Help Desk
Input from Agent: Failure Description
Watson suggests additional questions to be asked from the customer
Query Watson to find relevant content
Watson returns most relevant content to help in fixing the issue
Subject Matter Expert feedback and input for Watson training
1. Agent Assist (Watson Conversation Service, recognition of customer intent, logic of proposing additional questions to be asked)
3. Watson Discovery Service (corpus, natural language query, ranking of results by relevancy)
5. Training Orchestration (training of Watson services with data)
Content to be ingested to Watson
CDP
Technical Help Desk Knowledge Base
Ingestion of Content and training input to Watson Services

Figure 1. Architecture View – Key Components

1.1 Objective of the Thesis

This thesis tries to answer how to get an AI search engine perform better by comparing different use-cases and methods used in training the model. As part of the project, different strategies were performed to improve the result of the cognitive search engine. These strategies will be discussed and the results from each strategy will be shown.

1. How does the model that is not trained perform out of the box?
2. How much can the result improve if a positive training examples were added that was given by SMEs that will change how the documents will be ranked
3. How much can the results change if the documents were annotated using ontology-based technologies such as Watson Knowledge Studio (WKS) due to changing how the documents will be indexed
4. How much can the result change if query expansion can be incorporated that will not change any aspect of the ingested documents but the incoming query.
5. How much change can the result show if the incoming query is altered using Watson Assistant?

Additionally, this thesis will present a theoretical argument about how valuable a negative example is to a supervised machine learning model such as IBM Watson. Moreover, the thesis will give an insight on how to achieve generating these negative examples when they are missing from the training set given by subject matter experts (SMEs), as SMEs tend to focus on giving only positive examples.

1.2 Background Work for the Thesis

The thesis emerged after the implementation of a search engine for an industrial company that provides maintenance of its product to its customers. The maintenance task is carried out by its employees.

When clients of this company report a fault in their devices to the companies' service center agent, the agent creates a ticket. This ticket is then assigned to the technician that is nearby. If the technician has a difficulty fixing the issue, then they call a technical help desk. Technical help desk (THD) has experienced people who are capable of troubleshooting difficult problems, therefore when the technician calls, the technical help desk personnel that is on a standby will help in troubleshooting and fixing the equipment. If the problem happens to be more complicated, then the technician leaves the premises and the technical help desk personnel assisting the technician will go onsite to do the work instead.

This process was costing a lot of money for the company, furthermore, the client satisfaction rate was getting low as a result of delayed maintenance.

To address this issue, a machine learning model was used that had all the relevant documents; equipment manuals, THD documentation (reported by THD personnel after fixing an issue), back reported material reported by the technicians that describes what has been done to fix certain issue, and so on.

After ingesting this material, the model was trained for:

- Industry specific languages for keyword extraction
- Customer complaint made to help desk as a natural language query
    - Training the model by creating positive examples of what document is fit for these example question asked by the customer

    - This has a low level of accuracy in bringing correct answers due to the lack of knowledge from the customer to ask the correct questions.
- Technician complaint made to THD as natural language query.
    - Training the model by creating positive examples of what document is fit for these example question asked by the technician
    - This brings good material as the technician can ask technical questions

1.3 Thesis Structure

The thesis tries to cover the main concept around big data in chapter 2, this should give the reader an idea of what big data is and how institutions around the world can benefit from utilizing the insight that can be unlocked from it.

Chapter 3 will cover topics related to machine learning models. As the work that was done has utilized IBM Watson technologies, the tools and methods will be discussed briefly. This chapter also discusses the methods used to train the machine learning model

Chapter 4 covers the practical implementation of the work and compares the results obtained from untrained model with different methods of the trained model. This chapter also covers the possibilities of training the model from negative example in theory. Finally, a conclusion will be drawn.

Chapter 5 will show and discuss the results obtained by deploying different methods.

Chapter 6 will discuss what kind of work is still to be done in the future of machine learning models, in particular, supervised machine training.

2 Big Data

2.1 Introduction to Big Data

The vast amount of data generated in the past few years has been a driving force for disciplines such as machine learning, cognitive computing, artificial intelligence, etc. Most of these data exists in unstructured format (blogs, social media posts, pdfs, word files, etc.) unlike the structured way of storing date (databases, excel spreadsheet, etc.). Extracting insights from these unstructured data would be invaluable to companies in helping them understand their businesses and their customers better since nearly 80% of their data is structured that way. (Hariri, R. H et al 2019, As Mohammed, E., Mohamed et al 2018, Rai, R. 2017).

Machine Learning technologies offer a solution by analyzing big datasets and differentiating and understanding the numerical relevance in both structured and unstructured data.

They can be tuned to uncover insights by applying text classification process. But these processes are manually intensive, and therefore, tiresome, if not nearly impossible.

Researchers have come up with a method of sampling training example to a few sets and running the entire set based on the examples. This automates the tiresome process; however, the samples are entirely based on positive examples and lucks the negative examples (Xu, Z. et al, 2014). In other words, the information that we have has a relevance score of what is known, "for this type of query, this is the relevant document" but the documents are not labelled by saying, "this document is not relevant for this specific kind of query".

The term "Big data" can be used to describe any dataset that contains the "4V" characteristics of a data, Volume, Velocity, Variety and Value. (Mohamed et al, 2017).

1. Volume: describes the amount/size of data
2. Velocity: describes the motion or lack thereof; data that is at rest (stored in a DB, file system, etc), and data in motion (streaming video, video chat, etc)
3. Variety: describes the format that this data can exist (pdf, text, video, image, audio, etc)

4. Value: describes the importance of data and the insight one can potentially unlock when acquiring it.

By 2025, the world economic forum estimates that 463 exabytes of data will be produced on a daily basis (Desjardins, 2019). To provide a context, one would need nearly 213 million DVDs to store that amount of data every day. The number of data produced in 2018 was in excess of 2.5 quintillion bytes (Hariri, R. H et al 2019), where the acceleration of data generation is staggering to say the least. According to Hariri et al (2019), in the past two years alone, the world has seen the generation of 90% of the total data produced. Furthermore, Pouyanfar et al (2018) predict the number at 40 ZB (zettabyte) in 2020. He argues that "every person in the world will produce almost 5,200 gigabytes of data". This maybe explained as Adler (2016) suggests, many, but specifically the social media "enlightened" younger generation getting access to a cheap and yet very fast internet, combined with a very powerful devices that fits into a pocket, that can create, share and communicate content much faster, has contributed to the generation of vast amount of data in a short time.

In recent years, social media and Internet of things (IoT), has arguably been the biggest contributors to this acceleration. According to data taken from world economic forum (2019), Every day:

- 500 million tweets are sent,
- 294 billion emails are sent,
- petabytes of data are created on Facebook
- 4 terabytes of data are created from each connected car
- 65 billion messages are sent on WhatsApp
- 5 billion searches are made

Norjihan Abdul Ghani, et al (2019) attributes the vast creation of data to the invention of web 2.0. They claim, this invention has taken the process of content generation from publishers to the users. This would mean, anyone who has an internet connection is capable of producing content of different variety (text, video, image, etc), and is fully capable of distributing them to millions of people without needing to have a contract with a publisher. They also attribute the accessibility of the internet, along with the interest and influx of people towards social media, to be the biggest factor in generating a vast amount of data, which, if used wisely can unlock a great potential in many disciplines such as business and research (Norjihan Abdul Ghani, et al, 2019).

2.2 Important Technical Challenges of Big Data

Out of all the data generated from different resources, about 80% of it exists in unstructured format. Unstructured data is the data that is found in documents (word, pdf, txt, etc), media, etc. In other words, any data that cannot adhere to a pre-defined model (database, excel sheet, etc) and cannot therefore conform to a tabular structure. (Rai, R. 2017) Rai further outlines the task to perform any kind of analysis on these types of data can be very difficult.

The challenges that come with unlocking insights from big data have been the vital drivers in research areas such as AI, machine learning, and data mining. The vast amount of data that has been produced in the past few years, especially as the dominance of social media companies have become prominent in generating it, have generated a lot of interest. These data could be the sentiment of user comments of a post, interests from the product searches, product reviews or ratings, or media content uploaded to social media (Pouyanfar et al, 2018). And yet, even though the methods used to extract insights have improved and become more advanced, they have not reached their potential to utilize all of it. Taking the medical industry as an example; the industry accrues high costs not because there is a lack of data, but in utilizing the correct methods to extract the insights from the data. As Mohammed et al, (2018) have pointed out "[the]problem in health care systems is not the lack of data; it is the lack of information that can be utilized to support critical decision making" (p. 2). Furthermore, Mohamed argues, having big dataset has merely any value if not incorporated with the correct analytics that could unlock an insight from both the structured and unstructured data. The industries that have been able to unlock insights from big data, have significantly changed the way they work and have benefited from utilizing and incorporating such technological advancements (Rai, R. et al, 2017; Pouyanfar et al, 2018; Mohammed et al, 2017).

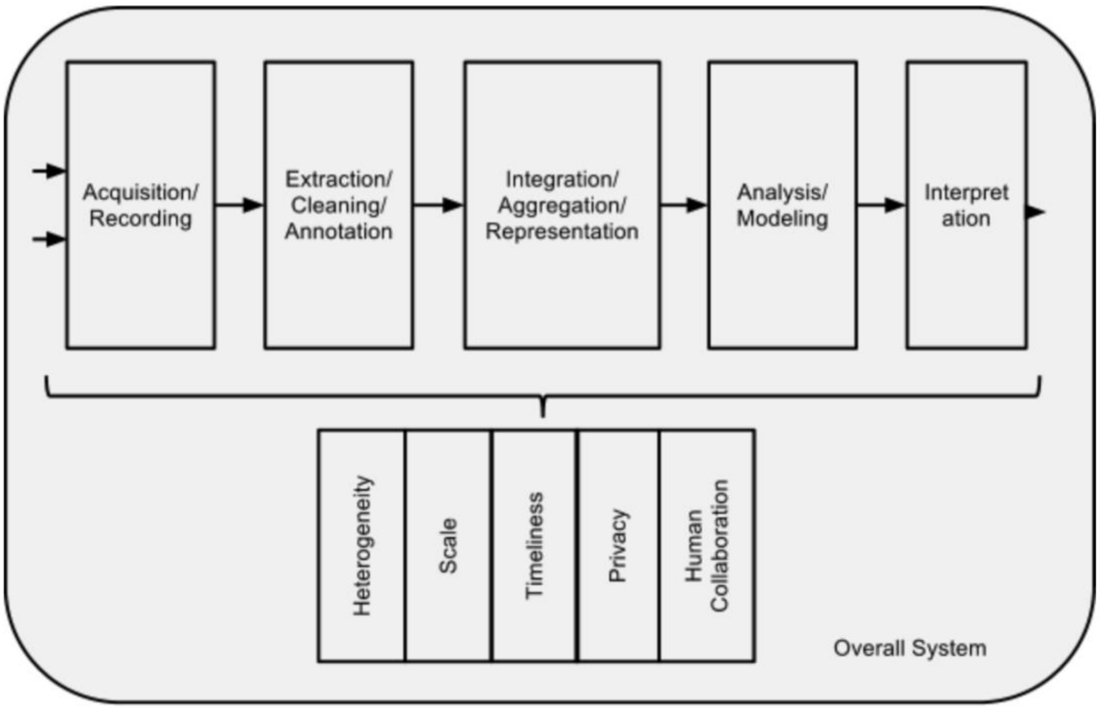

Figure 2. Big Data Analysis Pipeline (Jagadish et al, 2014).

While the top part of Figure 2 represents the major steps presented when analysing big data, the bottom half represents the challenges big data presents to the steps. The analysis in big data is an iterative process that consists of distinct measures taken in each step. (Jagadish et a 2014). Furthermore, they argue that even though data analysis/modelling is a crucial step, disregarding the other steps in the data analysis pipeline could prove to be costly and render the analysis to be of little use.

2.3 Phases of Data Pipeline

2.3.1 Data Acquisition

This stage of the pipeline is more concerned with the collection of data. One challenge raised by Jagadish et al (2014) and Dutta et al (2018), is that even though the collection and availability of large amount of data from different sources is vastly accessible, the challenge of filtering which data is relevant and which data to discard has proven to be difficult. This happens to be true even today as was the case in the project that inspired this thesis. During the data collection phase, even though the company has been collecting different sets of data from different resources and storing them in a data warehouse, the challenge has been to identify which data was relevant for the use case of providing a search engine to the technicians who need it. The data that was collected emerged not only from the back reporting but also from different sensors that were sending data frequently. This IoT data was for the most part not usable as it does not offer any insight to the use case except when there was a trigger that was sent from a failure of the device it was connected to. This was also the case that Jagadish et al (2014) explained as, even though the amount of data that is collected from different sensors of today (traffic signals from cars, weather sensors, medical equipment's, wearables, etc.) is of no use, there's still the need to put the correct filters to determine which data is relevant and detect anomalies when they occur.

2.3.2 Extraction, Cleaning, and Annotation

This part of the stage is more focused on correct representation of data. In order to make sense of the data that was sent from the sensors, the company had deployed an analytics tool that would determine which data points are "interesting" and should trigger an alarm, and in what situation should the alarm be taken seriously. For example:

1. How many times should the alarm repeat to be considered? A one-time alarm could be thought of as anomaly
2. Are there a combination of alarm that are more serious and should be addressed promptly?

This stage is has some parallel with Jagadish et al (2014) who describes the medical industry containing data such as transcribed assessment from the medical professional, medical imaging such as MRI, videos, and structured data that was collected from sensors such as heart rate and blood pressor. In order to properly use this data for analysis, we need a technological method where that extracts these different datasets and puts them in a structured format. (Jagadish et al 2014).

The other challenge with the project regarding the unstructured data was cleaning the data. Not all data was found to be relevant, and as the back reporting made by the technicians relied on the quality of information that they were able to submit, it brought a different challenge; cleansing of data and needing to have a strategy to cast some as not usable. As Jagadish et al (2014) states that understanding the need for data cleansing method could be a starting place because big data in its initial format cannot be trusted, as the data can be collected from faulty devices, false or biased statements from humans and so on.

2.3.3 Integration, Aggregation, and Representation

The project encountered two different data sources that contain similar information for the preposes of assisting a technician troubleshoot and fix a device. These data sources were placed in different geographical locations and maintained by different people within the company. The first one was a Technical Help Desk (THD) information where the THD personnel were creating contents for cases that was considered challenging to fix. The other set, which was Field Service Unit (FSU) was also being collected within the company but in different location serving similar purpose. The database schema that was used for these two data sources was also different. The project Integrated these data sources into one and presenting the content by removing possible duplicates. Similar phenomenon was described by Jagadish et al (2014) where they argue that "A set of data transformation and integration tools helps the data analyst to resolve heterogeneities in data structure and semantics." . They father describe a more "affordable" solutions that are capable of analysing data from aggregated sources recently, among which, the "pay-as-you-go" option has been a preferred one.

They also bring the argument in which, even if we consider a single dataset, different companies have been storing it in different database designs that it's not easily used by others. Bringing about a solution to design a system that can be used even in the absence of intelligent database design.

Data is collected from a heterogenous sources for the analysis to be effective. There has been a situation where the project encountered a data source called Service Order (SO). The data that is in the SO database is initiated when a client of the company calls when having a problem with their device. The service center agent at the call center writes down customer failure description (CFD) and assigns the case to a technician. If the technician fixes this problem, what has been done will be described and the steps taken to solve the issue will be recorded to the database. In other words, the database entry will be updated. But if the technician was not able to fix the problem, they call THD. THD has a different data source that is used to create content. After completing the task, the technician will update the database similarly as the previous case. The THD personnel will in some cases create a content of own observation about the issue, the steps taken to troubleshoot, and the method used to fix the problem. This is two separate information that exists in two different systems, written by two separate people. The next technician that encounters similar issue, could arguably benefit from having a complete view of the information from where it started (CFD) to the advice and information that the THD personnel provided. This is a similar case as the medical information of citizen in Finland today, people find their medical information from different medical providers, public and private. In order to have a complete view of a patient information, one has to aggregate this information in a manner with which one can do analysis. Jagadish et al (2014) note that this brings another set of challenge coupled with a massive dataset in "tracking the provenance" of data.

2.3.4 Analysis, Interpretation, and Modelling

The art of querying and mining big data is vastly different from doing a statistical analysis on smaller dataset. Big data is noisy, untrustworthy, heterogenous and so on. As Jagadish et al (2014) argues, these challenges of big data do not necessarily deem it unusable, on the contrary, insights extracted from a repeating dataset that has a frequently showing pattern, and can be corrected from multiple sources, can be very useful information. This was the case in many ways with the project as the company had installed thousands of IoT devices that each sends few signals every seconds. This presents a challenge in filtering the noise from the useful information and knowing when the alarm should be sent by running analytics on the data obtained. But more so, it was useful for incorporating these functionalities with the AI search engine that the project built. This feature alone become priceless as it was the most reliable information the project used to identify problems and diagnose root causes.

After the data analysis is complete, decision can be made by the responsible entity using the information obtained. When doing that, one should not forget the biases present in multiple places. For example, the consideration of whether or not one alarm sent by one IoT device about the product it is attached to be considered seriously should be put into consideration if technicians are to be sent to fix the product. The other case that bias could be encountered is when technicians themselves asses the recommended document. Some documents have been labelled as relevant by some whereas others have similar case but label the same document as irrelevant. This brings the issue of when to consider using what the technicians have rated as a training material to the machine learning model. Should we use the feedbacks as soon as they come, or does it make more sense to assign someone to validate these examples and have the final say on whether they should be used as a training material.

As Price et al (2014) observes, data collection is prone to having bias. This according to them could lead to some data simply being left out, which statisticians refer to as “selection bias”. Therefore, Jagadish et al (2014) suggests that one should not let the final decision be made by a computer model, but that this advanced statistical analysis should be used to assist the person to make a faster decision, while keeping in mind that the system could have a “bug”, machine learning models having biases that they exhibit from the data generated by a biased “human”, and the data that was collected simply could be erroneous.

2.4 Challenges in Big Data Analysis

2.4.1 Inconsistency and Incompleteness

Big data usually emerges from collecting data from multiple sources such as social media, blogs and news channels. This brings its own challenge as the sources themselves have a varying inconsistency with data. (ArulMurugan, et al, 2014, Jagadish et al, 2014, Dutta et al 2018).

2.4.2 Scale

With the name, big data comes the implication of the volume of data. The volume of data has been a challenge for a long time. As the data grow, the technology around processing speed has also become more advanced. But relaying on processing technology has not been the one bullet solution as the amount of data produced exceeded the speed of processors manufactured (Jagadish et al, 2014).

In the recent years, we have seen a big shift for companies towards cloud infrastructure, and the companies that do provide their service have made it possible to store large amount of data in their data wares for a relatively low cost. This has solved most of the issue with storage and resources for those who can afford it (Awodele et al 2016). They argue that even though cloud computing may have solved the issue of storage space and machines with bigger computational power to process the data, there is still the issue where the "data universe" is doubling in size every 2 years, whereas internet bandwidth has a problem of keeping up. This problem is exasperated especially due to video on-demand providers such as YouTube and Netflix account for a big chunk of the bandwidth

2.4.3 Timelines

As data increases in volume, the challenge to make real-time analysis, summery and a decision on what to save and what to throw away is vital, as it's not economical to save the row data (Jagadish et al, 2014). This row data, that is not summarized and analysed is deemed useless as there cannot be any insight to gain from it. There is also the need to find certain elements that fit certain criteria in a large dataset. Searching the entire dataset isn't a feasible choice. The best way to tackle this problem is to create an index pattern in advance that has these matching elements (Jagadish et al, 2014).

2.4.4 Privacy and data ownership

Privacy issues are one of the most challenging topics of our time, and big data makes the challenge even more difficult. This is mostly due to the possibly that a person with malicious intent can link data from different sources to acquire a private data about people (Jagadish et al, 2014). Furthermore, he points that collecting information from different social media applications is one good example of this, as one could fill a lot of gap on user's information by aggregating the data from multiple sources. In some cases, this could be taken even further by associating a person from images that were taken and posted by other "friends" on social media. One example of this could be, an image tagged of a person with a politician in a fund raiser, which could appear as if the person is a supporter of the politician for their presence.

Consider an application that requires the access for people's locations (ex. weather forecast), If those who may have a malicious intent can hack into the application providers server can access your location, and by inference can gain a very good idea of the people using the application. This is possible because the movement pattern of a person can reveal a lot about the identity of self (González, M.C et al, 2008, Jagadish et al, 2014). There is a big probability these applications can identify a user by only tracking the locations frequented by this user. Also, by collecting location information the user can reveal their religious and political affiliation (Jagadish et al, 2014) by gathering location from those locations that are frequented by the user. Another example could be in a situation where a user's frequent visit to a cancer clinic for example can reveal the user's medical condition.

3 Machine Learning Models, Tools and Methods Used

As the architecture diagram of the project shows in Figure 1, there are IBM Watson technologies as well us custom technologies incorporated in the project. This thesis will only be discussing about the IBM Watson technologies that was used for the project and will not discuss the custom components even though they may be mentioned.

Within the IBM Watson technologies, only the features that were used with the project will be mentioned. This thesis will not cover IBM technologies that were not relevant for the project, nor will the features that were not relevant be covered here.

As the diagram shows the flow of information goes in such a way that as the service center agent is assisted by Watson Assistant to get more information out of the calling customer. This will be briefly discussed as we show how intent and entities are built. The customer failure description (CFD) is then sent to a different collection of Watson Assistant that is deployed in front of Watson Discovery. This instance of Watson Assistant will enrich the incoming query. This will also be discussed in detail as the thesis describes building entities. Then Watson Discovery receives this incoming query, and it will retrieve the first 100 documents that made the hit with the search. Then it will rank these documents using the training material provided. The training done as well as Discovery will be discussed in the thesis.

The thesis will also discus Watson Knowledge Studio (WKS), which was used in the project to annotate documents by the team that worked on the project.

3.1 Discovery

Discovery is a cloud-based AI search engine that can ingest a structured and unstructured data that is either proprietary, public or third party. It can be trained with positive examples using natural language, can enrich the ingested data with a domain language to understand specific context within specific domains, and can be quired to get and unlock relevant data using natural language query. (Discovery, 2020a).

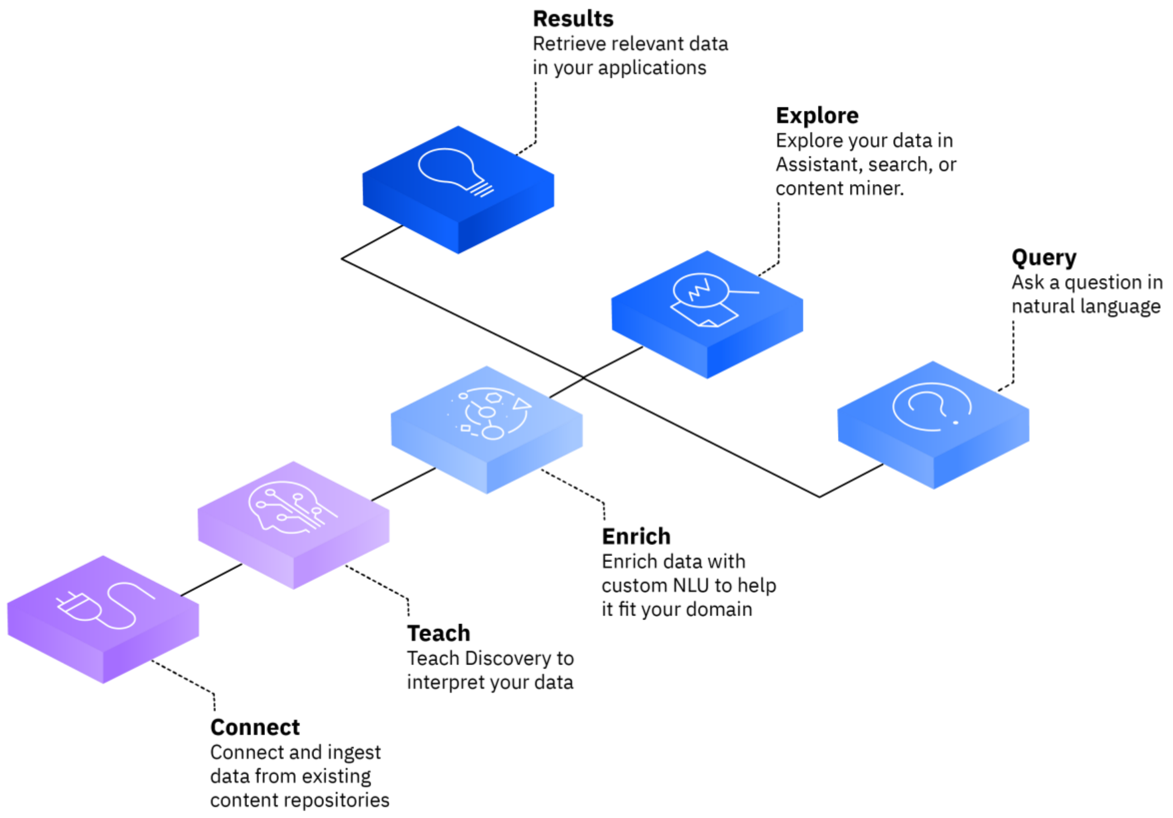
Results
Retrieve relevant data in your applications
Explore
Explore your data in Assistant, search, or content miner.
Query
Ask a question in natural language
Enrich
Enrich data with custom NLU to help it fit your domain
Teach
Teach Discovery to interpret your data
Connect
Connect and ingest data from existing content repositories

Figure 3: Discovery Pipeline (Discovery, 2020b)

3.1.1 Ingestion

Ingesting a document refers to the process of uploading documents to Discovery. This process can happen with different methods.

The project used Discovery as a search engine tool. This tool ingested both unstructured and structured data. The unstructured data were ingested in such a way that the readable contents were separated from the metadata of the content and saved with the same name. The metadata always had a JSON format and was as a JSON file. The content section of the data was ingested in different format. For example, JSON, PDF and HTML. As Discovery is known to distort the original content, when tables and images are present, additional metadata was created to indicate where the original content was saved. This original document is then severed to the technicians rather than the distorted content from Discovery.

The structured content was extracted and ingested into Discovery in such a way that, each row was considered as one JSON file. The column that contained information regarding the problem description with the device, and the description of the column that contains the solution to this problem were placed in the parent node of the JSON string as value. The key of the JSON being the name of the column itself.

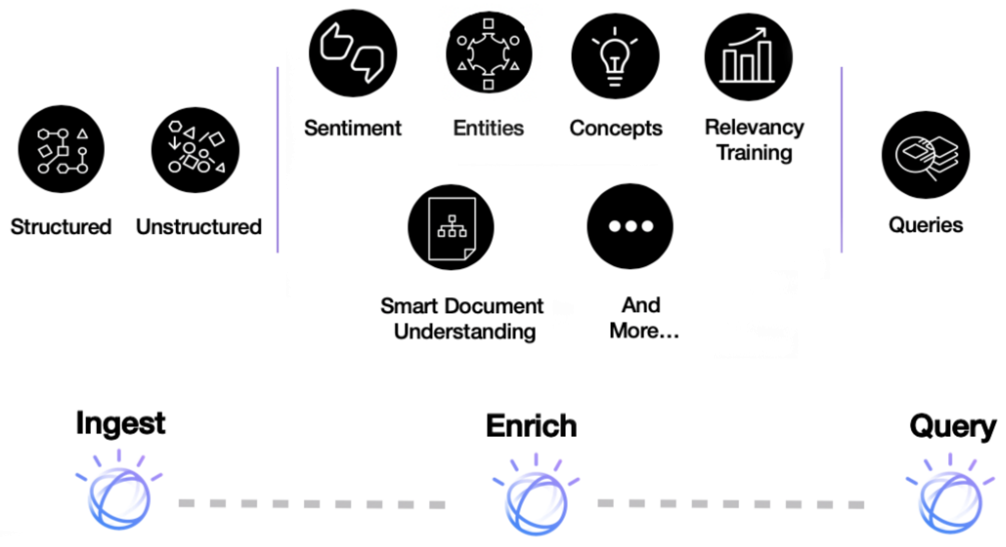

Figure 4: Architecture of Watson Discovery Discovery(2020a)

In general ingestion can be done using

- API
- Tooling: content ingestion mechanism that enables users to upload their content directly into Discovery from on-premises file locations
- Box crawler: Crawls data from cloud content managers such as:
    - Salesforce,
    - Microsoft SharePoint Online
    - IBM Cloud Object Storage
    - Microsoft SharePoint 2016 data sources

- Web Crawler: To crawl websites that are public and do not require password

3.1.2 Enrichment

Enrichment is the process of adding a cognitive metadata to the text field of documents “with semantic information collected by these four [updated to 8] Watson functions” (Discovery 2020)

Keyword Extraction: Extract important keywords from the specified field of the ingested document. After detecting these important keywords, Discovery ranks them for return positions (RP)

- Sentiment Analysis: Identifies the overall sentiment of a field within the ingested document as positive or negative
- Concept Tagging: Identifies general concept of a field within the ingested document. These concepts may not be directly mentioned in the field
- Category Classification: Classifies a specified field into its category. This could go five level deep
- Semantic Role Extraction: Deconstructs sentences in the ingested document to semantic triplets (subject, object, and action)
- Emotion Analysis: Analysis the emotional sentiment of a field within the ingested document. The most notable emotions being anger, disgust, fear, joy, and sadness
- Entity Extraction: Extracts entities from a field within the ingested documents. These entities could be person, companies, country, city, etc. One has the option of adding a desired entity by using a Watson Knowledge Studio (WKS) model.
- Relation Extraction: Discovery can recognize when two entities have a relationship and identifies the type of relationship. It’s also possible to add a WKS model that has a custom relationship model.

In addition to this enrichment options Discovery also provides

- Dictionary enrichment: allows the enrichment of a fields in the ingested documents. This can be enrichment added as synonym (computer, PC, workstation) or can also be done by adding the same category (computer, electronics, machine). These features are also provided by other Watson technologies such as Watson Assistant, WKS and Natural Language Understanding

- Identity and extract values (Regular expressions). This is also provided by services such as NLU and WKS
- Advanced rule model

3.1.3 Query

Discovery offers a powerful mechanism to extract information and insights from documents that are ingested and enriched. This is a content search capability that can be integrated to applications.

Some of the query behaviours that Discovery exhibits by default are:

- Stemming: The result of a query always matches the root word in the document, but it does not necessarily apply for any arbitrary word. For example, “computation” and “computing” will be reduced to “compute”. But "boatbuilding," "boathouse," and "boatload," are not reduced to "boat."
- Tokenization: For certain languages tokenization occurs by using certain standard separator such as space, hyphen, underscore, and punctuation marks for English
- Special character identification for some languages (English, German, French, Dutch, Italian, and Portuguese)
- (Discovery 2020e)

In addition to the default options, WDS also provides some optional features

- Passage extraction
- Queries can be configured to retrieve passages within a document instead of bringing the matching document as a whole. The default return is setting is 10 paragraphs and 400 characters long
- Complex queries and multiple filters and complex aggregations
- Although this feature is available only on the API, it increases the query length limit to 10,000 characters

When structuring a query in Discovery, it must follow the same hierarchy as the underlaying JSON. More information on query parameters can be found from Discovery, (2020e).

Example:

```
"enriched_text": {
    "concepts": [
        {
            "text": "Cloud computing",
            "relevance": 0.610029
        }
    ]
}
```

Then Query should be structured in a way that corresponds to the JSON string:

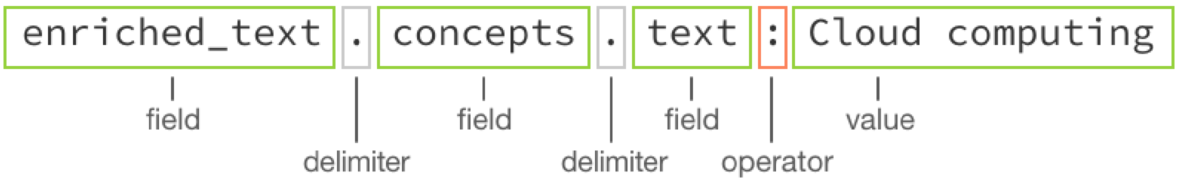

Figure 5. Query Structure for JSON String

It is also possible to aggregate a query in order to return a set of data value

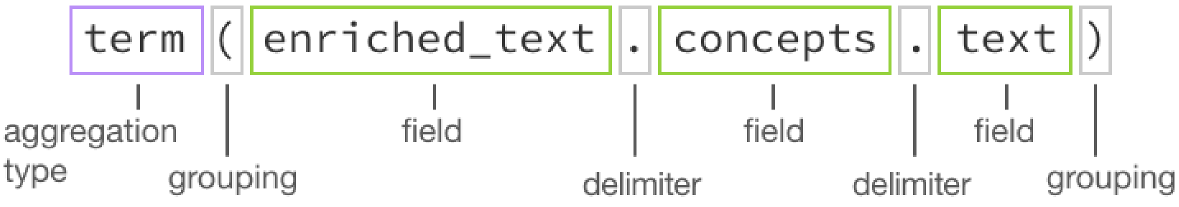

Figure 6. Query aggregate

In the above image, the query aggregates and find all the “concepts” in the collection, where the operator used is a . and the operator is a (),

In addition to aggregates, it is also possible to add filters. The reason to use filters is to limit the documents from which the search should be done against.

Example:

```
filter(enriched_text.entities.text:"printer").term(enriched_text.sentiment.document.la
bel,count:3)
```

This example aggregates results from a collection of documents that contain information on “printers” and returns “positive”, “negative”, or “neutral” sentiments (Discovery, 2020f)

3.1.4 Query Expansion

In some cases, the words used in the natural language query is not present in the documents that are ingested, thereby creating a situation where relevant documents are not returned in the first 100 results. Discovery has a rule that only the first 100 documents are returned for ranking, therefore, the query will not bring relevant documents. To solve this issue, one can expand the query beyond the original word. For example, one can add “PC” and “workstation” as an expanded word for a “computer”. This way when a user searches for any of those words, a result containing those words in the documents will be returned and therefore, ranked to be seen. As query expansions are applied on a collection level, querying multiple collections is possible. In this case each collection will apply their own individual query expansion to the query without affecting the others.

Another advantage of query expansion is that they can be added at any time. This is possible due to the fact that they are applied during query time and therefore, do not affect the indexes applied to the collection. (Watson Assistant, 2020a).

There are two types of query expansions:

1. Bidirectional

   Bidirectional query expansion works in a way that the original word is expanded to the expanded words and vice versa
2. Uni directional

   Uni directional query on other hand only works in one direction, meaning, the original word is the only word that is expanded

The project had

- 32 unidirectional expanded terms and

- 17 bidirectional expanded terms.

These queries expansion was collected over time by evaluating the rating of Discovery with SMEs. This was built after releasing the product for use by 30 technician who were the first adapters. The process of collecting the query expansion terms was done in such a manner that the project team will evaluate the rating by these technicians, if there was no document that was rated as relevant by these technicians, the team will try to see if the query string is not matching any terms. If that would be the case, the term that was used in the query will be compared to the term that was used in the documents. If there is a mismatch between these words, a query expansion will be created.

Bidirectional Examples:

```
{
    "expansions": [
        {
            "expanded_terms": [
                "computer",
                "workstation",
                "PC"
            ]
        }
    ]
}
```

Unidirectional Example

```
{
    "expansions": [
        {
            "input_terms": [
                "PC"
            ],
            "expanded_terms": [
                "computer",
                "workstation"
            ]
        }
    ]
}
```

Query expansion has few restrictions:

1. Only available for private collections and does not work for example for the news collections
2. The amount of “input_terms” are limited based on the account type
3. The amount of “expanded_terms” are limited to the account type
4. Compound words can be unpredictable in the way they return result as they may “OR” the expansion
5. Only one query expansion can be present in a collection.
6. Disabling the query expansion is only possible by deleting it
7. Can only be uploaded as JSON

3.1.5 Relevancy Training

When documents are Ingested in WDS, each document will be assigned a unique id called “document_id” unless one is assigned. This document_id is used for the training material as shown in the code snippet below. This training material contains natural language query (NLQ), filter (optional), and one or more example documents and their relevancy score, which in this case is measured from 0-10. This number could range from any number starting from 0 and ending at 100. (Discovery, 2020g).

In the project the relevancy training was performed using a star from 1 to 4.

1. One star means the document is not relevant for the query
2. 2-stars means the document covers the same topic as the query but does not talk about the issue
3. 3-stars means the document talks about the same topic and describes the issue, but it is not spot on
4. 4-stars means the document is relevant for the query

This star system is then scaled to 0-10 where 0 = 1-star, 3 = 2-stars, 7= 3-stars, and 10 = 4-stars and ingested to discovery as shown in the snippet below.

For situation where a document is rated with relevant or not relevant, the training example instructs WDS that for this “type” of NLQs, this “types” of documents are relevant. In other words, this example is not a mapping exercise where WDS looks at all the NLQs that are exactly the same as this and gives the documents to the user with the highest rating of 10. (Discovery, 2020g)

```
{
    "examples": [
        {
            "document_id": "b401cb55-53fa",
            "relevance": 0
        },
        {
            "document_id": "d8de6687-bfc2",
            "relevance": 3
        },
        {
            "document_id": "131e5e8e-ccbd",
            "relevance": 7
        },
        {
            "document_id": "246b95a5-5cf1",
            "relevance": 10
        }
    ],
    "natural_language_query": "paper stuck inside printer A",
    "filters": "metadata.printer_model: Canon Maxify MB2150 A4"
}
```

For the training material to start working, there must be a minimum of 50 training examples ingested to WDS. The more documents there are in WDS, the more training is needed to achieve a good level of outcome (Discovery, 2020g)

3.2 IBM Watson Assistant

Watson Assistant is a conversational UI that provides an interactive experience with people. The tool allows users to configure how it responds based on their use cases. The tool can also make it possible to search a knowledge base to provide an answer rather than responding with response that is unlikely to be what clients will be looking for. One way it does this is by rating the confidence level of its understanding of the query/question asked. If the confidence is below the desired threshold, then it can be asked to look into a search engine, such as Discovery, instead of replying to the client with a low confidence. Configurations like this can mitigate the frustration of a customer when getting inaccurate answers, increase in satisfaction, and saves time from both the client and service provider. (Watson Assistant, 2020a)

One of the big benefits of Watson Assistant is that it can alleviate wasting of time by responding to the routine and most common questions. The agent working for example in customer service, technical helpdesk, HR, or medical care can instead focus on the complex use cases rather than getting tied up with routine cases where their skills may not be adequately utilised. Also, as a customer one may not want to wait on a call to a customer service number when the information needed is trivial (Watson Assistant, 2020a)

As can be seen in Figure 7, it is possible to communicate with the assistant though either a virtual assistant, or a web application that has a chat embedded, or a custom application such as a mobile application. The Assistant then routes the user input to its dialog skill. The dialog then analysis the user inputs and redirects to any conversation path unless the conversation cannot be answered with enough confidence. In that case, the Assistance may direct the user input to a knowledge corpus that the company configures to use in such cases. (Watson Assistant, 2020a)

Depending on the subscription plan one can utilize the utilities that are found in Watson Assistant as can be seen in Figure 8. Intents, entities and dialog can be found on all the subscription with the option to add search skills by embedding existing help content.
Once the skill has been created, it is not possible to change the language. Instead, the skill can be downloaded and modified as a JSON string, where the "language" property can be modified to the desired language and uploaded back to Watson Assistant. For bidirectional language such as Arabic, skill preferences can be changed. (Watson Assistant, 2020c)

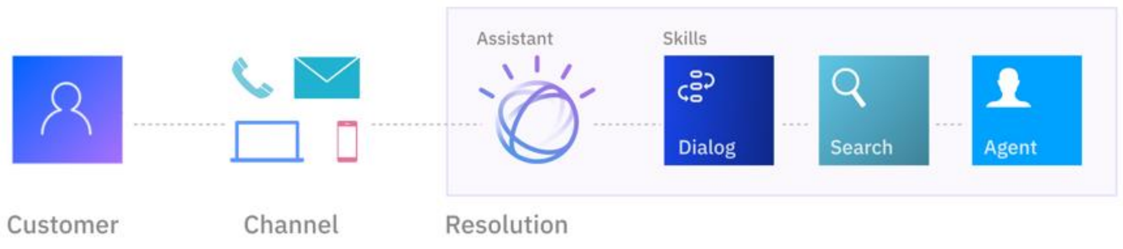

Figure 7. How Watson Assistant Works (Watson Assistant, 2020b)

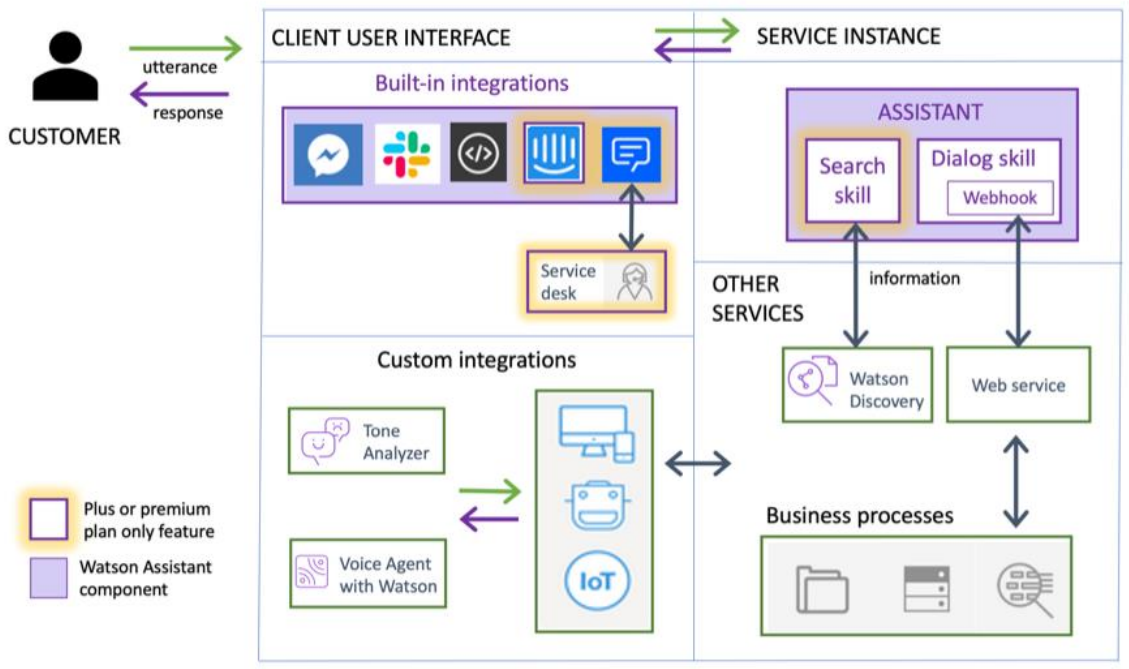

Figure 8. implementation steps

3.2.1 Intent

Intents are a list of categories that indicate the purpose that are "expressed in customers input". Watson Assistant (2020d). To be able to get a working conversational chatbot, intents should be defined. Using content catalogue is the easiest way to start with this as it contains the most common intents that can be edited as needed.

There are two instances in the project where intents were created. The fist for assisting the service center agent, and the second for augmenting the incoming query. Typically for collecting these types of data, SMEs are asked to come up with examples representing each category. In the case of this project though, the historical data from the database was used to collect these examples and SMEs were used to verify if the examples collected are valid for the category.

One of the ways that the tool was used to assist the service center agent was by trying to identify from the calling customer the possible failing component, and if possible, to get as deep level components as possible. The devices have parent components, each of these parent components have subcomponents, and each of these subcomponents have child components. The components are represented with a unique number. For example, if we were to consider a printer, the “peripheral” could be a parent component which can have an identification number of 4000, within this main component, there could be subcomponents, such as “usb flash drive” with id 4100 and “LAN connector” with id 4200. These components would be used as an intent category. Using the service order history, where technicians back report the component they fixed, this information is extracted from the database and verified by SMEs.

Overall, the total number of intents build on the basis of components were 272.

3.2.2 Entity

One way to look at entity is to think of them as the “noun” as the intent is the verb. For example, when the intent is to get the color of the fruit, the entities can be “blue”, “red” and so on. Recognizing the entity in the users’ input helps to refine the response given back to the user. For example, the intent can be #watch_movie, but the entity @movie_geners, can be created to answer exactly what genera of a movie the user is looking for within the intent. The response can therefore be curtailed to answer specific genera questions, rather than giving the user a general answer about movies. Additionally, entities can be referenced with intent creation by either mentioning the entity value/synonym or by annotating mentions. There is also a possibility to directly reference entity name in an intent example

As was the case in the creation of intent, there were two cases where entities were used.

1. The service center agent

The service center agent uses Watson Assistant's entity to identify exactly which component within the identified intent was mentioned.

2. Infront of Discovery to be used as dictionary

It is recommended to create a small amount of intent as a start and testing those before creating more

There are multiple evaluation methods used to detect entities in the user inputs:

1. Dictionary based method
    a. Synonym entity

       Entities will be created as a category of terms. These terms will have one or more values and these values will have one or more synonym. When a user inputs a query, Watson Assistant will check if there is an exact match within the synonym.

       In the project this method was used to either replace an incoming query string that was known not to be found in the technical document or to augment the incoming query to give it more relevant meaning. For example: in the case of the printer company, if the paper was said to be trapped and the word "entrapment" was used such as "paper entrapment in printer xxx", the words "paper", "in" and "printer" will not bring much value as they are most likely to be found in all the manuals. And the word "entrapment" is highly unlikely to be found in any document as the documents will be describing the situation as "stuck" or "jammed". Therefore, a dictionary can be created for "entrapment" with synonym of "jammed" and "stuck". When the incoming query hits this entity, the query will be changed from "paper entrapment in printer xxx" to "paper stuck in printer xxx" or "paper jammed in printer xxx". It could also be replaced if found to be relevant to emphasise on the entities "jam" and "suck" by making it "paper trapped jammed in printer xxx".

    b. Pattern entity

       Entities will be created as a category of terms. These terms will have one or more values and these values will each have a regular expression that defines the “textual pattern”. When a user inputs a query, Watson Assistant will check if there is a pattern that matches the words from the input. In the project the pattern entity was for example utilised to identify components as the technicians would type the codes as input query. These codes did not exist in any of the documents, therefore, identifying them and replacing them with their official name resolved some issues. For example, replacing a component code 4100 with “usb flash drive” brought some relevant documents.

c. System entity

These are prebuilt entities that can be used to cover common categories such as numbers, dates, months and times.

In the project @contactinfo system entity was used to filter if contact information was sent as input query to strip out that information.

2. Annotation based method

Also referred to as a contextual entity, is a method that can be used to build the entity using annotation.

a. Context entity

Entities will be created as a category of terms. Then the user examples that are inside of intent that contain the same term is annotated and labelled as the entity

3.3 Knowledge Studio

IBM Watson knowledge studio (WKS) is a tool that is used to be able to understand the linguistic nuances, meaning, and context of a text in a document. This machine learning tool can create a rule-based model that is capable of finding, for example, entities, based on rules defined within a specific industry or use-case. WKS can also use these custom entities and build a custom relationship between them. When trained, WKS, can pick up entities that are linked in by these custom relationships. (Knowledge Studio 2020a)

For example: *Paper got stuck when trying to print on the new Canon Maxify*

A custom entity can be built to recognise “Canon Maxify” as a *printer_model* entity and the word “stuck” as a *issue* entity. The two entities can be related with *hasSymptom* relation. (Knowledge Studio 2020a)

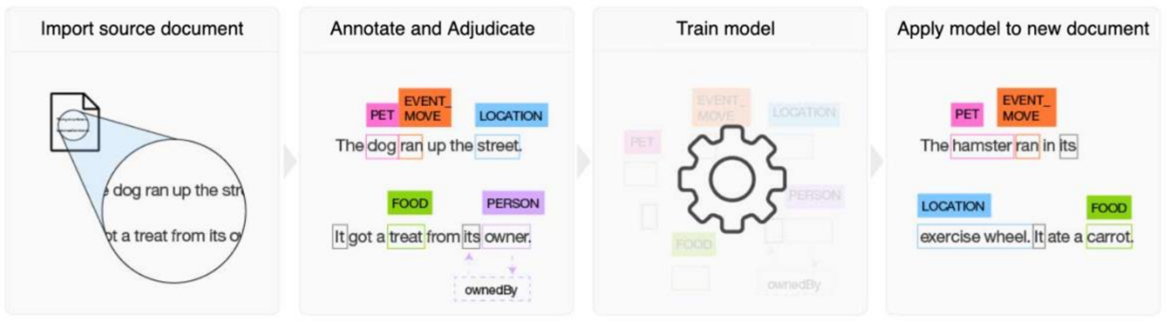

Figure 9. Building a machine learning model (Knowledge Studio 2020a)

As can be seen from the diagram, the project incorporated WKS by following these steps:

1. A domain specific document is imported, and a team creates a type system to identify entity types and relationship types. As the SMEs were not participating on this exercise, the development team developed this task for only one data source.
2. Different annotators annotated a small portion of the documents for a set of entity types. The group of annotators also annotate relationship types when there are relations found between the annotated entities. Annotation anomalies must be resolved before a “ground truth” can be established from one set of annotated documents.
3. Using the ground truth from the previous stage WKS is trained.
4. The trained model is used to find entities, relationships, and coreferences in a setting where the model has never seen the new documents.

(Knowledge Studio 2020a)

3.3.1 Rule Based Model

WKS provides a rule-based model editor to simplify finding patterns of rules in the documents. When constructing rules, similar to entity, classes can be used. These classes can be either complex classes or intermediate classes. Intermediate classes by themselves have no use but can be useful when used with other intermediate classes. (Knowledge Studio 2020b)

The rule editor provides a helpful in defining rules

1. Dictionary
   After adding a dictionary, assign a class name. Any word found that are defined in this class from the document is automatically annotated with a class of this type.

2. Regular expression
   Regular expression (regex) can be used to match the syntax of the word by matching sequence of characters. After a class is created, a regex expression is added. Any word that matches these regex's is automatically annotated in the document.

3.3.2 Creating a Watson Knowledge Studio Model

1. Assigning user Role
   The document is annotated people such as subject matter experts (SMEs), project managers and cognitive engineers. Administrators assign each user a role in which they will determine the authority of the role of the user. There must be at least one "Admin" role defined. The user assigned this role can create a workspace and work as project manager or human annotator. The user assigned the admin role cannot be downgraded as any user with any role can only be upgraded. In addition, the more human annotators there are, the better it is to make sure the work is done accurately by ensuring that there is a section of the documents that these annotators will have in common. There should be at least two human annotators in a workspace. (Knowledge Studio 2020b)

2. Creating a workspace
   A workspace is needed to define all the work that is needed with the model such as, documents, dictionaries, annotation done by human annotators, and type system.

3. Type System

Type system refers to anything within the domain system that are perceived as interesting and labeled with annotation.

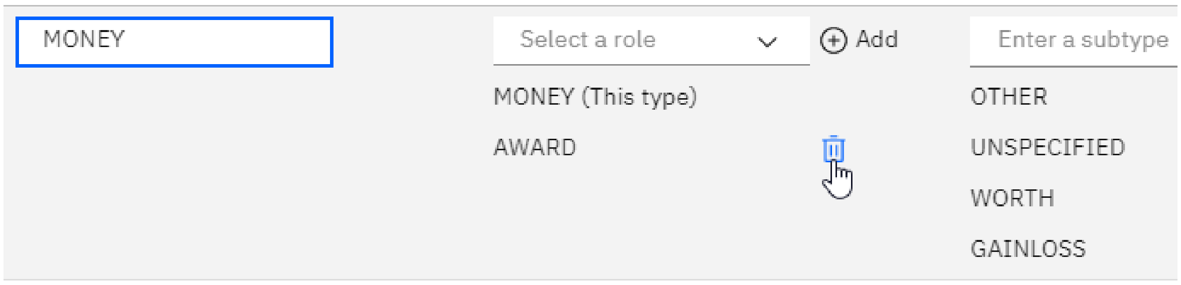

Figure 10. Type system (Knowledge Studio 2020b)

4. Adding a dictionary

   Even though this is optional step to create a model, adding a dictionary can automate the process of annotation. By importing a dictionary or building one using the tool, the words in the documents that match to the words in the dictionary are automatically annotated. This method simplifies the annotator's job (Knowledge Studio 2020c)

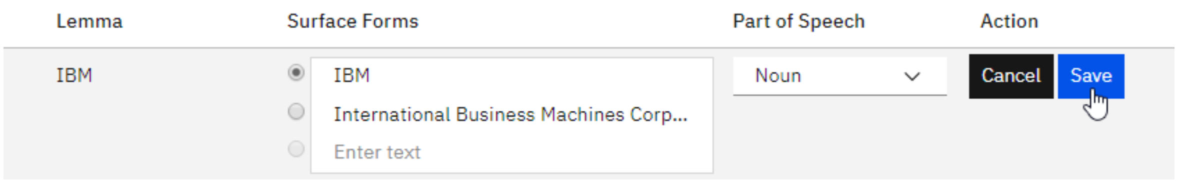

Figure 11. Adding a dictionary (Knowledge Studio, 2020b)

4 Practical Implementation

4.1 Data Source

The majority of time was spent in finding and engineering relevant information that is useful for the technician. At the end 4 useful contents were found,

1. Technical documentation (manual) in content delivery platform
2. Technical help desk information (THD) in a database
3. Field Support Unit (FSU) in a database
4. Back reporting (service order) in a database.

4.1.1 Content Delivery Platform (CDP).

The technical documentation (manual) of the product was found to be useful in enabling the technicians with the proper information to fix a problem. These documents are placed in a content delivery platform (CDP).

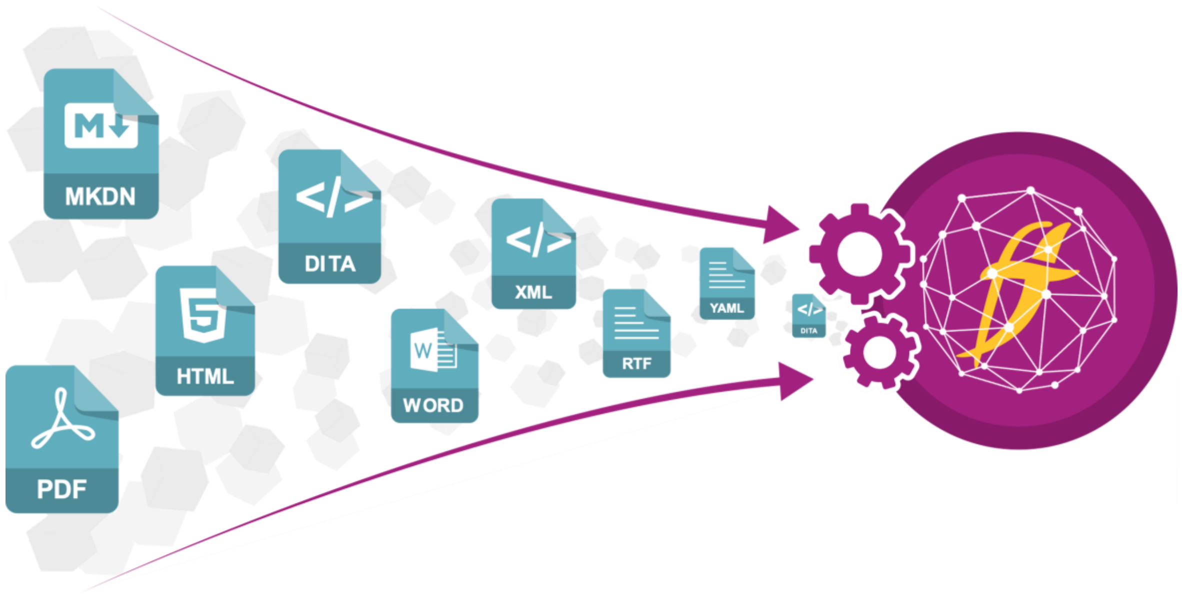

Figure 12. Content Delivery Platform (Content Delivery Platform 2020).

The topic of CDP is out of scope for this thesis, but as a technology that was used for the project, it will be briefly explained.

Content manager has an API integration where it is possible to extract content in a JSON format. The extracted content has a “topic ID”, “content URL”, “title” and “metadata”. This format made it possible to ingest the document without the need to reformat the content. The content URL returned a HTML content which was ingested to WDS as the content. The rest of the JSON content were ingested to WDS as the metadata of the content with the same filename as the HTML file. This metadata is used in order to filter out the content that applicable to the query. For example, if we assume that we are applying the same method for a printer company and a user was having a problem with printer model A, this user has no need for getting a manual for model B. Therefore, filtering the search criteria to only include model A will avoid all the other models from being returned. Additionally, the content had the same metadata incorporated as the data model that was working with the IoT signal. This was key to deliver content that was relevant to the customer’s or the technician’s query filtered by the fault registered by the IoT in order to signal the root cause of the problem.

This data source was crawled, and the content was ingested into a separate collection in Watson Discovery. For the sake of simplicity, this collection will be referred here to as CDP_collection. This collection had a total number of 5037 documents.

4.1.2 THD and FSU

At the initial stages of the project, it was decided that Technical Help Desk (THD) and Field Service Unit (FSU) will be separate source of information. This was done so because these two data sources were found in different system and served different purposes.

THD was written and updated by skilled and experienced employees of the company who have worked for many years as technicians. The way THD data was written was in such a way that when technicians encounter difficult cases and are unable to fix the problem themselves, they call these experienced colleagues. These experienced colleagues will try to assist the technicians in troubleshooting the problem and provide ways to fix the issue. If a lot of time is wasted in troubleshooting the problem, then the technician will leave the premises and the THD personnel will go instead. After the THD personnel has solved the issue, they will document what the problem was (OFD), what equipment it was (metadata for WDS), and what solution was worked to solve the problem. This is saved in a database.

FSU was discovered in latter stages of the project and it became apparent that it originated in a similar way except for the fact that it was even more advanced employees with longer experience that were responsible in creating the content, and it existed in a differ country. The description of the technical solution and the problem encountered were very similar, but the "metadata" section was slightly different. Therefore, some data engineering was required to merge these two data sources. Some of the method used to merge this data were:

1. Columns that had the same kind of content, but different names were given the same name
2. Unimportant columns were removed
3. All the information that was ingested from these different sources had a metadata field indicating where the data was taken from.

These data sources were ingested into WDS as a separate collection. For the sake of simplicity, the collection will be referred here to as *THD_FSU_collection.* After cleansing the data, this collection had about 7800 documents.

4.1.3 Service Order

As seen on Figure 1, service orders are created when a client of the company calls the call center to report on a failure of the equipment. The call center will record the description of the problem and assign it to a technician to solve the issue. The technician who receives this ticket will be able to see the customer's failure description (CFD). The device is then inspected by the technician who can troubleshoot and fix the issue. After resolving the problem, the technician will report what they have done and the tool they have used to fix the problem. This description of the solution along with the CFD will be saved to the database.

This data source was used as a historical lesson from colleagues to show the actions taken when similar problems were encountered with a similar type of device. Even though, this collection has the potential to provide a valuable information to colleagues, the quality of the data recording was not always relevant for different reasons ranging from negligent behaviour when reporting to luck of time between tasks. Therefore, there was a need to clean the data before ingestion. As this collection had more than 100,000 rows, it's nearly impossible to do data cleansing manually, therefore, some techniques were used to clean the data:

1. If the back reported solution contained less than 5 words and less than 15 characters, the row was removed completely
2. If the back reported solution contained customer address or information that had no relevance to fixing the equipment, it was removed using regex.
3. Addresses in the place of CFD were removed using regex.
4. If a row that did not contain relevant information was found by the technicians, it will be back reported by the application to be removed from the collection.

These data sources were ingested into WDS as a separate collection. For the sake of simplicity, the collection will be referred here to as *SO_collection.* This collection had 50700 documents in total.

4.2 Training by Subject Mater Experts

Training a supervised machine learning model relies heavily on the subject matter experts (SMEs). For this project there were three different tools that needed the SME to dedicate some time to do training, WDS, WKS, and Watson Assistant.

4.2.1 Training Watson Knowledge Studio

As describe in Chapter 3, Knowledge Studio, is used to annotate the documents that are ingested into a cognitive search engine such as WDS and Watson Web Explorer (WEX). Therefore, the tool is used on a document level and not necessarily on the incoming query level. But If a dictionary is built as discussed in chapter 3 using WKS in such a way that what is a commonly used word (for query) is the synonym for the technical word (language found in technical documents), then a query would return and rank a relevant document event though it is not mentioned in the document with that type of language.

A sample of 300 documents were used for annotation and two annotators were assigned for this task. These 300 documents were splits into train set (70% of the documents), test set (23% of the documents), and blind set (7% of the documents).

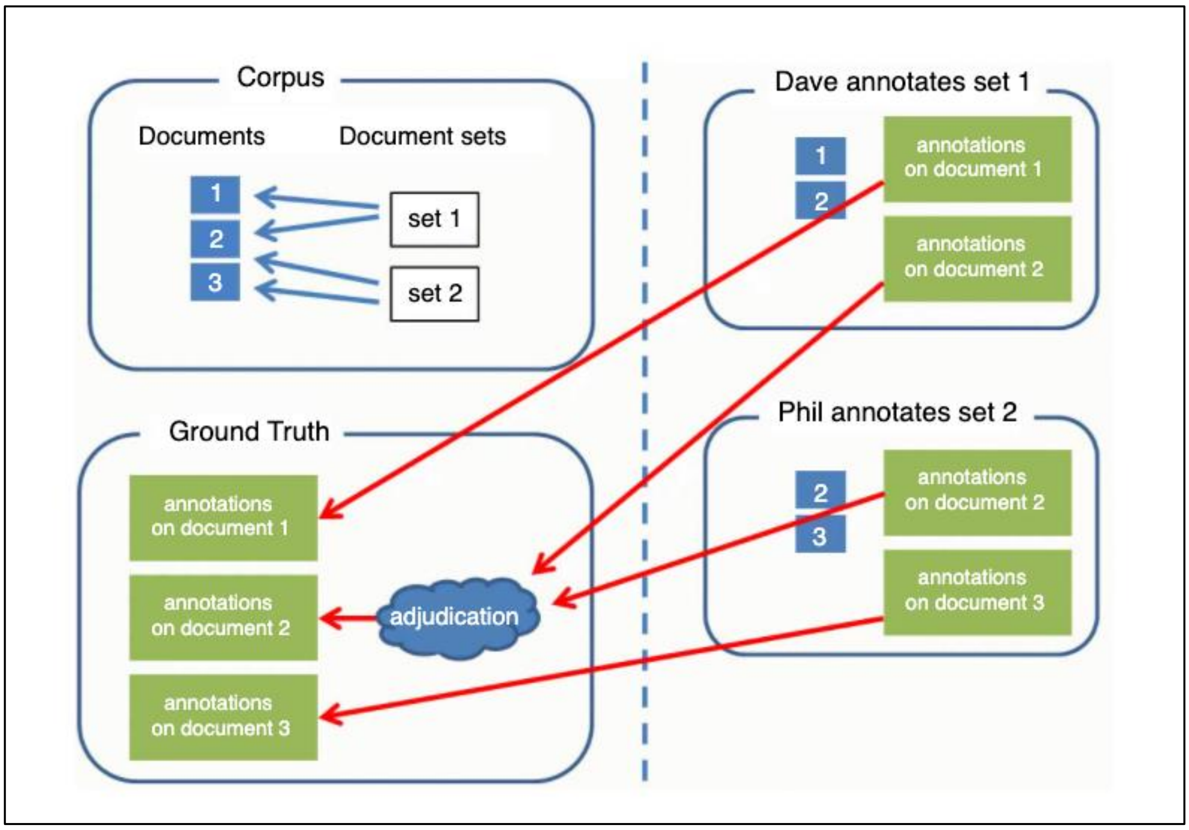
Corpus
Documents
Document sets
1
2
3
set 1
set 2
Dave annotates set 1
1
2
annotations on document 1
annotations on document 2
Phil annotates set 2
2
3
annotations on document 2
annotations on document 3
Ground Truth
annotations on document 1
annotations on document 2
annotations on document 3
adjudication

Figure 13. annotations and groud truth (Knowledge Studio 2020d)

The document set assigned for each annotator had a 30% document overlap. This means out of all the documents assigned for one annotator, 30% of the documents also existed in the other annotators list of documents. When the documents were returned, the documents that re not part of the documents will be promoted to ground truth, whereas the overlapping documents will be checked for any conflict. If there was a conflict, it will go through the adjudication process to resolve the issue and they are returned to both annotators until all conflicts were resolved. After all conflicts are resolved, the documents are promoted to ground truth. This was done to ensure that the annotation work was done correctly and to avoid/mitigate human error.

Furthermore, WKS was used to create entity types and relationships that were relevant for the industry. This thesis will mention 3 entities only as the others are out of scope. If we take the example of the printer company that has different models of printers, and there would be a need to fix them when they encounter a problem, WKS can be used to annotate the documents by creating "fault" entity, "solution" entity, and "printerModel" entity. Therefore, a relationship can be built in such a way that:

1. entity "printerModel" has a relationship "failedBy" with entity "fault"
2. Entity "fault" has a relation "fixedBy" with entity "solution"

This model was particularly effective with the THD_FSU_collection and SO_collection as the language of the query was similar with the language of the document ingested in WDS.

In the most ideal case, the annotators are SMEs that are experts within the domain of the content that is being annotated. But in most cases, as the process of annotation takes a lot of time and happens to be a tedious task, most would prefer not to do it. In this project WKS was used as an experimentational tool that was not annotated by SMEs but by the team that have undertaken the project. Therefore, the result is not expected to have a significant increase. None the less, the SMEs have assisted in building dictionaries to use and as part of the assessment process of the tool, SMEs have also been engaged in giving additional training and more dictionaries to use along the way.

4.2.1 Training Watson Assistant

Watson Assistant was used in two different instances in this project. Figure 1 shows these instances. The first instance of Watson discovery is to assist the call center agent in extracting more information out of the calling customer. This case is out of scope for this thesis and will not be covered here.

The second instance where Watson Assistant was used is to complement the WDS service functionality called query expansion covered in chapter 3.1.4. At the time of this project query expansion was in a beta phase, therefore, it came with its own challenges. The main issue encountered with query expansion was when using compound words. When using a compound word, instead of looking for the expanded compound word as one word, it separated the word into two different words and applied the logical OR operator. This yielded a result that was not sought out.

For example, living room in the sentence “the flower vase is inside the living room” is a compound word that describes a general everyday use room in the house, and that the flower vase can be found in this room. But if the words were to be separated, it will have an entirely different meaning, where it becomes,

“The flower vase is inside the living | The flower vase is inside the room | The flower vase is inside the living room”

To resolve this issue, the project resorted towards Watson Assistant to build the query expansion using Intent and Entity. Therefore, before going to WDS the query will be sent to Watson Assistant. If the intent is identified with the desired confidence level or above, and the entity matches a dictionary term, then the base word from Watson Assistant is either concatenated, or completely replaces the incoming word and a new query string will be constructed. This new query string is then sent to WDS to retrieve relevant documents.

Consider an example of the printer company that has official documents that address issues regarding paper being trapped halfway when printing. The NLQ could come from users as “*paper entrapment in printer*”. Due to similarities in language, solutions maybe found in *THD_FSU_collection* and *SO_collection*. But it is highly unlikely the search result will come up with an answer from an official documentation such as *CDP_collection*. This is due to words such as “paper”, “in” or “printer” are highly likely to be found in every manual, and the word “entrapment” is highly unlikely to exist in any manual. Instead, the word “stuck” or “wedged” maybe the official language used.

To solve this issue, two methods/flags could be introduced within Watson Assistant

1. Replace

This means the word “entrapment” would be replaced with one or more words that are listed as “synonym” in entities such as “stuck” or “wedged” as described in Chapter 3.2.2. Therefore, the search query will become “*paper* stuck *in printer” or “paper* wedged *in printer”.* If there is a need to make more emphasis on the replacing words, the query could also become “*paper* stuck wedged *in printer”*. This gives more weight on the word “stuck” and “wedged” as they have a similar concept in documents that are indexed to recognize concepts.

2. Concatenate

   In some other cases, instead of replacing the original word with the synonym listed in entity, the word could be left as is and the synonym(s) could be concatenated instead. By doing this, more weight will be given to the word as the concept of the concatenated words will be the same. In the previous example if the incoming query was “*paper* stuck *in printer”,* more emphasis maybe needed on the word “stuck” rather than any of the other words around it. But in reality, the other words may have more weight on the search results as they appear more frequently than the word stuck. Therefore, to give the word more weight, one could alter the query by adding synonym next to it, such that it becomes “*paper* stuck wedged trapped *in printer*” .

4.2.2 Training Watson Discovery Service

The training material was collected from SMEs in the company by considering the query input/natural language query (NLQ). As mentioned before there are two type of NLQs to consider. One coming from the customer (CFD) and the other from the technicians (OFD). CFD has been collected for many years by the company as one of the entry columns in service order record. On the other hand, OFD has never been documented and therefore, the SMEs needed to come up with an NLQ with the assumption of how a question would be asked if certain problem arises. As can be seen in the code snippet in chapter 3.1.5, document id that was relevant for these NLQs with their relevancy scores were also provided for all the collections.

The training material that was collected from SMEs was broken into two sets, a training set and a testing set. K-fold cross-validation (CV) was used to do training and test the efficiency of the result, where $K = 10$. WDS is then trained using $K - 1$ of the folds and tested with the rest of the dataset, which in this case is 1. Consider a cross-validation with $K = 5$ and the overall relevancy training material is 100, this means the training material can be broken into 5 datasets, each containing 20 training material ($\frac{100}{5}$). 4 will be used as training material and 1 will be used for testing. This will be run 5 times with different training set and test set each time.

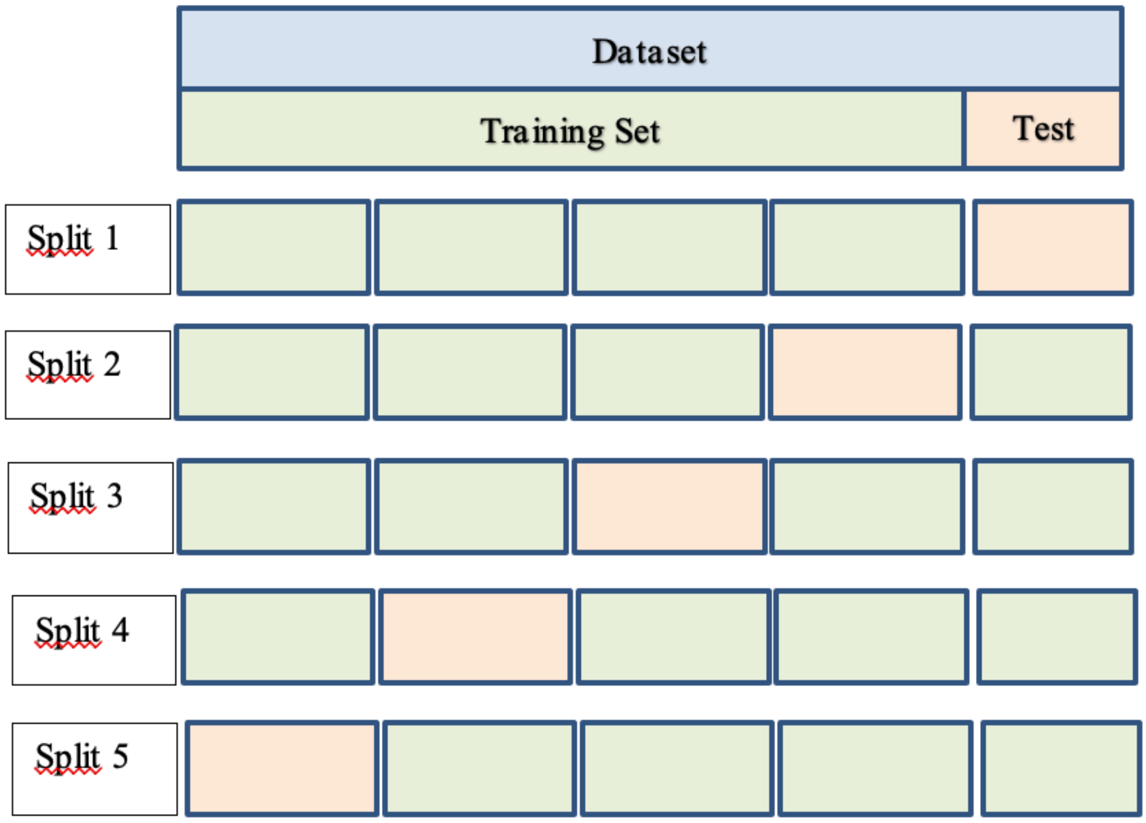

Figure 14. K-fold cross validation

As shown in Figure 14, the K-fold method is used to

1. By dividing the dataset into training set and test set, which in this case was 90% training set and 10% testing set, the model will not be exposed to overfitting issues that machine learning model could exhibit. Overfitting happens when the model learns the training set that is correct and also the noise too well that it almost

becomes a mapping exercise. To resolve this the model will be trained with the majority of the training dataset and will be tested with a training dataset that it has never seen before.

2. The activity is repeated K times. In the diagram shown below as example K=5, and hence 5-fold. This method is used to make sure that the testing does not contain a biased result. In other words, it offers a less biased and contains an estimate of a model that is less optimistic.

4.2.3 Enrichment of Training Set with Negative Samples

At the initial phase of the project, it was agreed that the training data would be collected by using an excel file as it would be the easiest way to do this type of exercise. The reason why it is considered to be an easier way is due to the amount of time it takes for a knowledgeable SME to look at a problem description (NLQ) and think of a document that could provide a perfect answer. Therefore, the training material that was collected did end up having only positive examples and no negative examples. To be able to generate these negative examples, natural language classifier (NLC) was used.

The documents that were labelled as relevant were first used to create categories such that there would be category A, category B, category C, and so on. The positive examples that were obtained from the training material were ingested into the NLC as examples of these categories. Each category would contain at least 10 examples, and the model is trained and tested the same way as Discovery.

It is important to mention that this is not the only tool that could be used for this purpose. Watson Assistant can also fulfil this requirement.

NLC can be queried using natural language query (NLQ). The response is a list of 10 classes that the NLQ could belong in order of confidence score for each class in descending order. Therefore, based on the desired minimum confidence score, the document category can be classified as not relevant for that specific NLQ. This returned result is then used as a training material for Discovery.

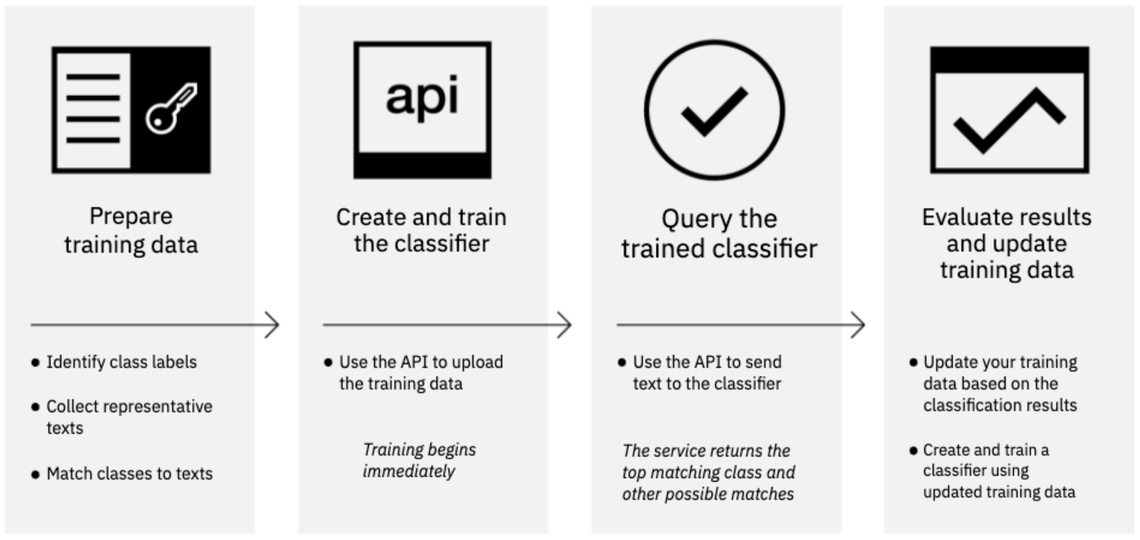
Prepare training data
• Identify class labels
• Collect representative texts
• Match classes to texts
api
Create and train the classifier
• Use the API to upload the training data
Training begins immediately
Query the trained classifier
• Use the API to send text to the classifier
The service returns the top matching class and other possible matches
Evaluate results and update training data
• Update your training data based on the classification results
• Create and train a classifier using updated training data

Figure 15. Natural Language Classifier (2020a)

As the project progressed it was agreed that a web interface will be built for collecting training material by loading all the NLQ into this interface. When the SMEs click on this NLQs, it will be directly used to fetch documents from Discovery for ranking. This meant that SMEs would need to go though few documents before finding one that they could label as relevant (4-star document), and the documents before these will get anything between 1-star to 3-stars. This method eliminated the need to undertake the task of generating negative examples, but it should be acknowledged that training material may not always have negative examples labeled by SMEs and this could be a valid way to generate them if needed.

5 Results

There are few ways that results from a cognitive search engine such as Watson can be measured. One way is to see if the confidence score returned from WDS meets certain threshold. For example, if a confidence greater than 75% is considered acceptable, then all results that have a confidence above 0.75 will be accepted and others will be considered as irrelevant or failed case.

```
{
    "matching_results": 4,
    "session_token": "1_2gYuWJyaWx792Ni4_DQ4C5cbnW",
    "retrieval_details": {
        "document_retrieval_strategy": "relevancy_training",
    },
    "results": [
        {
            "id": "eea16dfd5fe6139a25324e7481a32f89_13",
            "result_metadata": {
                "confidence": 0.7893963975910735,
                "score": 0.5006834
            }
        }
    ]
}
```

One of the way that the effectiveness of the model can be measured by referancing the confidence level of the model

1. True negative

   The model has low confidence for the document that was rated as negative example

2. True positive

   The document is relevant, and the model has high confidence

3. False positive

   The model has high confidence, but the document was rated as not relevant

4. False negative

   The model has low confidence, even though the document was rated relevant.

The other option of measuring weather the relevancy training was effective is by looking at the return position (RP) of the document that was marked relevant. The relevancy training can be considered to be successful if a certain percentage point has been achieved within the first X number of returned documents. For example, if the document that was marked as relevant (score 10) can be returned within the first 5 results 80% of the time, then the training can be considered as successful. Any number below that for the first 5 documents will mean that the relevancy training was not successful and has to be re-examined.

The project went with the second method, hut had a relatively high criteria of measuring results. The model was only deemed a success if it was able to get a document that was rated using 4-stars. Any other rating within the range is counted as fail, even if the rating is 3-star. Furthermore, NLQs that do not have documents that are rated with 4-star rating, if found in the test set, would automatically be considered as fail. The measurement that was used in this exercise is a binary scale; pass (1) or fail (0). Due to this high expectation of performance from the tool, the result is significantly impacted as a “good enough” document rated with 3-star or 2-star will now have to be considered as irrelevant and scored to have a value of zero (0).

All results presented are based on a 10-fold CV outcome.

5.1 Untrained

Untrained collection is a Discovery instance that has not been trained at all and therefore, has not been annotated or trained. This was what Discovery was able to achieve out of the box.

For *THD_FSU_collection*, the result of the untrained WDS instance return the result seen below. The result for the project was considered to be good considering that this is untrained collection. This collection was able to achieve 76% within the first 5 returned position (RP), and 89% for the first 10 documents.

RP	**Untrained**
1	48 %
2	63 %
3	65 %
4	75 %
5	76 %
6	81 %
7	83 %
8	84 %
9	86 %
10	89 %

Table 1. Untrained THD_FSU_Collection

For *CDP_collection*, the result of the untrained WDS instance return the result seen in Table 2. The result for the project was considered to be good considering that this is untrained collection. This collection was able to achieve 72% within the first 5 returned position (RP), and 93% for the first 10 documents.

RP	**Untrained**
1	41 %
2	53 %
3	59 %
4	71 %
5	72 %
6	83 %
7	84 %
8	86 %
9	88 %
10	93 %

Table 2. Untrained CDP_Collection

5.2 Trained

A trained collection of Discovery in this context is a Discovery instance that has been trained with roughly 370 examples of NLQ pointing to one or more documents. Not all of these NLQs have documents are relevant, in some cases there might not be any document labelled as relevant for NLQ (4-star). The testing set had a total of 41 NLQs and the results you see below was drown from these testing sets. These test sets have never been shown to the model before.

The training on THD_FSU_collection has improved the result. The major impact that the training had was on the first and second returned position (RP) document. The improvement on the first RP was 5 percentage points or 10.4% increase, and the second RP was 4% points or 6% increase. As the RP goes down, so does the improvement made by the training.

RP	**Untrained**	**Trained**
1	48 %	53 %
2	63 %	70 %
3	65 %	77 %
4	75 %	81 %
5	76 %	86 %
6	81 %	88 %
7	83 %	89 %
8	84 %	92 %
9	86 %	92 %
10	89 %	93 %

Table 3. Trained THD_FSU_collection

On the other hand, the improvement made from training the CDP_collection improves as the RP goes down. There is 2 percentage points improvement made on the first RP. That number goes to 23 percentage point (~32%) on the 5th RP.

RT	**Untrained**	**Trained**
1	41 %	43 %
2	53 %	71 %
3	59 %	79 %
4	71 %	84 %
5	72 %	95 %
6	83 %	97 %
7	84 %	100 %
8	86 %	100 %
9	88 %	100 %
10	93 %	100 %

Table 4. Trained CDP_collection

5.3 Annotated and Trained

The model had:

- 300 documents annotated
- 4 Entities
  - 283 components
  - 79 problems
  - 24 corrective actions
  - 76 codes (IoT)
- Relationships
  - Components *encounter* problems
  - Components *emits* codes
  - Problems *fixedBy* corrective actions
  - Codes *generate* corrective actions

The result on trained model verses the trained and annotated model is shown on Table 5.

Due to luck of SME annotators, the project only deployed WKS model on THD_FSU_collection. Neither the ontology nor the annotation was not done by SMEs; therefore, the result clearly shows a negative impact due to a model annotated by non-SMEs.

Looking at the major impact for this result, it can be seen that there is a small amount of sample that have been used. For example, the number of documents in the collection verses the 300 documents annotated could play a big impact in the outcome. Also, the entities are limited to 4, which could arguably be a small set of entities. When looking deeper into these entities, aside from the "component" entity, the other ones also have a small representation. For example, one can argue that there are more than 24 corrective actions for a product that contains 283 components, as well as more problem than 79 caused by these components.

RT	**UnTrained**	**Trained**	**Trained + Machine Learning**
1	48 %	53 %	52 %
2	63 %	70 %	70 %
3	65 %	77 %	76 %
4	75 %	81 %	80 %
5	76 %	86 %	83 %
6	81 %	88 %	86 %
7	83 %	89 %	88 %
8	84 %	92 %	90 %
9	86 %	92 %	91 %
10	89 %	93 %	92 %

Table 5. Trained + WKS model on THD_FSU_collection

5.4 Query Expanded and Trained

As the language coming as query is similar to the language found in both THD and FSU datasets, there was no need to apply the query expansion on the THD_FSU_collection. On the contrary, the CDP_collection contains documents that the manuals for the product, and as such contain formal language rather than everyday language used at a work setting.

Therefore, the CDP_collection needed to be query expanded to “translate” everyday language to the technical and official language.

As Table 6 shows, the first RP had a significant improvement of 10 percentage point (~19%). But as the RP goes down, the result obtained by the query expansion does worse when compared with the model that only had been trained. One explanation for this could be the number of expanded terms were very small considering that there were more than 5000 documents in the CDP_collection collection and therefore, not every possible scenario was covered.

- 32 unidirectional expanded terms and
- 17 bidirectional expanded terms.

The most common scenario that proved to be troublesome were noticed early on and mitigated with the query expansion.

The other explanation for this result is that the query expansion was used when it was at its beta state and was not production ready. This puts limitation on the anticipated result.

RP	**Untrained**	**Trained**	**Trained + QE**
1	41 %	43 %	53 %
2	53 %	71 %	62 %
3	59 %	79 %	74 %
4	71 %	84 %	81 %
5	72 %	95 %	84 %
6	83 %	97 %	88 %
7	84 %	100 %	88 %
8	86 %	100 %	91 %
9	88 %	100 %	93 %
10	93 %	100 %	93 %

Table 6. Query expansion on CDP_collection

5.5 Watson Assistant and Trained

Watson Assistant (WA) was used to tackle the issue of matching the incoming query (NLQ) to the ingested official document (manuals) in CDP_collection. This was previously attempted with query expansion but as mentioned the tool was in its beta phase and was not able to address the issue. Therefore, the project took WA as alternative. As shown in the result below, WA had made a much better attempt in answering this issue. The result has increased significantly on the first RP and the second one, 53.5% and close to 10% respectively. It seems to have no impact on the 3rd and 5th RP but there was a slight improvement of ~5% on the 4th. The project had

- 272 intents for recognizing which components of the products were mentioned
- 70 entities that are used for dictionary.

It is worth nothing at this pointing that the amount of improvement that could be done through training will become insignificant as more training data is ingested, therefore, significant improvement of the result becomes entirely dependent on techniques such as adjusting the NLQ and annotating the documents.

RP	**Untrained**	**Trained**	**Trained + WA**
1	41 %	43 %	66 %
2	53 %	71 %	78 %
3	59 %	79 %	79 %
4	71 %	84 %	88 %
5	72 %	95 %	95 %
6	83 %	97 %	96 %
7	84 %	100 %	96 %
8	86 %	100 %	98 %
9	88 %	100 %	98 %
10	93 %	100 %	98 %

Table 7. Watson Assistant on CDP_collection

6 Discussion

A cognitive based search engine that could unlock insights from unstructured data including proprietary data has become one of the main focus areas for big companies such IBM, Microsoft and Amazon. During the past few years there have been a significant improvement made with this tools that are now available for a wide verity of application. There has also been a realization and high demand from companies to unlock insights from their high-volume data source, unstructured data. The project conducted with IBM Watson was focusing on an industrial client, but there have been few projects with other disciplines such as insurance, medical sectors, HR, and banks.

Even though, there have been a high demand for the technological advances in natural language understanding (NLU), and there have been many brave companies that have taken the step to try such technologies, there are still some issues within the discipline to be considered mature. One major challenge that is associated with AI tools in general but also specifically in NLU regarding extracting insights from unstructured data is the readiness of data. This is to say that, even though many industries are aware of such technologies and are willing to incorporate them, the data that have been collected have not been collected with the consideration of AI, and therefore, contains a lot of noise. The evidence for the abundance of data is damning. Mohammed et al, (2018) agrees in regard to the medical industry about data; “[the]problem in health care systems is not the lack of data; it is the lack of information that can be utilized to support critical decision making”. (p.2)

The second challenge is the maturity of the technologies that are in the market. Even though there has been a lot of progresses that have been made in this regard from the ability to read unstructured data from different format to automatic annotation on wide range of industry materials, the technology is still not as advance as those technologies that deal with structured data.

The process of training these tools is also time intensive that many companies may not have the time, money, or resources to go through the exercise, even though the technology is very much relying on the involvement of SMEs to do most of the work. Even when companies agree on doing the task of annotating the documents assigning their top employees and going through the material to provide positive feedback for the training material, most of the document that gets returned and ranked happens to be irrelevant for the natural language query if the result is directly queried from Discovery. This ends up frustrating employees and the companies that are willing to undertake this kind of exercise. One solution that sould be considered is a simplified way of collecting the training data in excel sheet or CSV file. This will allow SMEs to look at the NLQ and give the relevant example without having to worry about the negative examples.

There are few issues that industries should emphasize in the future, one of which is to recognize the initial training process that SMEs spend much of valued time is prone to bring more negative training example than it is to bring positive training examples if querying directly from Discovery. In other words, SMEs fine more "not relevant" examples for a natural language query (NLQ), than they do a "relevant" document. Unfortunately, the tools that are in the market do not regard this information as a training material other than establishing a lower baseline. This means, if the range of non-relevant to relevant is from 0 to 10, the negative example will be scored as 0 and the relevant example as 10, and the tool will establish this range by looking at this example so as to scale it to 0-100. Any other document that has other value between the range of 0 to 10 will be seen in relation to these base lines. Even though it is enough to establish this baseline with one example for negative and one example for positive, the rest of the negative examples SMEs most commonly find, and mark, are thrown out and will be of no use. This should be something that these leading industries should focus on to utilize by making these negative examples as relevant to the training of the cognitive tool as the positive ones are currently.

The other issue is the emphasis that is put on the incoming natural language query and matching it to the documents that are ingested to these tools. One example that WDS is tackling this problem is by introducing query expansion. But as can been seen from the result on this thesis, the tool can be unpredictable and, in some cases, can have a negative impact. There is a continuous development that is going on with WDS's query expansion and this will likely have a positive impact in the future, but currently the emphasis on the incoming query even though highly impactful, have not had much attention.

7 Conclusion

In this thesis, general concept of big data is introduced along with the “5V” characteristics that collectively make it challenging to unlock insights. The benefit of unlocking insights and the ultimate challenges around big data have also been briefly shown. In addition, the cognitive tools that are undertaking these challenging tasks of unlocking insights from big data have also been discussed.

This thesis shows with a real-life example, the different methods and technologies that could be used to achieve an acceptable cognitive based search engine for industrial sector, but also acknowledges that similar approaches can be applied to any other sector. This thesis puts more emphasis on IBM Watson tools as the practical project were undertaken using those technologies. Furthermore, the thesis tackled the issue of improving search engines from the ingested documents using methods such as annotation, indexing, filtering and training the model, as well as from improving the quality of the incoming query by using available tools such as query expansion and Watson Assistant.

Watson discovery has been heavily used for the project, and as such the output has been examined with first evaluating how it performs out of the box. The result was arguably acceptable as there has not been any training given or annotation done to the documents. This result underlines the importance of data engineering before ingesting the documents. Then the training examples given by SMEs were introduced to the model, which brough a significant improvement to how the model performed. The trained model had improved the result by 10% points when considering the correct document being returned within the first 5 documents. As the number of training examples increase, the improvement increase becomes insignificant. When the model reaches this stage the more impactful strategy is to incorporate annotation to better index the ingested documents, therefore, WKS was introduced. Unfortunately, the task associated with the annotation and building dictionaries is quite intensive for SMEs and they opted out. The IBM team that has undertaken the project build a sample WKS model, and therefore, the result that was obtained had a reverse effect on the model. But it is good to note that annotation of industry documents should be left to the industry professionals to have a positive impact. The other methods that were introduced to improve the search result after the training examples have stopped bringing significant impact are query expansion and Watson Assistant. These two different tools try to tackle the problem from the incoming query's angle. At the time of this project, query expansion was not mature to undertake the task and therefore the result has only improved the first RP and did not have much significance in the other RPs. Therefore, Watson Assistant was introduced and as can be seen from Table 7, the result improved for the 4 RPs.

The thesis also discusses how to generate negative examples from positive training materials when the circumstances of gathering training material does not allow or may not be conducive for negative examples.

References

Rai, R. (2017). Intricacies of Unstructured Data. *EAI Endorsed Transactions on Scalable Information Systems, 4*(14), . doi:10.4108/eai.25-9-2017.153151

Pouyanfar, S., Yang, Y., Chen, S., Shyu, M. & Iyengar, S. S. (2018). Multimedia Big Data Analytics: A Survey. *ACM Computing Surveys (CSUR), 51*(1), pp. 1-34. doi:10.1145/3150226

Norjihan Abdul Ghani, Suraya Hamid, Ibrahim Abaker Targio Hashem, Ejaz Ahmed, Social media big data analytics: A survey, Computers in Human Behavior, Volume 101, (2019), Pages 417-428, ISSN 0747-5632, <https://doi.org/10.1016/j.chb.2018.08.039>. http://www.sciencedirect.com/science/article/pii/S074756321830414X

Hariri, R. H., Fredericks, E. M., & Bowers, K. M. (2019). Uncertainty in big data analytics: Survey, opportunities, and challenges. Journal of Big Data, 6(1), 1-16. doi: http://dx.doi.org.ezproxy.jyu.fi/10.1186/s40537-019-0206-3

Jeff Desjardins, World Economic Forum (2019). Accessed 2020, Oct 20, *How much data is generated each day?* https://www.weforum.org/agenda/2019/04/how-much-data-is-generated-each-day-cf4bddf29f/

Jeff Desjardins, Visual Capitalist (2019). Accessed 2020, Oct 20, *What Happens in an Internet Minute in 2019? <https://www.visualcapitalist.com/what-happens-in-an-internet-minute-in-2019/>*

Emily Adler. (2016). Social media engagement: The surprising facts about how much time people spend on the major social networks. Retrieved from http://www.businessinsider.com/social-media-engagement-statistics-2013-12.

H. V. Jagadish, Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M. Patel, Raghu Ramakrishnan, and Cyrus Shahabi (2014). Big data and its technical challenges. Commun. ACM 57, 7 (July 2014), 86-94.

González, M.C., Hidalgo, C.A. and Barabási, A-L. Understanding individual human mobility patterns. Nature 453, (June 5, 2008), 779–782.

Mohammed, E., Mohamed, M., Naugler, C. & Far, B. (2017). Toward leveraging big value from data: Chronic lymphocytic leukemia cell classification. Network Modeling Analysis in Health Informatics and Bioinformatics, 6(1), pp. 1-17. doi:10.1007/s13721-017-0146-9

.O, A., A.A., I., S.O, K. & F.Y, O. (2016). Big Data and Cloud Computing Issues. *International Journal of Computer Applications, 133*(12), pp. 14-19. doi:10.5120/ijca2016907861

Chen, Y., Li, Z., Wang, X., Feng, J., & Hu, X. (2010). Predicting gene function using few positive examples and unlabeled ones. *BMC Genomics, 11*(2), S11. doi:http://dx.doi.org.ezproxy.jyu.fi/10.1186/1471-2164-11-S2-S11

Xu, Z., Qi, Z. & Zhang, J. (2014). Learning with positive and unlabeled examples using biased twin support vector machine.(Report). *Neural Computing & Applications, 25*(6), p. 1303. doi:10.1007/s00521-014-1611-3

., A. R. (2014). BIG DATA: PRIVACY AND INCONSISTENCY ISSUES. International Journal of Research in Engineering and Technology, 03(19), pp. 812-815. doi:10.15623/ijret.2014.0319147

Dutta, P., Rahman, S. S., Baishya, B. K. & Acharjee, P. (2018). A study: On Big Data Implementation and its Challenges. *International Journal of Engineering Science and Technology, 8*(2S), pp. 95-97. doi:10.21817/ijest/2018/v10i2S/181002S015

Price, M., & Ball, P. (2014). Big data, selection bias, and the statistical patterns of mortality in conflict. *The SAIS Review of International Affairs, 34*(1), 9-20. Retrieved from https://search-proquest-com.ezproxy.jyu.fi/docview/1552151757?accountid=11774

Muhammad, I. & Yan, Z. (2015). SUPERVISED MACHINE LEARNING APPROACHES: A SURVEY. *ICTACT journal on soft computing, 5*(3), pp. 946-952.

doi:10.21917/ijsc.2015.0133

Discovery (2020a). Accessed 2020, Nov 6, *Discovery-About, Retrieved From* https://cloud.ibm.com/docs/discovery?topic=discovery-about

Discovery (2020b). Accessed 2020, Nov 6, *Discovery data, Retrieved From* https://cloud.ibm.com/docs/discovery-data?topic=discovery-data-about

Discovery (2020d). Accessed 2020, Nov 9, *Adding Enrichments, Retrieved From* https://cloud.ibm.com/docs/discovery?topic=discovery-configservice#adding-enrichments

Discovery (2020e). Accessed 2020, Nov 9, *Query Reference, Retrieved From* https://cloud.ibm.com/docs/discovery?topic=discovery-query-reference

Discovery (2020f). Accessed 2020, Nov 9, *Query Reference, Retrieved From* https://cloud.ibm.com/docs/discovery?topic=discovery-query-concepts

Discovery (2020g). Accessed 2020, Nov 9, *improving result relevance with the tooling, Retrieved https://cloud.ibm.com/docs/discovery?topic=discovery-improving-result-relevance-with-the-tooling*

Watson Assistant (2020a). Accessed 2020, Nov 10, Watson Assistant, *Retrieved From* https://www.ibm.com/cloud/watson-assistant-2/

Watson Assistant (2020b). Accessed 2020, Nov 10, About Watson Assistant, *Retrieved From* https://cloud.ibm.com/docs/assistant?topic=assistant-index

Watson Assistant (2020c). Accessed 2020, Nov 10, Assistant Language Support, *Retrieved From https://cloud.ibm.com/docs/assistant?topic=assistant-language-support*

Watson Assistant (2020d). Accessed 2020, Nov 10, Creating Intents, *Retrieved From* https://cloud.ibm.com/docs/assistant?topic=assistant-intents

Watson Assistant (2020e). Accessed 2020, Nov 10, Adding Entities, *Retrieved From* https://cloud.ibm.com/docs/assistant?topic=assistant-entities

Knowledge Studio (2020a). Accessed 2020, Nov 10, About, *Retrieved From* https://cloud.ibm.com/docs/watson-knowledge-studio?topic=watson-knowledge-studio-wks_overview_full

Knowledge Studio (2020a). Accessed 2020, Nov 10, Rules, *Retrieved From* https://cloud.ibm.com/docs/watson-knowledge-studio?topic=watson-knowledge-studio-rule-annotator

Knowledge Studio (2020b). Accessed 2020, Nov 10, User roles in Knowledge Studio, *Retrieved From https://cloud.ibm.com/docs/watson-knowledge-studio?topic=watson-knowledge-studio-roles*

Knowledge Studio (2020c). Accessed 2020, Nov 10, creating dictionaries, *Retrieved From https://cloud.ibm.com/docs/watson-knowledge-studio?topic=watson-knowledge-studio-dictionaries#wks_projdictionaries*

Knowledge Studio (2020d). Accessed 2020, Nov 10, Adding documents for annotation, *Retrieved From https://cloud.ibm.com/docs/watson-knowledge-studio-data?topic=watson-knowledge-studio-data-documents-for-annotation*

Natural Language Classifier (2020a). Accessed 2020, Nov 2, about, *Retrieved From https://cloud.ibm.com/docs/natural-language-classifier?topic=natural-language-classifier-about*

Content Delivery Platform (2020). Accessed 2020, Nov 17, content delivery platform, *Retrieved From https://www.fluidtopics.com/content-delivery-platform/*