Pre-standardization report on development, usage and maintenance of Chatbots

UNDER THE GUIDANCE OF:

KSHITIJ BATHLA

SCIENTIST-C/ DEPUTY DIRECTOR

ELECTRONICS & IT DEPARTMENT

SUBMITTED BY:

KAMALPREET KAUR

INTERN, BIS

INTERSHIP TIMELINE:

June - July 2024

ACKNOWLEDGEMENT

I would like to extend my heartfelt gratitude to the Bureau of Indian Standards (BIS) for granting me the opportunity to undertake this internship and contribute to the Study of documents utilized in the development, usage, and maintenance of Chatbots. Special thanks to Shri Kshitij Bathla, Scientist-C and Deputy Director, Electronics & IT Department, for his invaluable mentorship, guidance, and unwavering support throughout the entire duration of this internship. I am deeply appreciative of Shri Kshitij Bathla's expertise and dedication, which have been instrumental in shaping the scope and direction of this study. His insightful feedback and constructive critiques have immensely enriched the quality and depth of the research.

I also want to express my sincere appreciation to the Mrs. Reena Garg, Scientist G, Head of Electronics and IT Department (Bureau of Indian Standards), for creating a conducive and collaborative environment. The opportunities for knowledge-sharing and exposure to real-world challenges have been truly rewarding.

Furthermore, I am indebted to all the experts and stakeholders in the field who generously shared their insights and expertise during the visit at National Informatics Centre (NIC). Their inputs have played a pivotal role in shaping the methodology and conclusions of this report.

Lastly, I wish to acknowledge the support and encouragement of my family and friends. Their belief in my abilities and constant motivation have been a driving force throughout this internship.

This study on the development, usage, and maintenance of Chatbots would not have been possible without the collective efforts and support of all those involved. I am sincerely grateful to each and every one of you for being an integral part of this enriching experience.

Thank you.

Sincerely,
Kamalpreet Kaur
Email: kkamal101203@gmail.com, kkaur_be21@thapar.edu
Intern, Bureau of Indian Standards (BIS)

EXECUTIVE SUMMARY

Artificial intelligence has recently integrated chatbots as one of the most important technologies to redefine human–machine interaction. They have been integrated into areas such as customer service and personal assistance in remarkable ways to help boost the level of efficiency and user experiences in these areas. This report provide details of the different importance or application issues involved with the use, design, and maintenance involved in using chatbots to assure full understanding of their working. It shows the need for standardization in this process that will ensure uniformity, consistency, and quality.

The sources most commonly used encompass scientific articles, technical documents, user-generated content, and real conversational data that, in turn, provide unique features and functions of the developed chatbot in the dimensions of natural language understanding, knowledge about the domain, and conversational abilities. This research work therefore sets out to review the most common forms of documents used in chatbot creation, elaborate on methods for information extraction and integration to provide a clearer understanding of best practices and challenges in design, and add to the standardization of the same with a view to developing and using effective and consistent methods. It also addresses ethical aspects of such diverse datasets and data protection issues, with an emphasis on the laws in place.

The study provides key insights to stakeholders by detailing the documents used in chatbot construction. This will foster a profound understanding of chatbot technology as a way of improving intelligent and highly responsive chatbot mechanisms. The standardization will thus make the chatbot systems more trusted, reliable, and wide-adoptable, thus promoting commendable uses across industries.

TABLE OF CONTENT

S.No.	Content	Page No.
1.	Introduction	6
2.	Scope	6
3.	Overview of Chatbots 3.1 Types of chatbots 3.2 Chatbot use cases	6
4.	Key Concepts 4.1 Key terms 4.2 Advanced Technologies in Chatbot Development 4.3 Chatbot stakeholders 4.4 AI application functional characteristics applicable to Chatbots 4.5 AI application non-functional characteristics and considerations applicable to Chatbots	9
5.	Chatbot Development Lifecycle 5.1 Chatbot Requirements 5.2 Chatbot Design 5.3 Data for training models 5.4 Modeling 5.5 Testing and validation 5.6 Chatbot Deployment 5.7 Monitoring and Maintenance	21
6.	Insights from field visit 6.1 Domain Identification and Requirement Elicitation 6.2 Data Collection for Chatbots 6.3 Building a Conversation 6.4 Deployment 6.5 Post Deployment	61
7.	Conclusion	63
8.	References	65
9.	Bibliography	68

LIST OF FIGURES

S.No.	Figure	Page No.
1.	Data life cycle framework	28
2.	Processes across the multiple stages	29
3.	The relationship between the DLC model and the DQPF	30
4.	One-hot encoding	33
5.	Bag of Words (BoW)	33
6.	TF-IDF	33
7.	Word2Vec	34
8.	General Chatbot Architecture	35
9.	Dialog Manager and flow of information	38
10.	Dialog Management approaches	39
11.	DM model defined using a finite state machine (FSM)	40
12.	Accuracy	48
13.	Precision	48
14.	Recall	49
15.	F1 Score	49
16.	Confusion matrix	49

1. Introduction

Recently, Artificial Intelligence (AI) technologies have been rapidly developing, and their influence is being felt in every sphere of business. Among the most vivid applications of AI in recent years are chatbots. Using natural language processing (NLP), natural language understanding (NLU), machine learning (ML), and deep learning (DL), chatbots have grown to be a tool for enhancing customer engagement and efficiently supporting customers, as well as streamlining different business processes. As chatbots are being increasingly adopted, it becomes very important to make sure that the design, application, and maintenance meet qualitative and reliability standards.

This paper will try to determine whether the current standards on AI, software, and data quality are sufficient within the context of chatbot technologies. We will assess whether the current standards are adequate to cover the challenges and requirements of chatbots, which arise as peculiar, using very basic building blocks in the creation of chatbots, like root models and the above-mentioned AI techniques.

While trying to identify gaps between the current standards and industry practices related to chatbots. This paper contributes a detailed review of the current standards based on case studies and real implementations that can give an all-rounded evaluation. The suggestions entail recommendations for enrichment or the creation of new standards that focus on evolving chatbot technologies. Indeed, such efforts will help to a great extent in bridging standards and industrial practice toward a better effectivity and reliability of chatbots across different applications.

2. Scope

The current research is focused on the fundamentals of chatbot development, including root models, natural language processing, natural language understanding, machine learning, and deep learning. The current standards for artificial intelligence, software, and data quality are investigated with regard to applicability in chatbots. This is mainly about checking if the existing standards would be enough or sufficient for developing, using, and maintaining chatbots, and whether there is a need for a standard that is specific to chatbots.

Furthermore, we are working to identify the gaps between the current standard and industry practices related to chatbots. While evaluating the standards that exist today and industry practices in order to come up with a comprehensive review of the current standards and make suggestions for improvements or new standards that might be more suitable to meet the needs of chatbot technologies.

3. Overview of Chatbots

A chatbot is a computer program designed to simulate human-like conversation and be able to understand natural language and interact with users through text or voice. Modern chatbots use

advanced conversational AI methods such as NLP and ML to both understand unstructured human language and take the appropriate action accordingly. As they process and analyze the data, chatbots learn from it to refine their responses over time, increasing how well they can automate tasks or provide information/assistance to users [21], [22].

3.1 Types of Chatbots

3.1.1 Rules-Based, Menu-Based, or Button-Based Chatbots

Rule-based chatbots' operational modalities depend on conditional logic and interactive menus; they work as advanced, interactive FAQs. They take the user down set paths by asking pointed questions and then having the relevant answers. For example, a chatbot involved in e-commerce would ask, "Are you looking for product information or order status?" Depending on how the user responds, the bot will channel the conversation along pre-validated routes. These chatbots can be easily trained for specific queries but often miss the unexpected or complex questions. They miss the details, forcing users to repeat information—this is frustrating and might require live support [23].

3.1.2 AI/ML and Generative Chatbots

AI-driven chatbots use NLP and NLU to handle various types of user queries, like an airline chatbot that answers, "Will I be able to change my flight?". They get trained over time, and with machine learning, they enhance the user experience, which gets automated.

Chatbots run by NLP and text-to-speech allow for the possibility of voice interaction. The most recent examples, Alexa and Google Home, are systems that process spoken commands, further improving customer interaction and decreasing waiting time.

Chatbots, such as ChatGPT, are instances of generative AI that can imitate human responses and generate content in words, images, and even sound based on large language models. They facilitate professional and warm interactions without action by a human, thereby engaging the customers [23].

3.2 Chatbot Use Cases

Chatbots are used in numerous fields because they are useful tools for interaction and constant, 24/7 support. Here are three examples of chatbot applications across different sectors [24]. Here are three examples of chatbot applications across different sectors:

- Customer Service Chatbots: These bots are effective in managing customers’ questions, providing accurate answers to frequently asked questions, assisting customers in solving problems, and passing through complicated client queries to other real agents. This has the effect of rationalizing customer support and relationships and improving service delivery.

- Sales Chatbots: In sales, chatbots perform lead management as well as cross-selling and upselling by conversing freely with customers. They actively help in completing transactions and counseling clients on how to make a purchase, hence playing a vital role in increasing sales revenue.
- Healthcare Chatbots: In healthcare, for instance, chatbots are used in symptom tracing exercises, scheduling appointments, and even managing pills. They contain data critical to insurance, patient information and education, care service accessibility, and organizations' performance.

Here, we only pointed to some of the many prospects that show how chatbots are changing industries. Starting with finance and ending with real estate, they are universally applicable instruments that help to make the work more efficient, increase the level of customer satisfaction, and build a successful business.

4. Key Concepts

Before moving on to the chatbot development life cycle, it is important to understand the basic concepts and Frameworks used in chatbot development. There are some of the characteristics of AI applications that are also described in this clause. Given that chatbot is an AI application, it means that chatbots also have similar stakeholders.

Terms related to natural language processing, machine learning concepts, terms related to trustworthiness of AI systems and applications and AI system lifecycle are mentioned in ISO/IEC 22989:2022, Information technology — Artificial intelligence — Artificial intelligence concepts and terminology. Some of the information from this standard is used in this clause [1].

4.1 Key Terms

To improve the understanding in the construction of the chatbots we need to understand supervised training models based on NLP and ML and this will help us to better understand the user's intention and to extract entities from the text. Below mentioned are a few terms that we need to know before starting with chatbot development [34].

4.1.1 Intents

Intents refer to what the user wants or what the purpose or goal of the query is. For example, if a user asks a question like, “What is the weather like today?” The intent is to obtain information about the current weather. In this case, chatbots can determine the correct intent “weather” in the queries made by the users, hence providing the right responses. With the help of intent recognition models, chatbots are capable of recognizing numerous types of user’s requests, thereby improving the possibilities for positive interaction with a user [25].

4.1.2 Entities

Entities refer to specific pieces of information that provide additional context and modify the intent of the user’s message. By recognizing and extracting entities, chatbots can better understand the user’s needs and provide more accurate and relevant responses [25].

There are two types of entities commonly used in chatbots:

- System-defined entities: System-defined entities are pre-built within the chatbot system and include commonly used data such as dates, locations, email addresses, and other standard information.
- Custom Entities: Business logic customizations, on the other hand, are unique to a chatbot depending on the nature of the bot and the needs of the business. It can be sectoral data,

the specifics of transactions, a user's preferences, or any other data relevant to the purpose of the chatbot.

4.1.3 Named entity recognition (NER)

It refers to the component of information extraction that aims to identify and categorize the named entity within unstructured text [30].

4.1.3 Utterance

An utterance can be defined as any single communication that the user or the chatbot gives during a conversation. For instance, a user may say 'hello' or 'I want to pay my bill'. Such statements are some of the examples that are part of the conversation and assist the chatbot in recognizing the user's goal.

4.1.4 Exchange

An exchange is between two or more turn-taking units, including the user and the chatbot. It means the whole dialogical communication exchange. For instance, when a user says, "What's the weather?" and the bot replies, "It is sunny today," such an interaction is an example of an exchange.

4.1.5 Contact

Contact is the interaction of a particular user with an operator or a chatbot. This takes place when the user starts engaging, for instance, by starting a conversation, usually by opening a chat box. If the user has not reacted to the greeting of the bot, no interaction has taken place; however, a connection has been made. Though made only once, multiple exchanges can take place within a contact.

4.1.6 Domain

A domain is the general field or subject area of concern within which a chatbot can work. For instance, customer support chatbot has interfaces and interactions from fields like billing, technical support, accounts, etc. Small talk chatbots, like ChatGPT, can discuss any topic.

4.1.7 Topic

Topic anchored on a certain area of focus usually deals with a certain subject or many tasks in a given field. For instance, in the domain of customer support, subjects may include billing, store opening and closing hours, or returns. Every topic corresponds to specific user intents, which the chatbot must comprehend and process.

4.1.8 Escalation

Handing over the conversation from the chatbot to an actual person is known as escalation. It can happen as an initial escalation (for instance, for multifaceted queries) or when the chatbot fails to understand the client's purpose.

4.1.9 Step

A step is each turn of the conversion, including a single input from the user and the resulting single output from the chatbot. Steps might consist of greetings, confirmation/clarification questions, actions, or even courtesy questions. For instance, a question such as "How can I help you today?" and the user's reply to the question are the two steps.

4.1.10 Flow Logic

The flow is the rule according to which the chatbot behaves in response to each statement and determines the next action. This can range from easy if-else, decision tree, more complex algorithms, or even a probabilistic logic based on machine learning to guide the conversation to be fluid and productive.

4.1.11 Preferred Response

A liked response is a turn that the user and the other participant find satisfactory and progresses the conversation toward addressing the user's goal. For instance, the bot using "can I take your account number?" when the user is in a section to check the balance is preferred.

4.1.12 Non-Preferred Response

Imagining an undesirable turn in the conversation, a non-preferred response is an interaction that does not lead to the conversation's desired outcome: resolving the restaurant's intended course of action. For instance, if the bot replies, "I didn't understand that," without additional instruction, it is an example of a non-preferred reply.

4.2 Advanced Technologies in Chatbot Development

The establishment of a professional and highly effective chatbot requires the use of Natural Language Processing, machine learning algorithms, and the neural network perspective. In this section, the foundation elements and technology that constitute the basis of chatbot creation will be discussed in detail.

4.2.1 Natural Language Processing (NLP)

NLP is a subcategory of artificial intelligence that deals with the interaction between computers and humans via natural language. Basically, it involves the ability of computer software to understand human language, whether spoken or written [20]. Such main tasks in NLP include text analysis, translation, sentiment analysis, speech recognition, etc.

4.2.2 Natural Language Understanding (NLU)

It is the subset of NLP that comprises machine reading comprehension, understanding what the text means, and the context in which it is applied. This functionality allows chatbots to understand what a user meant by a specific input, identify entities such as names, dates, or products, and identify stylistic variations in language, like synonyms and ambiguous phases [53].

4.2.3 Natural Language Generation (NLG)

The subset of NLP dealing with generating human natural language text by a computer is NLG. Essentially, it is the conversion of structured data into human-readable text. Applications include report generation, formulating conversational responses in chatbots, summarizing information, and several other human-using applications [54].

4.2.4 N-Grams

N-Grams are a fundamental concept in Natural Language Processing (NLP), representing contiguous sequences of 'n' items (usually words) from a given text sample. There are different types of N-Grams, including unigrams (single words), bigrams (two-word sequences), and trigrams (three-word sequences), among others. The main idea behind N-Grams is that they capture the contextual relationship between words based on their order of appearance. For text generation, an N-Gram model analyzes a large corpus of text to calculate the probabilities of words following given sequences.

For example, a bigram model calculates the likelihood of a word appearing after another word, while a trigram model considers the probabilities of a word following two preceding words. By using these probabilities, the model can generate new text by starting with an initial word or sequence and iteratively selecting the next word based on the highest probability, thereby creating coherent and contextually relevant sentences. This technique, although simple, forms the basis for more sophisticated text generation models and applications [56].

4.2.5 Recurrent Neural Networks (RNN)

RNN stands for recurrent neural network, which is a specific type of artificial neural network utilized for the analysis of sequential or time series data, which implies its application in such

operations as language translating, NLP, speech identification, and generation of image descriptions. RNNs are the components of Siri, voice search, and Google Translate. Feedforward and Convolutional Neural Networks (CNN) do not have a memory of inputs; rather, they work on inputs in a blind fashion, which denies them the ability to understand context in sequences of data, which is a characteristic of RNNs.

There is parameter sharing, where parameters in an RNN are the same for each layer in the current network; this is different from feedforward networks, where weights are different for each node. These shared weights are adapted using backpropagation through time (BPTT). BPTT entails the accumulation of the errors that occur at various time points so as for the model to make probable adjustments that would make it fit the parameters correctly [26].

One of the challenging issues with RNNs is that the gradients lawfully explode and vanish.

- Exploding gradient: When gradients are too large, this will render the model unstable and result in very large weights.
- Vanishing gradient: when gradients become too small, making the learning process ineffective.

4.2.6 Bidirectional Recurrent Neural Networks (BRNNs)

The bidirectional recurrent neural network (BRNN) processes input sequences at every time step in both the forward and backward directions, providing full context. Dual processing captures information from past and future inputs; this information is very important to be understood in the sequence data. Training of BRNN involves updating weights for both the forward and backward RNNs using techniques like backpropagation through time (BPTT) [27].

4.2.7 Long Short-Term Memory (LSTM) Networks

LSTM is a advanced version of RNN, designed to capture long-term dependencies and this architecture is also able to solve problem associated with RNN like the vanishing gradient problem. On this basis, they showcase very good potential in NLP and chatbot development, where the context is to be held for long sequences. For instance, during a conversation, the beginning statements may greatly affect how accurate later responses are. LSTMs are particularly good at enabling chatbots to maintain coherent and relevant conversations throughout a prolonged dialog.

LSTM have memory cells controlling the flow of information, regulated between them through input, output, and forget gates. It's what makes LSTMs capable of learning to memorize information regarding when data should be stored or discarded. The gating mechanism enables an LSTM to memorize useful information over long sequences and apply this kind of memory for tasks requiring long-term context. LSTM can be computationally expensive, but these models are quite robust for sequence modeling [26].

4.2.8 Gated Recurrent Units (GRU)

Gated Recurrent Units are a relieved variant of Long Short-Term Memory networks, designed to discover long-term dependencies in sequence data while making computations easier. For example, in any dialog system, the meaning of an answer can be very dependent on the previous statements. The GRUs will be able to handle this context very effectively, enabling chatbots to produce extremely coherent and contextually appropriate responses.

While LSTMs have an individual self-updating gate and a reset gate, combining the input and forget gates makes GRUs really simplify the LSTM architecture. Though they are capable of performing as well as LSTMs in many cases, they are computationally much more efficient. The update gate controls how much information flows through, and the reset gate decides what to forget from the past. Such a design makes GRUs very effective in modeling dependencies existing in the data. Due to their simplicity, GRUs still handle long-term dependencies pretty well, so they can fit perfectly in tasks related to text generation, speech recognition, or any other application using fast sequence modeling [26].

4.2.9 Transformers

Transformers are a powerful type of neural network architecture. Unlike traditional RNNs and their variants (like LSTMs and GRUs), transformers do not rely on sequential data processing; instead, they use parallel data processing. They use self-attention, which weighs the relative importance of different words in a sentence irrespective of their position and hence captures context more informatively.

There are encoder and decoder layers in transformers. The encoder processes the input sequence to form a set of attention scores that reflect the importance of each word in the whole sequence. Transformers dominate tasks in language translation, text summarization, and conversational agents because the decoder uses these later to create output sequences. For example, in a chatbot application, it understands the context of what a user asks for—considering all the words together in the sentence, all at the same time—to respond more correctly and in context.

Training transformers requires a large amount of data and high computational resources; however, their parallelizable architecture makes them more efficient compared to sequential models like RNNs. Considering handling efficiency and contextual understanding, transformer models are at the core of many state-of-the-art techniques, including BERT, GPT, and T5, and underpin many research efforts that continue to progress NLP and smart chatbots [28].

4.2.10 Large Language Models (LLMs)

Large language models (LLMs) are one of the biggest AI breakthroughs in NLP and chatbot development in recent years. These include giant models like GPT-3 (Generative Pre-trained Transformer 3) set characterized by their size and pre-trained on extensive textual data, LLMs are really good at understanding human-like text and, therefore, are able to do most tasks attributed to a language: converse, translate, summarize, and answer questions.

LLMs employ the Transformer architecture, which is designed to deal with long-range dependencies in text based on self-attention mechanisms. They learn contextual relationships between words and sentences from massive datasets, thereby enabling coherent responses appropriate to context. For instance, it may understand complex questions, maintain dialog coherence, and only provide accurate information when the learned patterns from the training data optimize its task execution. While these LLMs are computationally demanding during training and deployment, they have remarkably improved the power of chatbots and, by extension, NLP systems [29].

Different neural networks that are described above are most commonly used models for chatbot development. There are multiple concerns related to robustness of these neural networks. Robustness of neural networks, different methods for assessing model robustness and how to maintain robustness during development life cycle of model are discussed in ISO/IEC TR 24029-1:2021, Artificial Intelligence (AI) — Assessment of the robustness of neural networks, Part 1: Overview and ISO/IEC 24029-2:2023, Artificial intelligence (AI) — Assessment of the robustness of neural networks, Part 2: Methodology for the use of formal methods. These standards are applicable on these models and can be used to ensure that models used are robust.

4.3 Chatbot stakeholders

Chatbots can involve several stakeholder roles and sub-roles in various stages of the Chatbot development life cycle. The name of the stakeholder is also indicative of its role or sub-role as a provider, as described in IS/ISO/IEC 22989:2022, 5.19 [1]. These AI stakeholders play crucial roles in the development and deployment of chatbots.

4.3.1 AI provider

An AI provider is an organization or entity that provides products or services that use one or more AI systems. AI providers encompass AI platform providers and AI product or service providers.

4.3.1.1 AI platform provider

An AI platform provider is an organization or entity that provides services that enable other stakeholders to produce AI services and products.

For chatbots, the AI platform provider provides computational resources, storage, different development tools like SDKs, APIs, libraries, pre-trained models that can be further fine-tuned for specific application domains, deployment support for multiple channels, features like intent recognition, providing scalability and reliability for different loads, and tools for monitoring and analyzing chatbot performance.

4.3.1.2 AI product or service provider

An AI product or service provider is an organization or entity that provides AI services or products, either directly usable by an AI customer or user or to be integrated into a system using AI along with non-AI components.

For chatbots, AI products or service providers provide specialized AI solutions and products that can be integrated into existing products to enhance their functionality. Examples of integration services are customer relationship management (CRM), databases, or any other enterprise software. Examples of AI products are sentiment analysis tools, language translation services, voice recognition modules, and more.

4.3.2 AI producer

An AI producer is an organization or entity that designs, develops, tests, and deploys products or services that use one or more AI systems. AI developers are concerned with the development of AI applications. AI developers are responsible for model design, model implementation, and further model verification, validation, and testing.

For chatbots, AI developers are responsible for choosing the right kind of technology, model, and platform for deployment, along with building chatbots while meeting certain technical and non-technical requirements and documenting every process.

4.3.3 AI customer

An AI customer is an organization or entity that uses an AI product or service, either directly or through its provision to AI users. For chatbots, AI customers refer to chatbot users.

4.3.4 AI partner

An AI partner is an organization or entity that provides services in the context of AI. AI partners can perform technical development of AI products or services, conduct testing and validation of AI products and services, audit AI usage, evaluate AI products or services, and perform other tasks.

4.3.4.1 AI system integrator

An AI system integrator is an organization or entity that is concerned with the integration of AI components into larger systems, potentially also including non-AI components.

For chatbots, an AI system integrator is an organization or entity that integrates chatbots into existing products and services.

4.3.4.2 Data provider

A data provider is an organization or entity that is concerned with providing data used by AI products or services.

For chatbots, a data provider is responsible for collecting data for training, testing, and validation. Data providers should ensure that data comes from reliable sources, is consistent, and avoids bias, misinformation, or misleading data representation.

4.3.4.3 AI auditor

An AI auditor is an organization or entity that is concerned with the audit of organizations producing, providing, or using AI systems to assess conformance to standards, policies, or legal requirements.

For chatbots, an AI auditor is responsible for verifying that chatbots handle user data in compliance with data protection regulations and organization policies. Assess chatbot algorithms and data to detect and mitigate biases, identify security and operational risks, and mitigate potential risks.

4.3.4.4 AI evaluator

An AI evaluator is an organization or entity that evaluates the performance of one or more AI systems.

For chatbots, AI evaluators assess the chatbot's ability to understand user queries accurately, identify common errors and misunderstandings, measure key performance indicators (KPIs), and document observations.

4.3.5 AI subject

An AI subject is an organization or entity that is impacted by an AI system, service, or product.

4.3.5.1 Data subject

A data subject in AI refers to an organization or individual whose data is used to train AI systems. This raises privacy and security concerns, especially when personal information of individuals is involved, necessitating careful handling to protect privacy rights and data security.

4.3.5.2 Other subjects

Other organizations or entities impacted by an AI system, service, or product can, for example, be individuals or communities.

4.3.6 Relevant authorities

Relevant authorities are organizations or entities that can have an impact on an AI system, service, or product.

4.3.6.1 Policy makers

These are organizations and entities that have the authority to set policies within an international, regional, national, or industrial domain that can have an impact on an AI system, service, or product.

4.3.6.2 Regulators

These are organizations and entities that have the authority to set, implement, and enforce the legal requirements as intended in policies set forth by policymakers (4.3.6.1).

4.4 AI application functional characteristics applicable to Chatbots

Chatbots are considered as an AI application. There are some functional characteristics that distinguish AI applications from non-AI applications. These characteristics are described in ISO/IEC 5339:2024, 5.4 [2]. These characteristics are explained below, and how are they applicable to chatbots:

- An AI application is built with the capabilities of an AI system that implements a model to acquire information and processes with or without human intervention by algorithm or programming.
  Chatbots use AI models powered by NLP to understand and interpret user input. They process this information using algorithms and programming logic to generate appropriate responses. Chatbots operate autonomously once deployed without human intervention.
- An AI application applies optimizations or inferences made with the model to augment decisions, predictions, or recommendations in a timely manner to meet specific objectives.
  Chatbots continuously analyze incoming data to optimize responses and make inferences about user intent and preferences while ensuring timely interaction with users.
- In some cases, an evaluation of interaction outcomes results in an update to an AI application, improving the model, system, or application.
  There is a continuous evaluation, feedback, and improvement cycle that enables chatbots to improve, effectively meet user needs, and deliver more accurate responses.

4.5 AI application non-functional characteristics and considerations applicable to Chatbots

The non-functional characteristics of an AI application should also be considered by the stakeholders when making decisions about the AI application. This clause introduces AI application-specific non-functional requirements. These characteristics are described in ISO/IEC 5339:2024, 5.5 clause [2].

4.5.1 Trustworthiness

To a chatbot, trustworthiness simply means that the system becomes reliable to meet all expectations set by users and interested parties. A trustworthy chatbot is dependable, reliable, predictable, or operates without failure in the attainment of intended functions.

4.5.1.1 AI Robustness

The robustness of AI in chatbots refers to their 'invariance'—in particular, the degree by which they are able to hold on to their performance under different conditions.

A robust chatbot will keep working correctly at unexpected inputs, user mistakes, and other unwise events, and thus the delivered service will become consistent.

4.5.1.2 AI Reliability

AI Reliability for chatbots refers to the entity's ability to perform required functions accurately under specified conditions over time.

In other words, a reliable chatbot would, therefore, be the one offering accurate responses all the time and handling interactions without frequent downtimes or failures.

4.5.1.3 AI Resilience

AI resiliency in chatbots details the ability of a bot to recover from faults or disruptions very quickly.

A resilient chatbot would quickly resume normal operation after experiencing some problems, reducing downtime and preserving users' trust.

4.5.1.4 AI Controllability

Basically, controllability means the possibility for AI in chatbots to provide an external agent with the means to interfere and change its operation in case of troubleshooting, updating of capabilities, or even bringing down the system in the event of serious mistakes.

4.5.1.5 Explainability of AI

Now, the explainability of AI in chatbots is that part of the system that gives reasons for giving such responses. As a result, the users will know why the chatbot has given such an answer, thereby increasing transparency and trust.

4.5.1.6: Predictability of AI

AI predictability in chatbots refers to the fact that behavior and output are invariant with respect to variability and are thus predictable by the user and other stakeholders. Predictable chatbots behave according to expectations, hence increasing user experience and building trust.

4.5.1.7 AI Transparency

AI transparency for chatbots refers to the provision of information about chatbot purpose and development to stakeholders. Transparent chatbots refer to those able to convey to user insight into their functioning and user data management in order to build confidence and trust.

4.5.1.8 AI Verification and Validation

Chatbot AI verification checks whether the chatbot is developed correctly and that it meets all the requirements as specified by the system. Conversely, AI validation makes sure that the chatbot does its job in an effective way. These two procedures are crucial and necessary for keeping the chatbot at the highest level of quality and performance.

4.5.1.9 AI Bias and Fairness

Bias and fairness in chatbots are features that ensure all activities concerning the user are impartial and just. Unfair behavior and biases should be avoided because they have a bad impact on individuals and groups. Make sure this bot deals with all users equally.

4.6 Guidance of AI Applications

The guidance for AI applications is formulated as a set of questions that each type of stakeholder should ask, and these questions can lead to opportunities for stakeholders to further investigate the applicability of relevant international standards for specific applications. These sets of questions for guidance can be found in clause 7 of ISO/IEC/5339: Information Technology—Artificial Intelligence —Guidance for AI Applications [2]. These guidelines can also be applicable to chatbot stakeholders.

5. Chatbot Development Lifecycle

The following are the steps of a chatbot development lifecycle that should be followed [13], [31], [37], [38]. This is for any team that is looking to build a new bot or add a skill to an existing bot. Each phase of the lifecycle should be well documented.

5.1 Chatbot Requirements

This phase maps to the "Inception" phase in the AI life cycle (Sections 6.4.1 to 6.4.3) outlined in IS/ISO/IEC 5338:2023 Information Technology- Artificial Intelligence- AI System Life Cycle Processes, where the business or mission analysis, stakeholder needs, and system requirements are defined.

During the requirement elicitation phase, it is essential to pre-determine and thoroughly document various types of requirements for a chatbot. These include system requirements, stakeholder requirements, compliance and legal requirements and ethical considerations. These documented requirements serve as the foundation for validating and verifying the chatbot both during testing and post-deployment, ensuring that it adheres to the specified criteria.

5.1.1 System Requirements

System requirements refer to the technical specifications and capabilities that the chatbot and its underlying infrastructure need to meet in order to function effectively [32], [33]. These include:

- Channels: Determine which communication channels your chatbot will support, such as websites, WhatsApp, Facebook, SMS, Instagram, email, etc.
- Languages: Specify the languages in which the chatbot should communicate—whether it's a single language or multilingual capabilities are required.
- Integrations: Identify the systems and tools the chatbot needs to integrate with, such as CRM, payment systems, calendars, maps, and custom internal tools.
- Technologies: Indicate any preferred technologies or frameworks for building the chatbot, including considerations for Natural Language Processing (NLP), Machine Learning (ML), or Artificial Intelligence (AI) capabilities.
- Accessibility: Address any accessibility requirements the chatbot must meet, such as compliance with WCAG or ADA standards.
- Security: Specify security measures and standards that the chatbot and its vendor must adhere to, ensuring the protection of user data and system integrity.
- Hosting: Decide whether the chatbot and user data will be hosted on your own servers or on the cloud. If cloud-based, specify the preferred service provider and server location considerations.

5.1.2 Stakeholder Requirements:

Stakeholder requirements refer to the expectations and needs of the individuals or groups who have a vested interest in the chatbot project. These requirements typically focus on usability, user experience, and business objectives [32], [33]. They include:

- Chatbot's Look and Tone of Voice: Define the visual appearance and personality traits (tone of voice) of the chatbot to align with your brand identity and user expectations. Chatbot's personality impacts most elements of conversation design. It should reflect your brand, be appropriate for its intended users, and function as follows: A fitness assistant bot should use active language; a healthcare diagnosis app should avoid jokes.
- KPIs and Metrics: Outline specific Key Performance Indicators (KPIs) and metrics that the chatbot should track and achieve, such as user engagement, automation rate, etc.
- Analytics and Dashboards: Specify requirements for real-time analytics and dashboard features, including data points like user statistics and performance metrics.
- User Base: Estimate the number of users within your team and customer base who will interact with the chatbot regularly.
- Rich Media: Determine if the chatbot's responses should include text, hyperlinks, images, gifs, videos, and PDF attachments to enhance user interaction.

These requirements ensure that the chatbot meets the needs of both technical functionality (system requirements) and user expectations (stakeholder requirements), thereby contributing to a successful implementation aligned with organizational goals.

5.1.3 Compliance and Legal Requirements:

Depending on industry and geographic location, there may be specific legal or regulatory requirements that the chatbot must comply with, such as data protection laws (GDPR, HIPAA), industry standards (Payment Card Industry Data Security Standard, PCI-DSS for payment processing), or accessibility guidelines (WCAG) [32], [34].

5.1.4 Ethical Considerations

Ethical principles for AI guide the development and use of chatbots toward their execution in a responsible way. For instance, chatbots shall be designed to foster social good and positively serve intended beneficial functions and stay away from bad impacts.

Such ethical principles have to be applied both with regard to the chatbot provider/developer and with regard to the intentions behind a chatbot system's use. Making chatbots available for customers and other stakeholders has to include considerations that minimize the possibility of misuse, abuse, and disuse of information. There are some ethical considerations that should be taken into account for development of AI systems and applications. These considerations are listed in ISO/IEC 24368 Overview of Societal and Ethical Concerns, in clause 6 [3], few of these considerations that are relevant to chatbot are listed below:

a) **Accountability**: Accountability in the context of chatbots would then imply that the organization accepts relevant scrutiny and acknowledges the requirement to respond. It means being responsible for chatbot decisions and recognizes that an organization is not relieved from responsibility for incorrect decisions because they were based on AI or machine learning outputs.

b) Fairness and non-discrimination:

The fairness and non-discrimination objective is to make sure that chatbots work for all people within the different social groups, more so for people who have been left behind or undervalued in the local, national, and international scenes, either socially, politically, or economically. This includes:

- Reducing undesired bias against members from different groups
- Making sure that the collection and application of training data and user data truly represent the diversity of the members of the different groups.
- Minimize the impact of human cognitive or social biases during data collection, processing, system design, model training, and at all other stages of development.

c) Transparency and explainability

The transparency implies that, if the stakeholders of a chatbot engage with it, all of its decisions should be very transparent and understandable. Traceability information and disclosure on algorithms, training, and user data and how these have been collected. The methods and metrics followed to validate how the chatbot works.

d) Privacy

Chatbot privacy ensures that the system is designed and deployed with due regard toward protection of the individuals' right to privacy. The concern for privacy remains at the forefront of AI developments, mainly because of laws such as the GDPR in the European Union. Organizations must make use of mechanisms to apply and showcase adherence to the legal provisions regulating privacy and data protection. Common dimensions of privacy for chatbots predominantly include:

- Limiting data sourced, collected, used, or disclosed to what is relevant and needed to perform the defined purposes and tasks.
- Stating the purposes of processing personally identifiable information (PII) and sharing, as applicable.
- Making known to the individual what data is held on them and ensuring no data is collected or used without knowledge or permission.
- People should have the right to prohibit or restrict processing, thereby blocking data collection or use that has resulted from chatbot technology in any way.

e) Safety and security

Safety and Security can also mean “loss prevention”. It means AI systems and applications doing what they are designed to do and without compromising on key aspects, such as reliability or performing as per expectations. Testing and monitoring should be done with rigor to reduce the risks of misuse and abuse, unexpected circumstances, or opponent attacks. AI systems shall do what one would expect of them in every case, including continuous learning systems.

f) Level of automation and control

The degree of automation indicates the degree of independence of an AI from human supervision and control. It thus determines how much information an AI system provides to the operator about its behavior and to what degree and in what form human control and interference are possible. Highly automated systems also give rise to dangers in the performance characteristics of reliability and safety.

As a result, a chatbot's level of autonomy can range from having:

- zero autonomy—in which case the operator has complete control over the system.
- full autonomy, in which case the system is free to change its operating domain or objectives without any outside input or influence.
- assistance level autonomy, the system assists the human operator,
- Partial automation has some individual sub-functions that are able to operate on their own but are still under the control of a human.
- Conditional automation would then be when part of the system can execute specific tasks autonomously, although there would still be a need for standby external agents to take over when the situation calls for it.
- High automation would therefore imply that portions of its mission could be executed without intervention
- under full automation, the system can execute the entire mission on its own. Autonomous systems go ahead and alter their operations and even goals on their own.

Different levels of automation come with different potential risks. Systems at an assistance level pose a risk of operator over-reliance, reducing vigilance. Partial and conditional automation can result in miscommunication and delayed human intervention. Systemic failures are likely to be a result of high and complete automation that will reduce human oversight. Autonomous systems are related to the risks of losing human control, unpredictable behavior, and ethical and legal implications.

Technical failures, security vulnerabilities, issues related to ethics and the law, loss of confidence on the part of the users, and scalability challenges are common risks at all

levels. Therefore, redundancy, regular updating, human oversight, transparency, and ethical guidelines minimize such risks.

Level of autonomy and controllability of automated artificial intelligence systems is discussed in ISO/IEC TS 8200:2024, Information technology — Artificial intelligence — Controllability of automated artificial intelligence systems and practices and considerations for building and using ethical and socially acceptable AI systems and applications are discussed in ISO/IEC TR 24368:2022, Information technology — Artificial intelligence — Overview of ethical and societal concerns [3]. Both of these standards are applicable for chatbot development.

Apart from this, according to National institute of standards and technology (NIST), Artificial Intelligence Risk Management Framework (AI RMF 1.0) [59] following are the characteristics of trustworthy AI systems include: valid and reliable, safe, secure and resilient, accountable and transparent, explainable and interpretable, privacy-enhanced, and fair with harmful bias managed, which are also mentioned in in ISO/IEC TR 24368:2022, Information technology — Artificial intelligence — Overview of ethical and societal concerns [3]. These are a few considerations that should be taken care of while developing chatbots.

5.2 Chatbot Design

After defining requirements for chatbots, the next step is to design a workflow for the development of chatbots. During the design phase, the overall architecture of the chatbot is defined. It also includes software and hardware considerations like where components will be sourced, whether it will use external platforms or services, whether it will be built from scratch, or whether it will use open source software.

This phase corresponds to the "Design and Development" phase (Sections 6.4.4 to 6.4.5) outlined in IS/ISO/IEC 5338:2023 Information Technology- Artificial Intelligence- AI System Life Cycle Processes, which involves defining the system architecture and design specific to the AI system.

This phase also determines the testing methods and techniques that will be used later on during validation and verification and to determine whether the chatbot is ready for deployment. All these design phases should be fully documented. This will help developers or testers track down any risk or threat that chatbot may face after deployment. It will provide traceability and risk identification, easier maintenance, and updates.

5.2.1 Research and design

Initial research and planning for designing the chatbot include market research, user stories and use cases, and platform selection. These points are discussed below:

- **Market Research:** Before starting the development of the chat bot, we need to analyze existing chat bots in a particular domain or industry, and we need to analyze their

weaknesses and strengths. Market research will help us get user feedback and reviews regarding their own experience with chatbots. Market research also identifies the target audience's needs and preferences. Furthermore, market research also provides benchmark metrics and performance analysis of successful chatbot implementations [33].

- **User stories and use cases:** User stories and use cases are both ways of capturing users' needs and goals. User stories are simple, concise statements that describe what the user wants and why. There is a simple template for writing user story cards: “As a <user>, I want <goal/desire> so that <reason/benefit>”. Use cases are a more detailed description, and they also describe how users interact with the system and the responses generated by the system. It typically describes the series of steps that the user takes to accomplish a specific goal, along with the system's responses at each step. It is used to specify requirements and document the expected behavior of the system [35].
- **Platform selection:** Choosing the right platform for deploying chatbots is critical to ensuring that we are able to reach a target audience. Different platform options could be mobile apps, websites, social media platforms, and messaging platforms like WhatsApp and Facebook [33].

5.2.2 Design Conversational Flows

Conversational flow refers to the interaction between the chatbot and the user. It encompasses the design and structure of conversation, user intent, and response generated by the chatbot [36]. Following are the benefits of a conversational flow diagram:

- Creates a human-like conversation, making it easier for the user to engage with the bot and have meaningful conversations.
- It provides better customer service, reduces the need for human agents to handle simple inquiries, and frees up resources to handle more complex queries.
- Increase engagement and improve conversational rate.

Designing conversational flow is done at the early stages before complex stages of developing our chatbot or voice assistant. Key elements of conversational flow are user intent, context awareness, natural language processing, and response generation.

5.3 Data for training models

This phase maps to "Design and Development" (Sections 6.4.7 to 6.4.8) outlined in IS/ISO/IEC 5338:2023 Information Technology- Artificial Intelligence- AI System Life Cycle Processes, particularly focusing on knowledge acquisition and AI data engineering necessary for training the model.

For the development of chatbots, data is crucial. Data is needed for model training, testing, and validation.

- Data should be domain-specific and represent real-world user questions, queries, and interactions.
- Data should be free from any kind of bias,
- anonymized, and in compliance with privacy regulations.
- All information related to data should be well documented; it may include source of data, any updates to data, how data is processed and stored, and how data is disposed of after it is no longer relevant.

All the steps involved in data collection and preprocessing should be well documented. This documentation will help chatbot developers track problems related to bias, data drift, or concept drift that may affect model performance. Also, in the case of preprocessing of data, all information related to steps of data preprocessing should be modeled, including their model versions and hyperparameters like weights and learning rate. This documentation will ensure that data is reproducible, maintain transparency, maintain model performance, and be free from any bias or data drift.

Data life cycle framework and its different stages mentioned in ISO/IEC 8183:2023, Information technology — Artificial intelligence — Data life cycle (DLC) framework is also applicable for data used for chatbot training. Data life cycle framework and its different stages are shown below in Fig.1 Data life cycle framework.

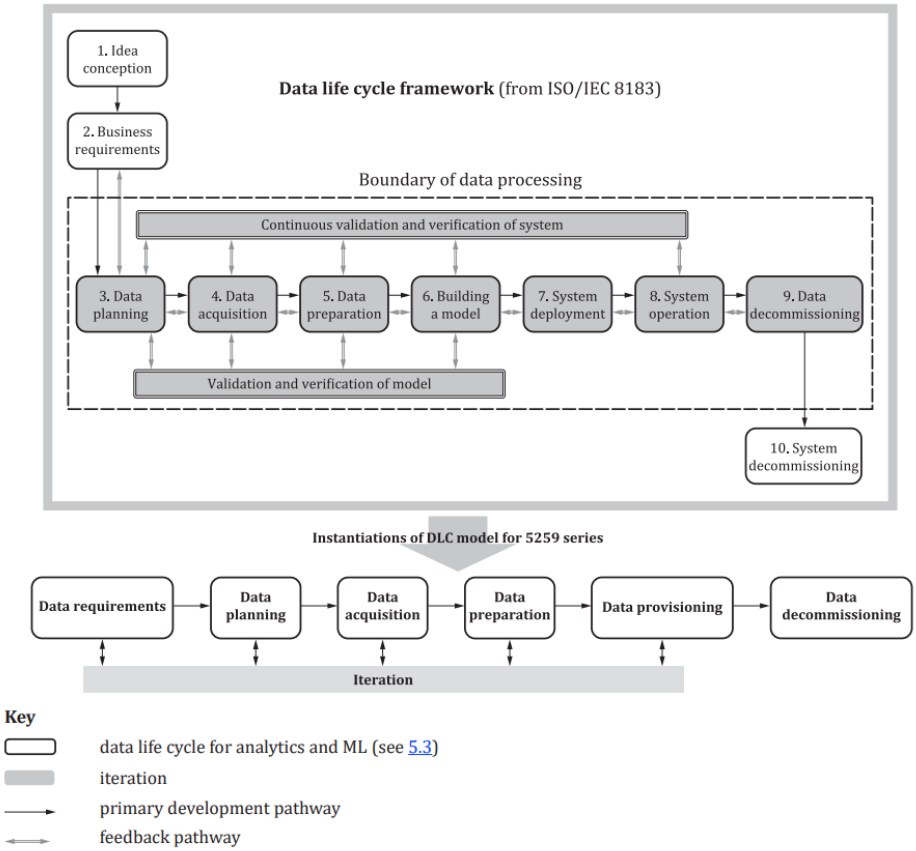

Fig.1 Data life cycle framework

Apart from this ISO/IEC 8183, there is a series of standard that is also applicable on data for training models and ensures that data quality (DQ) is appropriate for ML models:

i. **ISO/IEC FDIS 5259-1, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 1: Overview, terminology, and examples.**
   This standard provides a foundation for conceptual understanding of data quality for analytics and machine learning. Data quality concepts include data quality management, data quality governance and data provenance.
   DQ management includes a DQ model, DQ measures and DQ assessment. DQ model is a defined set of data quality characteristics which provides a framework for specifying data quality requirements and evaluating data quality. DQ measures that are used to evaluate data quality. DQ assessment uses the results of selected data quality measures to assess whether a dataset meets its needs and requirements. Lastly any improvement in data and reporting of that improvement.
   Data quality governance means data quality processes should conform to the organization's data governance policies. Data provenance records can be used to gather and maintain data provenance information which can provide a basis for determining whether the data have

been intentionally manipulated or altered. This standard is applicable on chatbot training data. Figure below describes processes that should applied across the multiple stages of the DLC model for analytics and ML [6].

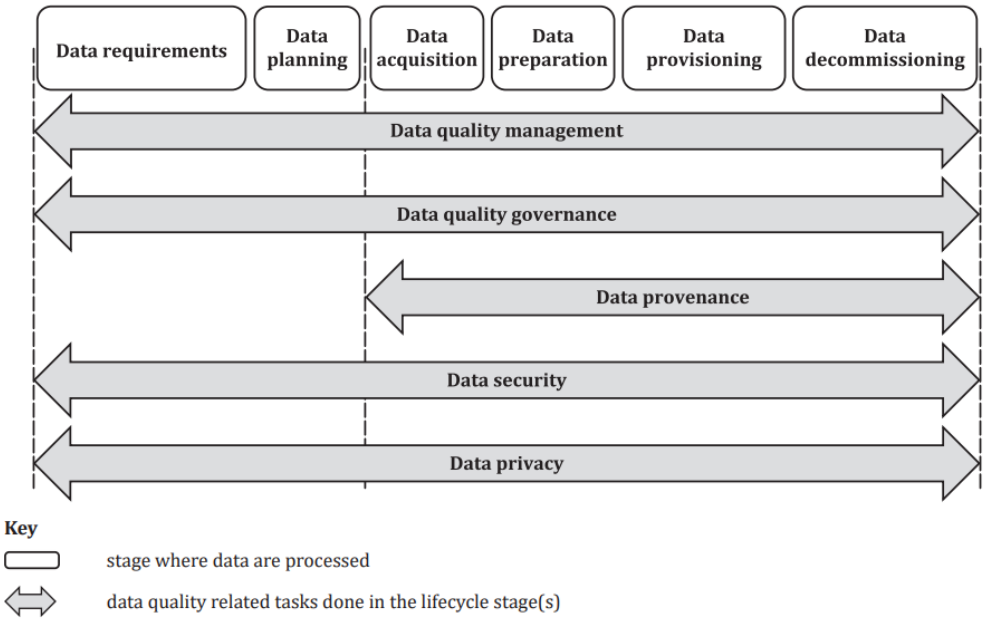

Fig.2 Processes across the multiple stages

ii. ISO/IEC DIS 5259-2, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 2: Data quality measures.

This standard describes all data quality characteristics. These characteristics involve inherent data quality characteristics, inherent and system-dependent data quality characteristics, system-dependent data quality characteristics and additional data quality characteristics.

Inherent data quality characteristics: Among mentioned characteristics in standard accuracy, completeness, consistency, currentness and credibility are applicable on data used for chatbots.

Inherent and system-dependent data quality characteristics: Among mentioned characteristics in standard compliance, efficiency, precision and understandability are applicable on data used for chatbots. While accessibility to data can cause risk and serious threats to chatbot.

System-dependent data quality characteristics: Among mentioned characteristics in standard portability is applicable on data used for chatbots, which means ability to move data from one system to another, within a specified context, while preserving its quality.

Additional data quality characteristics: Among mentioned characteristics auditability, identifiability, effectiveness, balance, diversity, relevance, representativeness are also

applicable on data used for chatbots. While similarity is only applicable when intents are labeled to data and timelessness is applicable if data is taken from public datasets that are not relevant [7].

iii. **ISO/IEC FDIS 5259-3, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 3: Data quality management requirements and guidelines.**

This document specifies requirements and provides guidance for establishing, implementing, maintaining and continually improving the quality of data used in the areas of analytics and machine learning. As data quality requirements vary with context and application domain, this document provides a generic set of requirements and recommendations relating to common data life cycle stages [8].

iv. **ISO/IEC FDIS 5259-4, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 4: Data quality process framework**.

This document is applicable to training and evaluation data that come from different sources, including data acquisition and data composition, data preparation, data labeling, evaluation and data use. Data quality process framework include DQ planning, DQ evaluation, DQ improvement and DQ process validation.

Apart from this, the document also includes data labeling methods and processes and how to ensure quality during the labeling process. It also includes the DQ process for supervised, semi-supervised, unsupervised and reinforcement learning [9].

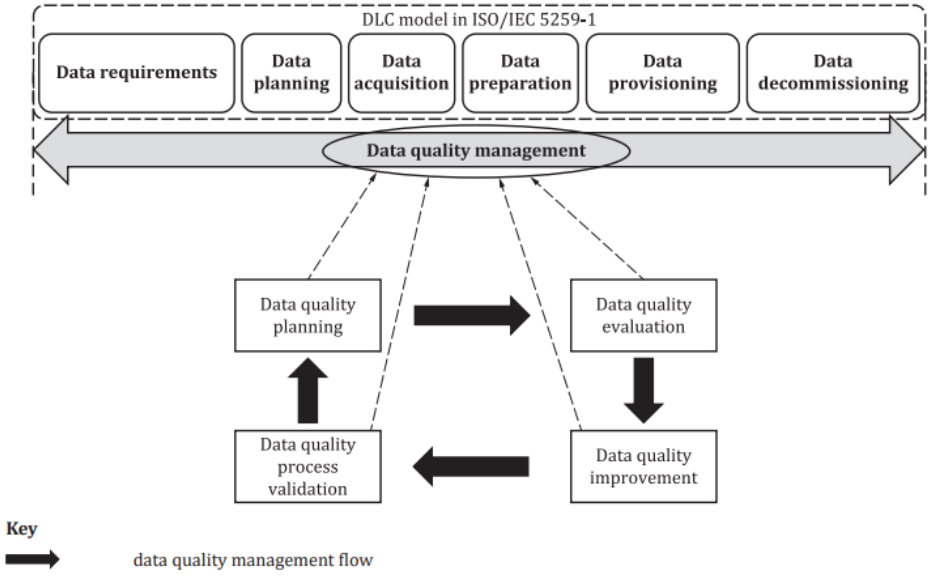

Fig.3 The relationship between the DLC model and the DQPF

v. **ISO/IEC DIS 5259-5, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 5: Data quality governance framework.**
This document provides a data quality governance framework for analytics and machine learning to enable governing bodies of organizations to direct and oversee the implementation and operation of data quality measures, management, and related processes [10].

These standards provide a good insight on how to maintain DQ during DLC. However, there are few extra steps that are needed for data cleaning and preprocessing because majority data is in unstructured form but models need numerical representations of data for training.

5.3.1 Data Acquisition

Data used for chatbot training is generally in the form of text. To get data for a particular domain, there are multiple ways [55]. Some of these ways are mentioned below:

- Conversation Logs: Conversation logs can be used for training, testing, and validation data. These logs are historical chats between users and chatbots. These kinds of logs contain real-time service interactions and show how users interact.
- Public Datasets: While developing a chatbot for a particular domain, there are multiple public data sets that are available on platforms like Kaggle and GitHub that can be used to train models. If particular data does not completely satisfy the requirements of the application, then developers can merge data from different sources.
- Custom data collection: Due to any data quality or legal or compliance concern, if developers are not able to use any public data or conversation log, then they might collect their own custom data using surveys, simulated conversations, or crowdsourcing platforms.

5.3.2 Data Cleaning

Data preprocessing is an important step; data is usually in text form. It is important to preprocess data and ensure that data is in consistent format [38]. Below are the steps that are generally used for text preprocessing:

- converting text to lowercase.
- removing punctuation and URL links.
- correcting misspelled words.
- expanding contractions
- removing emojis and numbers
- Stop-word removal: Stop words are commonly used words like a, an, the, is, what, etc. These words are removed because they don’t add any significant meaning to text.
- Tokenization is the process of splitting text into smaller units, i.e., tokens. Tokens can be words, numbers, symbols, n-grams, or characters.

- Stemming is a rule-based process for converting a token to its root form. For example, change the word ‘increased’ to ‘increas’.
- Lemmatization: Similar to stemming, but properly using grammar and morphological analysis of words to remove inflections from words and return base or dictionary form of the word, also known as lemma. For example, change ‘increased’ to ‘increase’.

After cleaning and formatting data, it is important to label data with corresponding intent. This ensures that the chatbot is able to classify the user's intent. It is recommended to train the model with data that is balanced, having an equal number of examples for each intent that the chatbot should be able to recognize.

Entities are also annotated in the training data. This will help the model to grasp how to extract entities from data and which entity should be extracted from data. Intents and entities are mapped using one to one or one to many relationships, for a given entity there can be multiple intents. Models can be trained using supervised, unsupervised or reinforcement learning. ISO/IEC FDIS 5259-4, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 4: Data quality process framework include data labeling methods and process and how to ensure quality during labeling process. This standard can be used to ensure the quality of training data for supervised learning or reinforcement learning.

If we are not labeling data then, there is another approach that is used to group training data with similar intent. Text vectorization techniques are used to generate numerical vectors and then these numerical vectors can be used to calculate the distance or similarity between data examples, and this will determine how close the data examples are to each other.

5.3.3 Text Vectorization

Text vectorization is a process to transform text into numerical vectors so that machine learning models and neural networks, which can only comprehend numerical representation, can process it. Vectorization techniques like TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings capture semantic meaning and the relationship between the words [39], [40]. If online platforms and services are used for chatbot development, then they often provide tools and APIs to facilitate text preprocessing and vectorization. Word embeddings are numeric representations of words in a lower-dimensional space, capturing semantic and syntactic information. Some of the word embedding techniques are defined below:

- One-hot encoding: the vector representation of one hot encoded vector is represented in the form of 0 and 1, where one indicates the position where the word exists and 0 everywhere else [52].

Labels
Sunny
Rain
Wind

Sunny	Rain	Wind
1	0	0
0	1	0
0	0	1

Fig.4 One-hot encoding

- Bag of Words (BoW): The bag of words creates a vocabulary of unique words in the corpus and counts the occurrence of each word for each document [51].

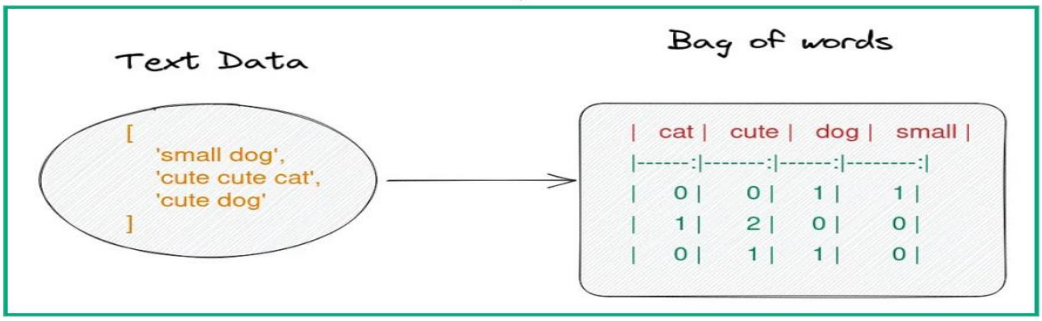

Fig.5 Bag of Words (BoW)

- Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF measures the importance of words depending on how many times they occur in a document or a corpus. To understand this, let's understand term frequency and inverse document frequency. Term frequency (TF) refers to the frequency of a word in a document. It is the ratio of the number of times a word appears in a particular document to the total number of words in that document. Inverse document frequency (IDF) measures the importance of a word in the corpus. It is the logarithmic ratio of the number of total documents to the number of documents in a particular word. TF-IDF is TF (word)* IDF (word) [49].

$$TF = \frac{Number\ of\ times\ a\ word\ "X"\ appears\ in\ a\ Document}{Number\ of\ words\ present\ in\ a\ Document}$$

$$IDF = log\left(\frac{Number\ of\ Documents\ present\ in\ a\ Corpus}{Number\ of\ Documents\ where\ word\ "X"\ has\ appeared}\right)$$

$$TF\ IDF\ =\ TF * IDF$$

Fig.6 TF-IDF

- Word2Vec: Word2Vec is not a singular algorithm; it is a family of model architectures and optimizations that can be used to learn word embeddings from large datasets [41]. Word2Vec aims to capture semantic relationships between words by mapping them to

high-dimensional vectors. The main idea behind this is to ensure that words with similar meanings have similar representations. Word2Vec utilizes two architectures: continuous bag of words (CBOW) and Skip Gram [50].

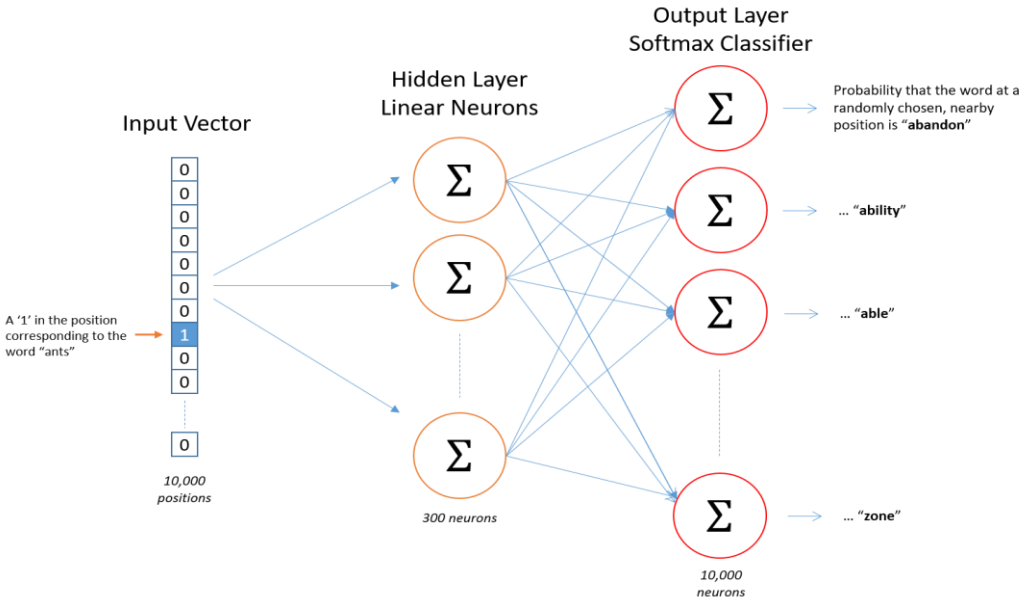

Fig.7 Word2Vec

- PreTrained Word-Embedding: Pre-trained word embeddings are representations of words that are learned from large corpora and are made available for reuse in various natural language processing (NLP) tasks. Some examples of pre-trained word embedding are GloVe, Fasttext, and BERT (Bidirectional Encoder Representations from Transformers).

Using these techniques, text is converted to a numerical vector, which is used to find similarity between data and label data with intent. Now, Data can be used for training models [40].

5.4 Modeling

This phase also falls under the "Design and Development" phase (Sections 6.4.6 to 6.4.10) outlined in IS/ISO/IEC 5338:2023 Information Technology- Artificial Intelligence- AI System Life Cycle Processes, where system analysis, implementation, and integration of the model take place.

There are two main processes for which model is trained, these processes are:

- Entity Extraction: Every chatbot has different sets of entities, and chatbots need to capture these entities for better understanding of context and to improve chatbot assistance for users.
- Intent Classification: Intent Classification is finding the intent of the user from the utterance. Like "hi,", "hello," or any other form of greeting, it will be classified as a greeting.

Intents and entities are a way to find out what users want. For chatbots, models are generally used to classify intent of user utterance, extract entities related to user queries, use dialog management to store relevant information that is used later on for response generation [19], [38].

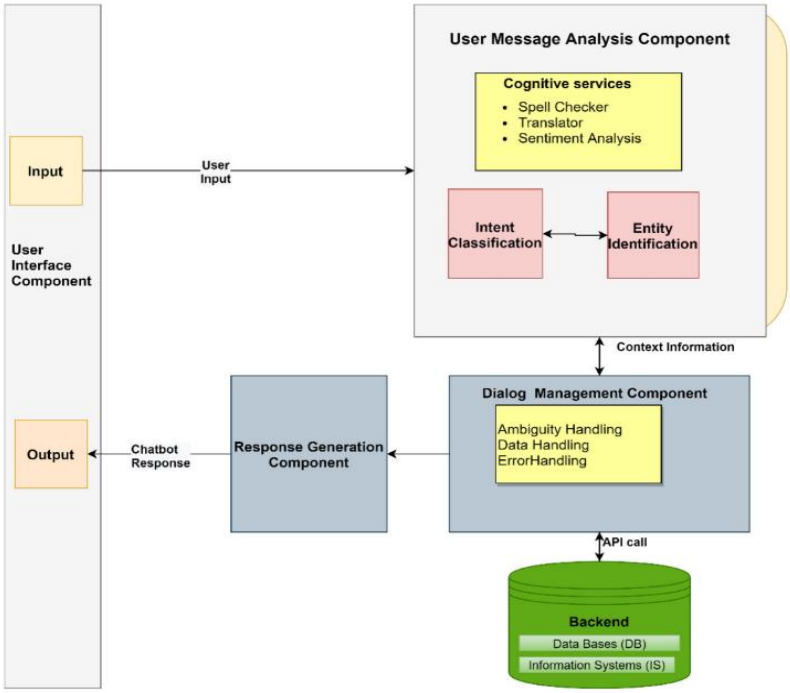

Fig.8 General Chatbot Architecture

Documentation for the training of models in chatbots should include:

- Data collection and annotation procedures followed in getting data used to train the models
- Available feature engineering Details in choosing a specific model architecture to fit in the domain or the problem, with the addition of any modification or adaptation to the model
- Details in selecting hyperparameters of the model and the reason guiding the selection
- Detailed working principles, including applied optimization processes,
- Methods used for interpreting model decisions and ensuring model transparency and explainability.

5.4.1 Intent Classification and Entity Recognition

Now, after labeling our data and completing the preprocessing of the data, the next task is to train a model to classify intent and recognize entities for generating appropriate responses. In intent classification, the model is trying to map user utterances to predefined intents. The order of execution of the overall process is:

- Splitting data into training, testing and validation data
- using tokenizer to generate tokens

- Encoding the target variable or intent
- Generate embedding matrix from input
- Initialize model architecture
- Initializing learning rate, model callbacks and techniques to avoid overfitting
- Fit the model on training data and save the model with updated parameters that perform best for the given data.
- Load the saved model for validation and testing

There are a few considerations that need to be taken care of regarding the embedding layer:

- It is important to choose the right pre-trained embeddings that are appropriate for the domain of the chatbot. For example, while developing a Twitter-based chatbot, it is not recommended to use embedding trained on Wikipedia or any other data that is not similar to Twitter data.
- The pre-trained embedding used should cover all the vocabulary of the data.

There are some considerations that need to be taken into account for model architecture:

- To make sure that output layer dimensionality is same as number of intents,
- learning rate should be slowed down after certain number of iterations and epochs and early stopping should be used to stop training early if a certain threshold is reached, which will prevent overfitting

The second important task is to train a model to extract entities from user input to get relevant information related to intents. The order of execution of the overall process is:

- Create a dictionary that stores entities relevant to chatbots,
- train a named entity recognition model for custom data,
- Save the trained model for later use.

Once the model is able to find the intent of the user and entities related to user input, the chatbot can generate an appropriate response [38].

5.4.2 Dialog Management

Dialog management (DM), one of the key components, needs to handle information coming from other components. It is in charge of controlling and updating the context of conversations and governing all actions regarding the chatbot [17], [18]. In that line, a core aspect of conversation systems development is designing a dialog management component able to support robust, intelligent, and engaging conversation. Some of the different roles played by DM are:

- It consists of keeping track of user input data, either implicitly or explicitly.
- It understands the conversational context and resolves ambiguity, controls conversation flow between users and the system and roles,

- communicates with external services and databases and identifies system actions to fulfill the ultimate user goal.

5.4.2.1 Design of dialog management system

DM is a very critical component of the chatbot architecture, where one makes very conscious, deliberate choices. The conversation strategy definitely determines who holds the guide for the conversation. Some conversations can either be user-directed, system-directed, or mixed-initiative [14].

- If it is a user-directed conversation, then the user takes the initiative, and to every query by the user, the system responds.
- If the dialog is system-directed, the system takes the lead, and the user simply responds to the system query.
- When the dialog is mixed, both the user and system can take control.

Dialog management also includes an error handling module. When coping with uncertainty and ambiguity problems in case of information reception, there are two types of confirmation strategies that can be used to confirm whenever it is necessary.

- In an explicit confirmation strategy, the system can confirm its understanding by asking another question. Below is an example of explicit confirmation:

  User: I want to know hours the closest CVS is open

  System: Do you want to go to the closest CVS?

  User: Yes

- In implicit confirmation strategy, the system embeds some information it receives in the input and asks another question. The following is an example of an implicit confirmation:

  User: I want to know the hours of the closest CVS.

  System: When do you want to arrive at CVS?

They can either be goal- or task-oriented or non-task-oriented.

- Task-oriented systems are used by agents created in order to meet such specific tasks as reservations, answers to frequently asked questions, and schedule meetings.
- Dialog systems that are non-task-oriented concentrate on open-domain conversations with no particular task completion in view, like lightweight, casual messages for chit-chat.

5.4.2.2 DM key terms

In its simplest form, an interaction with a conversational agent involves a user input in natural language or user utterance, followed by the system response, in what is called a conversation turn. In each turn, the conversational system first uses the Natural Language Understanding (NLU) component to convert the user utterance (e.g., "I would like to book a flight from Lyon to Sydney") into a structured representation **U** that is typically encoded using three main types of information: intent, entities, and dialog acts. Dialog acts are hidden actions in user utterances to indicate whether the user is making a statement or asking a question.

All this information is fed into the so-called Dialog State Tracking (DST) model, which is a function that maps a given structured representation **U** into a suitable state, known widely as a dialog state **S**, from the state space of the conversational system. The dialog state **S** keeps track of all the information that the conversational system requires to make its decision about how to answer the user.

A second model known as dialog policy (DP) decides what the next system action should be based on the dialog state. The DP model can be viewed as a function that maps a given dialog state into an appropriate action. These terms are defined below:

- Dialog state: This describes the representation of dialog state, the data it contains, and whether engineers created it manually or learned it from data.
- Dialog state tracking: This dimension identifies how the dialog state is determined.
- Dialog policy: This dimension identifies how the next action is determined.

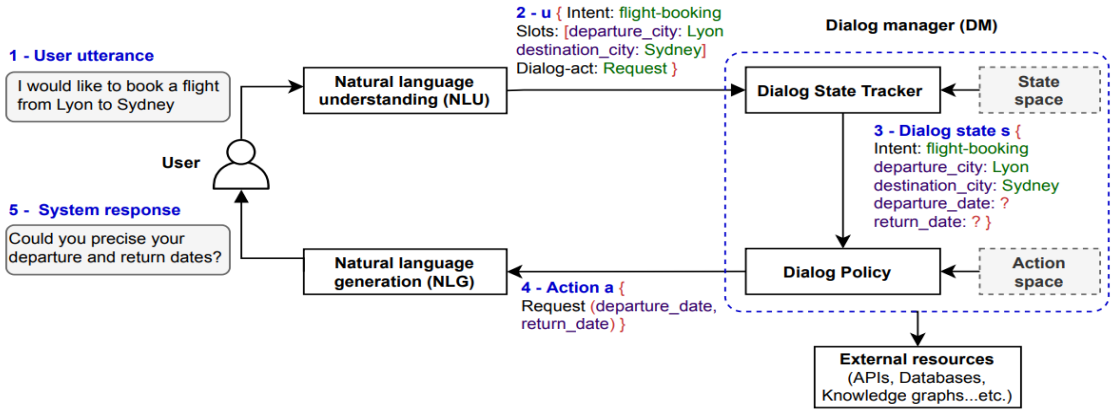

Fig.9 Dialog Manager and flow of information

5.4.2.3 DM approaches

We analyzed the literature and platforms representative of a wide range of bot development to identify three approaches that have been essentially adopted for implementing DM models.

- Handcrafted approaches: To track the dialog state and define the policy, handcrafted approaches rely on fully specified programs or models created by developers or domain experts.
- Data-driven approaches: In contrast to handcrafted approaches where the DM logic has to be defined by hand, data-driven approaches learn the dialog state and policy directly from data. They involve mainly machine learning (ML) approaches, including supervised learning (SL) and reinforcement learning (RL).
- Hybrid approaches: This hybrid approach to DM rests on the idea of combination. There could be multiple approaches, either handcrafted or data-driven, in order to capitalize on the benefits of each.

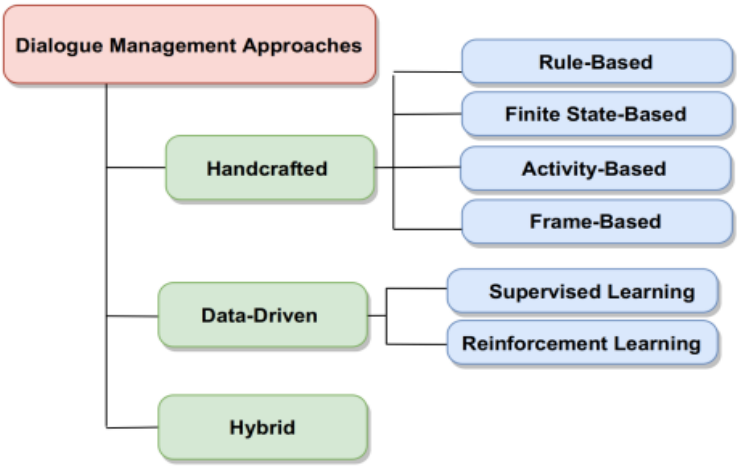

Fig.10 Dialog Management approaches

5.4.2.3.1 Types of Handcrafted approaches

We can, therefore, distinguish four kinds of handcrafted approaches: rule-based, finite state-based, activity-based, and frame-based.

Rule-based: In this approach, the bot developers claim that a set of rules encodes both the dialog state and policy. The major advantages of a rule-based approach include simplicity and no requirement for training data. They have no flexibility and involve a lot of effort on behalf of the developers in order to encode the rules. They have overlaps and even conflicts between them with growing rules.

Finite-state-based: a finite-state model offering a fixed number of sequences of steps indicating the state of the dialog at any time in the conversation; every state is limited to a prespecified number of transitions to other states, specifying the set of actions a conversational system can/should perform in a certain situation. Transitions are triggered as a result of recognizing a pattern that matches a user utterance.

The major benefits and limitations that characterize the rule-based approach are found in finite state-based. This approach is not robust and versatile in cases where the user does not follow predefined sequences of states. Now, considering the FSM example, even if the first utterance by a user contains all the necessary information—"I want to book a one-trip flight from Lyon to Sydney next Monday"—the dialog manager cannot handle this situation because it is programmed to solicit the information one by one according to the sequence of the states defined.

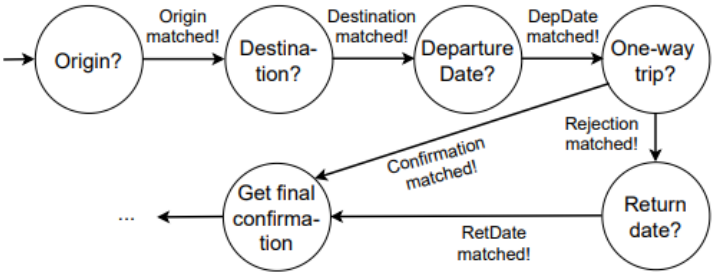

Fig.11 DM model defined using a finite state machine (FSM)

Activity-Based: With activities, triggers, and actions as the basic units of workflows, constructing a DM model based on an activity-based approach becomes easy. The activity-based model represents the procedural approach, hence often directly implementing dialog management by specifying exactly the workflow it may take in the course of conversation. Most platforms, like Chatfuel, FlowXO, and ManyChat, simplify this process of defining workflows by providing both a design canvas and visual elements for them.

Frame-Based: This model relies on a domain ontology defining a set of frames. Each of these frames specifies the type of information the system is designed to elicit from the user when carrying out any particular dialog task.

It is this sort of system wherein the dialog state shows the current condition of the frame, that is, which slots are filled and which are not. It figures this out using the DST and NLU outputs, which normally automatically derive from hand-written rules. The DP model is relatively straightforward in that it involves just asking questions until the filling up of the entire frame, and after that, reporting back the results of the action associated with the frame to the NLG.

The frame-based approach is able to be more flexible compared to FSM and activity-based approaches; besides its ability to process over-informative inputs entered by the user, it lets users fill in slots in different orders and in different combinations. This in turn involves complex DP algorithms to determine the next system action or question from a set of features, mainly the previous utterance of the user, slots to be filled, and some priorities devoted to dialog control.

5.4.2.3.2 Types of Data-driven approaches

These are primarily machine learning approaches that include supervised learning and reinforcement learning. Supervised and Reinforcement learning are also discussed in ISO/IEC 23053:2022, Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML).

Supervised: A pure corpus of labeled data is learned through the supervised approach. It includes statistical learning algorithms and neural-based approaches with attention mechanisms. There are two ways: first, a pipeline module is trained independently from DST and NLU modules in the DP model. On the other hand, DP may be implemented as an end-to-end model where it takes the user utterance directly, feeds it, and gives outputs on system actions. One of the assumptions concerning supervised learning models is that they require enough and good-quality training data, and such models do not adapt to user feedback with regard to changing the dialog policy.

Reinforcement learning: RL primarily focuses on treating dialog management as an optimization problem wherein performance is improved over time and continuously due to experience with the user. During any conversation, there is a given state that the dialog manager is in, takes some action for a particular request, and then transitions to a new state. In the case of a correct state, there is a reward, but if the model is not able to reach a correct or required state, then there is a small penalty. The drawback here is that this approach easily runs into intractable issues in large sets of states. Reinforcement learning recovery also requires enough and good training data, and it also requires domain-specific knowledge.

5.4.2.3.3 Hybrid approaches:

In these respects, some hybrid models combine a rule-based model with a supervised learning model or field more than one model of supervised learning. These machine-learning-equipped models could, therefore, turn in better performances than both ML and rule-base models working separately. A form of hybrid approach is ensemble learning, where the output from many DST models is synthesized to improve performance beyond any single model. Hybrid approaches can reduce the learning complexity and amount of training data, and they can provide control over conversational flow. Conversely, they require powerful resource settings and sufficient and high-quality training and data.

Data-driven approaches are much ahead of handcrafted ones because they have the capacity to learn such contextual information through training. In data-driven approaches, neural networks like RNN and models such as LSTM and GRU are used. They can trace the long-term dependencies of the input sequence and thereby update the dialog state appropriately.

5.4.2.4 Response Generation

For response generation, generally, dialog management uses an external database for information retrieval. There are two types of models that can be used: Retrieval based chatbots and Generative Based chatbots.

Retrieval models learn a set of responses and experiences from which to determine an appropriate reply, taking into account any input and context. The standards by which the evaluation should take place may be something as simple as rule-based charting or as complex as machine learning ensembles. These methods select one response from a collection of many, compared to generating new text. Unless trained with a large and diversified dataset with sufficient annotation and feedback, retrieval-based chatbots hardly make any mistakes. But they may come across as rigid, and the replies don't seem "human"; hence, their approach is always limited.

On the other hand, there are no predetermined responses when generating complex chatbot models. Original ideas are created from scratch. In the generative-based chatbot models, auto-conversion is common, and these models learn from scratch; therefore, they are used in the development of advanced chatbots that are pretty progressive in functionality [16].

Discussed below is a response generation pipeline with five stages [12]: signal analysis, data interpretation, document planning, microplanning, and realization.

Basically, numerical data input goes through two stages: signal analysis and data interpretation.

- Signal analysis involves the identification of patterns and trends through the pattern matching technique in data.
- Data interpretation uses symbolic reasoning methods based on domain knowledge in an effort to analyze complex messages and relationships between messages.

Document planning can also be viewed as consisting of two steps: content determination and discourse planning.

- Content determination is a step where it is decided what information is required to be given to the user. It consists of filtering information and summarizing it.
- Discourse planning deals with the text structure and organization to present it in such a way that it is understandable to humans.

Micro planning: Micro planning can also be referred to as sentence planning. It involves coming up with the sentences that deliver the information. It has three tasks to complete: referring, expression aggregation, and lexical selection.

- Referring expression refers to choosing the appropriate way of addressing an entity given a particular context.
- Aggregation is the issue of joining content with the use of conjunctions.
- Lexical selection is the issue of looking for appropriate words for the text.

Realization deals with the linguistic knowledge of the content. For example, to pick the correct tense, enforce the subject-verb agreement by applying the word ordering rule.

There are multiple approaches for response generation, like statistical approaches such as N-gram models and neural-based networks like RNN, LSTM, and transformers.

5.5 Testing and validation

This phase aligns with the "Verification and Validation" phase (Sections 6.4.11 to 6.4.13) outlined in ISO/IEC 5338:2023 Information Technology - Artificial Intelligence - AI System Life Cycle Processes. During this phase, the AI system, including chatbots, undergoes rigorous testing and validation to ensure it meets the required standards and performs as expected. For chatbots, this involves:

- Ensuring proper testing to evaluate the chatbot's accuracy, speed, and overall utility.
- Identifying any issues or limitations during testing to optimize chatbot performance.
- Testing across all channels and integrations to verify consistent functionality.
- Checking visuals, links, and language understanding to ensure a seamless user experience.

The following phase of documentation:

- Description of the criteria for estimation and applied measurements that decide the performance of the model and the performance of the application.
- Test and validation results were documented in a manner detailing the performance metrics obtained.
- Description of changes done to improve model performance, or hyperparameter changes based on the findings from evaluation.
- Methods for identifying and mitigating bias in the training data used for making predictions to ensure fairness and reliability.

While all the software testing methods and techniques mentioned in ISO/ IEC/IEEE 29119-1, Software and systems engineering — Software testing — Part 1: General concepts are applicable for chatbots, mathematical based testing is not typically used in chatbots but can be helpful in some cases. Mathematical-based testing refers to the application of mathematical techniques and theories to design, implement, and evaluate tests for software systems. This approach can include formal methods and theorem proving. Automated tools can be used to check correctness of algorithms or for boundary conditions in code for loops.

5.5.1 Types of chatbot testing

There are multiple kinds of testing for chatbots [43], [44], [47].

5.5.1.1 Functional testing

Functional testing aligns with Clause 4 of the ISO/IEC/IEEE 29119-1:2022 standard, specifically sections 4.4.4 (Test design techniques) and 4.4.5 (Experience-based testing). This testing ensures that the chatbot correctly identifies user intents, extracts entities, and follows the conversation flow, verifying the functionality of each component.

a) **Intent identification testing:** Intent identification testing is the process of recognizing what the user wants. It is essential for any chatbot to recognize the intent or purpose behind the conversation. The chatbot has to understand what the needs of the user are and give an accurate reply. There are a number of metrics that may be used to check the performance of a model for classifying intent.
b) **Batch testing**: Batch testing is a series of tests used to check the sharpness of the AI brain. It is, hence, an evaluation metric telling us how good our bot is at recognizing intents and entities. There are multiple kinds of utterances used in batch testing. A few of them are explained below:
    - Frequently used utterances: This is the most common case that a chatbot can encounter. For example, who is my manager?
    - Command-like utterances: Utterances that are like command. For example, get me the manager's name.
    - Short-form and specific terms: The chatbot should be able to recognize any short-form abbreviations or terms that are used in a domain, and these cases must be covered in the test.
    - Utterances with noise words: It is necessary that our chatbot understand the intent of an utterance that has some noise words or unpleasantly worded words.
    - Spelling mistakes: It should be kept in mind that humans are able to make mistakes when they type. The chatbot has to be able to recognize any spelling mistakes and should be able to handle such cases.
    - Longer Utterance: This may be because of encountering a longer utterance and the chatbot should be able to handle the longer utterance and respond appropriately.

c) **Entity extraction testing:** Entities are the intrinsic parts that make chatbots actually smart. There are a few kinds of testing that should be done to ensure that the chatbot can extract entities accurately.
    - Invalid entity values: Testing on invalid entity values is required. It helps us understand what the output of the chatbot will be if the entity is not aligned with the expected value.

- Entity synonyms: humans can use synonyms for entities that may be used interchangeably. In that case, the chatbot should be able to map different words to same entity. For example, when a chatbot offers a yes or no choice to the user, the synonyms used could be OK, go ahead, go ahead; for no, it could be not OK or nope.
- Sub-intent-based extraction of entities: The chatbot should extract all of the entities and sub-entities in a composite query.
- Varied-length entity values: This helps the chatbot deal with varied-length input and can extract the entities pertaining to that data.

d) **Small talk testing:** It is that kind of testing that ensures that our chatbot has the capability to do small talk. If our bot does not perform well during small talk testing, then it will create an issue in answering user intent and FAQs.
e) **Emoji testing:** A lot of times people use emojis to convey their feelings. The chatbot should interpret and understand the meaning of different emojis to enhance the user experience.
f) **Testing Conversation Flow:** Testing Conversation Flow will ensure that the chatbot's conversation with the user is natural, and it will consider all possible use cases and directions that the chatbot can take.

5.5.1.2 Usability Testing

Usability testing, covered in sections 4.4.12 (Test environments) and 4.4.7 (Manual and automated testing) of the ISO/IEC/IEEE 29119-1:2022 standard, evaluates the user experience, ensuring the chatbot is accessible, intuitive, and meets user expectations across different devices and platforms.

a) **Navigation Testing:** Navigation testing is centered on how the user can navigate their way through the chatbot's different features and options. The menu structure, buttons, and link testing should also be done to make sure that it will be easy to find information for customers. This criterion of testing facilitates making the navigation and usability of the chatbot optimal.
b) **Visual and link testing:** It is important to enhance the visual display of images and other GIFs without distortion, which will improve the user experience. It checks if visual elements are loaded quickly to avoid unwanted delays, and it checks if visual elements are uniform among different deployment channels. Tests on the hyperlinks should find out if they port users to where they are supposed to go, that they are clickable and functional, and that they execute consistently across a number of browsers and devices.

5.5.1.3 Performance Testing

Performance testing falls under sections 4.4.8 (Continuous testing) and 4.4.12 (Test environments) of the ISO/IEC/IEEE 29119-1:2022 standard. This testing assesses the chatbot’s response time, stability, and scalability under varying conditions, ensuring consistent performance.

This test is conducted to gauge response time, speed, and the number of concurrent interactions that the chatbot can handle. This testing thus evaluates the speed and chatbot response time to ensure that user queries can be supported efficiently with very relevant and timely responses. Performance testing will enable organizations to enhance the overall user experience of their product and increase customers' satisfaction.

In terms of speed and responsiveness, a chatbot is supposed to have benchmarks set and its performance gauged against those parameters. This can be done through simulation of all sorts of user scenarios and checking the performance of the chatbot because of multiple interactions handled simultaneously. Organizations can stress test under high traffic conditions, identify performance bottlenecks, and optimize measures undertaken to improve the bot's performance.

5.5.1.4 Integration Testing

Integration testing aligns with sections 4.4.4 (Test design techniques) and 4.4.7 (Manual and automated testing) of the ISO/IEC/IEEE 29119-1:2022 standard. It verifies the chatbot's ability to function correctly with other systems, ensuring seamless operation across various platforms.

a) **Multi-Channel Testing:** Checking the consistency of all the deployment channels for seamlessness. Deploy the chatbot on multiple channels, for example, websites, messaging apps, and social media, and verify its performance and output with each of them.
b) **Browser and Screen Size Compatibility:** The chatbot has to be checked to confirm that the responses will be the same across different browsers and devices. Test if the chatbot responds without any problems across different browsers, including Chrome, Firefox, and Safari, and on various devices like smartphones, tablets, and desktops with different screen sizes and resolutions.
c) **API compatibility testing:** Testing API compatibility means checking whether a chatbot is able to communicate well and exchange data with different systems. This test case is usually aimed at making sure that there is smooth interoperability and accuracy during the transfer of data between the under-tested chatbot and external APIs.

5.5.1.5 Error Handling Testing

Error handling testing is covered in sections 4.4.4 (Test design techniques) and 4.4.5 (Experience-based testing) of the ISO/IEC/IEEE 29119-1:2022 standard. This testing checks how the chatbot manages unexpected inputs and provides feedback, ensuring robust error management.

a) **Negative testing:** testing to ensure that the chatbot does not identify an intent in some cases wrongly. There may be multiple types of utterances under the category of negative testing.
    - Out-of-scope utterances: These are types of utterances that do not align within the scope of the chatbot service. Some of these out-of-scope requests may not be immediately recognizable as true negatives. Our chat will be able to recognize these kinds of intents. For instance, if a chatbot is built for customer support and someone

is trying to order a new laptop or computer using that chatbot, then these kinds of utterances are known as out-of-scope utterances.

- Out-of-domain: Any other out-of-domain utterance is otherwise a request or a question that doesn't fit the domain of a particular chatbot and should thus be recognized as true negative during batch testing. This would be the case when a chatbot has a domain relative to IT and a user sends an inquiry about buying a chair; it should recognize this kind of utterance as a true negative.

b) **Localization testing:** Depending on the region where we are deploying a chatbot, we should also consider localization testing. This can include grammar differences in different regions, genders, people, ages, and so on.

c) **Ambiguous intent testing:** It could be a situation in which the user's utterance is not clear, leading to multiple possible interpretations. The system can find more than one intent that is pertinent to the user's request.

5.5.1.6 Chatbot security and compliance testing

Security and compliance testing are crucial components addressed in sections 4.4.11 (Mathematical-based and fuzz testing) and 4.4.7 (Manual and automated testing) of the ISO/IEC/IEEE 29119-1:2022 standard. Security testing ensures that the chatbot properly implements security controls, such as data encryption, authentication, and authorization, to safeguard user data and prevent breaches, unauthorized access, and other security threats. Rigorous testing identifies vulnerabilities early, allowing them to be fixed before deployment. Compliance testing ensures that the chatbot adheres to relevant data protection regulations and industry standards, such as GDPR and HIPAA, ensuring privacy and maintaining user trust. Some of the key features for security and compliance are described below:

- **Data encryption:** Testing of the encryption algorithms and protocols that the chatbot implements assures that the user's data is robustly protected both while in storage and during transmission.
- **Authentication and Authorization:** The mechanisms for authentication and authorization of users should be verified to ascertain that there is secure verification of user identities within the chatbot and conducting activities with proper access privileges. Rigorous testing of these mechanisms prevents unauthorized access to sensitive data and ensures that users may access only such information and functionality as they are authorized to use, improving the overall security and trust of users.
- **Data protection regulations compliance:** Checking if the chatbot complies with relevant data protection regulations such as GDPR or HIPAA ensures the handling of user data in accordance with legal requirements. This includes the assessment of user data collection, storage, processing, and sharing, along with safeguards for users' privacy and rights under the law.

- **Data Storage and Retention:** Users should be aware about their data that is stored, managed, or what deletion processes are used. Practices of data storage will be assessed to avoid unauthorized access, methods for deleting data securely so that user data remains private, and compliance with provisions under the regulations relating to the period that data can be stored to reduce the risks of having data stored for a very long time.
- **Secure Integration with APIs:** Testing the integration of the chatbot with external systems and APIs involves checking for secure data exchange without violating the user's confidentiality, integrity, or availability.
- **Penetration testing:** Conduct simulated cyberattacks to identify vulnerabilities and weaknesses in the chatbot's security posture. Penetration testing aids in identifying security flaws that malicious actors might exploit.

These testing methodologies, including functional, usability, performance, integration, error handling, and security and compliance testing, fall under the categories of test design techniques, environments, and experience-based testing as outlined in Clause 4 of the ISO/IEC/IEEE 29119-1:2022 standard. They collectively ensure that chatbots are functional, user-friendly, secure, and reliable across various scenarios.

5.5.2 Testing metrics for models

There may be numerous testing metrics to be used, and all of these might help in the verification that our chatbot does work regarding the requirements [42]. Different classification metrics and control criteria in machine learning classification performance assessment are discussed in ISO/IEC TS 4213:2022, Information technology — Artificial intelligence — Assessment of machine learning classification performance. Among these metrics following could be used in intent classification and entity extraction:

- Accuracy: This is a metric for computing the ratio of correctly classified instances out of the total number of instances in the testing set.

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

Fig.12 Accuracy

- Precision: This measures the ratio of correct positive instances classified under some intent to the total number of instances predicted as an intent. Precision is used to find out how nice our model is doing for each intent, not considering the false negatives themselves.

$$Precision = \frac{TP}{TP + FP}$$

Fig.13 Precision

- Recall: This is the ratio of the number of valid positive instances to all positive instances in the intent category. Recall is useful because it shows how a model can detect all instances of a particular intent, which ignores false positive responses.

$$Recall = \frac{TP}{TP + FN}$$

Fig.14 Recall

- F1 score: This indicates the harmonic mean of precision and recall. The F1 score will give a balanced measure of a model's performance, especially when dealing with an imbalanced dataset or when both types of errors are equally important.

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Fig.15 F1 Score

- Confusion metric: table with the count of true positives, false positives, true negatives, and false negatives for every class of intent. We will use this later to easily compute accuracy, precision, recall, and visualize model predictions.

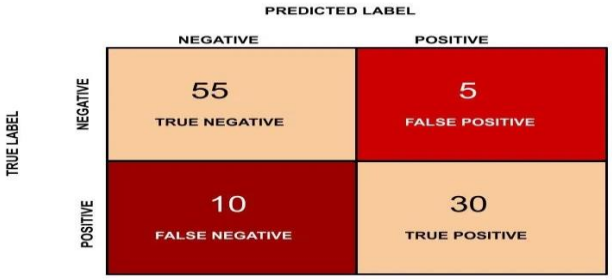

Fig.16 Confusion matrix

There are some metrics that are used to evaluate the text generated by chatbot. These metrics are among many used in an effort or attempt to quantify the similarity between what the generator produced and what the reference is:

- BLEU looks at groups of words—n-grams—and determines how many of these groups exist in a candidate translation and also in the reference translation. It tends to focus on precision without worrying too much about missing parts. BLEU has the tendency to be extra harsh on shorter translations due to its penalty for brevity. The combined BLEU score ranges from 0 (worst) to 1 (perfect). For natural language processing-based tasks, a score of more than 0.6 is considered exceptional [11], [15].
- Cosine similarity can also be computed between the generated response and the reference response in vector representation. The cosine similarity value will be near one if there is more similarity between the two vectors; otherwise, it will be near zero [11].
- METEOR tries to present a balanced view, considering both precision and recall—that is, it looks at the proportion of words correctly translated and needed words present. It also considers word order—ranking sequences incorrectly—and is, therefore, more sensitive to the arrangement of words. Unlike BLEU, METEOR does not overly punish shorter translations. In other words, even as BLEU focuses more on the precision of the word

groups, METEOR has a tendency to align with human judgment by paying more emphasis on individual word matches and their proper order. METEOR scores range from 0 to 1, where higher values indicate increasing similarity to the reference [15].

- ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. ROUGE primarily focuses on recall, which means it checks how many of the important words or phrases from the reference text appear in the candidate translation or summary. It looks at various n-grams (sequences of words) and also includes measures like ROUGE-N (for n-grams), ROUGE-L (for longest common subsequences), and ROUGE-S (for skip-bigrams). Also, ROUGE scores range from 0 to 1, where higher scores indicate that the generated descriptions are closer in similarity to the reference [15].

There can be other metrics that can be used to check whether the chatbot is able to classify intents, extract entities, generate responses that satisfy user needs, and comprehend user intentions.

5.6 Chatbot Deployment

The next step is deployment of a chatbot and defining its purpose for integration with existing platforms. This phase corresponds to the "Deployment" phase (Section 6.4.12) outlined in ISO/IEC 5338:2023 Information Technology - Artificial Intelligence - AI System Life Cycle Processes. In this phase chatbot is transitioned into a production environment.These foundational steps ensure that a chatbot meets business objectives and provides a frictionless user experience [46]. The documentation process for this phase should include considerations for deploying the trained model into the production environment and documentation of any compatibility or integration issues with existing systems or APIs.

a) Choosing the Right Platforms

Choose the right platform(s) for the chatbot's deployment so that it can reach its targeted audience. First, it is important to understand where your users are most active: on the website, on mobile apps, or on the top messaging platforms like Facebook Messenger, WhatsApp, or Slack. This selection will help to improve engagement and satisfaction.

b) Infrastructure Setup

Setting up the required infrastructure is important for the smooth functioning of the chatbot. This involves setting up the servers, databases, and networking parts. It should be scalable to bear the estimated load and traffic. Cloud services like AWS, Azure, or Google Cloud can be used for scalable and reliable features. Proper set-up features in the infrastructure lay the foundation for stable and efficient service for your chatbot.

c) Scalability and Reliability

The solution should support scalability, in that it should be able to scale up the dynamic resources whenever there is an increased demand from users. Using cloud services may

mean that scalability of your infrastructure can easily grow with the number of users. Load balancers distribute the incoming traffic from the servers so that no one server becomes overwhelmed. Scalability and reliability improve performance and availability to give users a feeling of consistency and responsiveness.

d) Integration

Integration of your chatbot with the selected platform is done using APIs or SDKs that a certain platform provides. This step will ensure that the chatbot is well-connected with the platform itself, enabling it to use all the platform-specific features and adhere to any constraints. Proper integration is critical for the chatbot to work within the ecosystem of the platform, providing a seamless and cohesive user experience.

e) Connecting Your Chatbot to Other Systems

The integration of a chatbot with other previous systems smooths operations and creates the best experience for customers. It designs and establishes a clear route of communication that the chatbot will use to interface with databases, applications, and software. This is done through APIs, or middleware. It should, therefore, be compatible and scalable to handle a growing volume of interactions while keeping up with changing business processes and technology.

f) Security

Security in chatbots is very important, which means keeping the data of the user and all interactions secure. Implement encryption methods during data transmission with secure authentication methods to preserve the identity of users. Therefore, compliance with the existing laws on personal data protection, for example, GDPR and HIPAA, is very necessary to avoid legal implications and gain the trust of the users. Insecurity, you would have ensured information protection for your users while also maximizing the level of your bot's trustworthiness.

g) Testing

Before you go live with your chatbot, proper testing has to take place. Functional testing continues the review of all features that are working well; performance testing is necessary to check how a bot reacts under load; and security testing gives an indication if there can be hacks in the chatbot or not. Beta testing with an audience will produce worthwhile feedback and show problems before they are launched widely. Comprehensive testing will ensure that the chatbot is ready for deployment and able to deliver a good-quality user experience.

5.7 Monitoring and Maintenance

Monitoring and maintenance are essential tasks that must be performed continuously to ensure the chatbot operates smoothly. This involves monitoring performance metrics, uptime, and user interactions. Tools can be employed for anomaly detection, triggering alerts for potential issues, and providing insights into the chatbot's overall health. Regular updates are critical for fixing bugs, introducing new features, and enhancing performance based on user feedback and analytics. This approach ensures the chatbot remains reliable and efficient.

This phase aligns with the "Operation and Monitoring" phase (Sections 6.4.15 to 6.4.16) and the "Continuous Validation" phase (Section 6.4.14) as outlined in ISO/IEC 5338:2023 Information Technology - Artificial Intelligence - AI System Life Cycle Processes. It encompasses ongoing operations, monitoring, and validation to maintain performance and implement necessary updates.

Collecting user feedback is a continuous process that helps one better understand the needs of the users and enhances the functionality of the chatbot. The insights are derived from surveys, analytics, and direct user feedback. On the basis of this feedback, iterate over the design of the chatbot and its features for any necessary adjustments to improve user satisfaction. It creates continuous improvement and keeps a relevant and effective chatbot by meeting user expectations through user feedback.

The documentation for this phase should include:

- methods for error detection, including automated monitoring tools and manual review process,
- techniques for error analysis to identify root causes and patterns in error,
- procedure for model iteration based on user and performance feedback,
- strategies to retrain and improve the model,
- documentation of strategies to improve the model based on user feedback and real-world data,
- implementing version control for modes, data, and code to ensure reproducibility,
- identification of threats and attacks and methods for mitigating attacks,
- procedures for regular security assessments and updates to address vulnerabilities.

5.7.1 AI-specific threats and attacks

There are multiple threats and attacks that are possible post-deployment. These kinds of threats need constant monitoring, and if we are able to detect these threats, then it will be easier to resolve them. Some of these threats and attacks are listed below:

a) Data drift

Data drift with respect to chatbots refers to the scenario in which the runtime distribution of input data drifted from the data used during training and led to a degradation in performance or safety. Often, this is due to incomplete representation at training time against the domain of the input—for instance, not considering seasonal variations in the input data. It is either the unsuitable choice of training datasets or the data distributions being trained on that just do not model the real-world application context properly. Such issues can, in principle, be reduced and removed through better modeling of input data and retraining [5].

b) Concept Drift

Concept drift is the term for changes in the relationship between the input variables and the model's output, which typically follow changes in the distribution of the input data. This means chatbots, user queries, or patterns of interaction have changed over time. By analyzing chatbot performance metrics and errors, it is possible to detect concept and data drift in chatbots, and this can further help to resolve the issue using retraining of models [5].

Data and concept drift can make robustness difficult, both for chatbots' performance and their dependability in a changing environment. This needs to be taken into account to make sure the chatbot meets user needs in this area.

c) Adversarial attacks

In chatbots, one expects special vulnerabilities, especially in complex AI models such as neural networks. One of these involves their susceptibility to adversarial machine learning attacks. Adversarial attacks fool an AI model to produce the wrong output by adding very small, usually imperceptible perturbations in the input; they can be intentionally introduced. Because most of the adversarial examples tend to remain effective across different model architectures, they are highly successful and pose a serious threat to AI systems, including chatbots. Several countermeasures can, however, be taken to mitigate the impact of such adversarial attacks on chatbots. Adversarial training involves training an AI model with adversarial examples to help the chatbot tackle such attacks more effectively [4].

Risk factors of AI systems, problem related to verification and validation, possible solutions and control and mitigation of risks, are mentioned in ISO/IEC TR 5469:2024, Artificial intelligence — Functional safety and AI systems [5] some of which are applicable to chatbots are discussed above.

d) Model poisoning

Another emerging area of concern is model poisoning, in which malicious data is injected during the training phase. Therefore, there is a need for protection in the protection-for-learning processes and data collection steps. These vulnerabilities, if addressed, would help in guaranteeing reliable performance and safety from adversarial threats toward chatbots [4].

e) Model stealing

These model-stealing attacks are a major concern within the machine learning community, particularly with applications like chatbots, where models are exposed through APIs. These adversaries query the target model, typically a chatbot, a large number of times to collect its responses. With these responses, attackers can try to train another model that can be used to reproduce that behavior and performance. If the chatbot deals with sensitive information, the stolen model can be used to recover private details from the new inquiries, even if access to the original data is not obtained directly [4].

- Attack methodology: Attackers can obtain a wealth of responses from the target chatbot by sending multiple queries.
- Model training: After obtaining these responses, the attacker can use these responses to train a shadow model that effectively emulates the target chatbot in question.

These are the risks that model stealing imposes.

- Security implications: Model stealing infringes upon intellectual property rights held by the original developers of the model
- Privacy issues: Stolen models might empower fresh queries to infer sensitive information, thereby eroding the privacy of the user.

Mitigation measures can include rate limiting, authentication, response randomization, and monitoring as features to prevent such attacks.

f) Bias

Bias in AI is a phenomenon in which AI is biased or unfair toward certain groups, outcomes, or perspectives [4]. In regard to chatbots:

- Data Bias: This happens when there is some form of societal bias in the training data used to develop and fine-tune the chatbot, or if such training data does not represent all probable user groups. For instance, if a chatbot is trained only with data from one demographic group, then it will end up poorly serving users from other demographics and giving biased responses or service delivery.
- Algorithmic bias: This happens in the algorithms themselves and affects how the chatbot interprets and responds to user input, which may result in unintended discriminations or favoritisms during the interaction with the chatbot.

- Automation bias: over-reliance on the recommendations provided by chatbots, where users simply assume that they are unbiased or accurate in every context; this can exaggerate biases that could already be in the system.

Here are some ways to minimize bias in chatbots:

- Knowing where the biases may be within data, algorithms, or user interactions: Sources of bias identification.
- Mitigation strategies: diverse dataset collection techniques, algorithmic fairness checking, and continuous monitoring to detect and mitigate bias.
- Ethical considerations: ensuring that design and implementation are ethical in nature and that fairness and inclusivity in the interactions become hallmarks.

Bias in chatbots can be evaluated by defining metrics that measure performance across different user groups and enabling assurances that responses from the bot are fair to and accurate for all users, thereby defining what extends beyond simple demographic factors or other personal characteristic-based elements. Bias, types of unwanted bias and treatment of unwanted bias are discussed in ISO/IEC TR 24027:2021, Information technology — Artificial intelligence (AI) — Bias in AI systems and AI aided decision making. This is applicable to chatbots also.

g) Unpredictability

The responses from a chatbot can be very unreliable if they are of high unpredictability. Users want their chatbots to provide consistent and reliable interaction with clear responses related to the query. Otherwise, users might get frustrated and dissatisfied. Further, such inconsistent performance will naturally deteriorate faith in the abilities of the chatbot and dissuade users from further interacting with it in the long term [4].

One of the reasons for unpredictability in chatbots is AI hallucination. AI hallucination in AI chatbot or computer vision tool, perceives patterns or objects that are nonexistent, creating outputs that are nonsensical or inaccurate. It is due to lack of constraints that limit the possible outcome. It can be prevented using high quality data, predefined data template for response generation and testing [60].

h) Opaqueness or lack of transparency

The users mostly want to know why chatbots make certain responses and how they arrive at their conclusions. If chatbots are not able to provide valid reasons for assumptions used for any output or response, then this will cause users to lose interest in chatbot capabilities, like certain kinds of recommendations for financial advice or customer support. It also complicates the operation of feedback, given that users and developers alike find it hard to identify problems effectively [4].

Vulnerabilities, threats, challenges and mitigation measures related to AI systems and applications are discussed in ISO/IEC TR 24028:2020, Information technology — Artificial intelligence — Overview of trustworthiness in artificial intelligence [4]. Some of the relevant to chatbots are discussed above.

i) Data Breaches

A data breach is unauthorized access to sensitive information about users stored in a chatbot database. There are many other paths by which this may happen, such as hacking, malware, or even negligence from inside, causing the unauthorized use of personal information like usernames and passwords, among other confidential data. Data breaches can be very dangerous because they can lead to identity theft and financial scandals and tarnish the reputation of the organization.

Mitigation Strategies:

Strong encryption of the data at rest and in transit needs to be applied so that, in case of a data leakage scenario, an unauthorized entity would not easily get access to and understand it. Secure access controls and authentication mechanisms are to be implemented, ensuring that all data is accessible to only authorized personnel. Regular auditing and monitoring of access logs would identify possible breaches with suspicious activities, take action, and Limit damage in real-time [47], [48].

j) Authentication Vulnerabilities

The description is that weak authentication mechanisms may grant access to chatbot functionalities, letting the attacker conduct manipulative attacks on the chatbot, which probably will lead to actions, data breaches, and user account compromises, hence reducing the security level and reliability of the whole chatbot system.

Mitigation Strategies:

Strengthening security through multi-factor authentication guarantees that, in case the factors of authentication get compromised, access to information is still not allowed to unauthorized persons. Adopting safe authentication protocols, like OAuth, provides a sure way to ensure user session security. Reviewing and updating the mechanisms for authentication regularly helps in dealing with new threats and vulnerabilities to the system [47].

k) Integrity Attacks

The intention of the information that is given to users can be modified or manipulated, and such is a threat to chatbots. It is referred to as an integrity attack. This can further purposely mislead the user and degrade the reliability of the bot, which might do damage due to the spreading of wrong or harmful information.

Mitigation Strategies:

Information should be protected from unauthorized modifications by adopting cryptographic techniques. Integrity checks and validation procedures performed at regular

intervals help identify and correct manipulated data. Establishment of secure channels between the chatbot and its backend systems ensures that data exchanged from these agents is tamper-proof [47].

l) Denial-of-Service Attacks

DoS attacks are based on loading the chatbot with excessive requests in order to prevent it from performing its normal work. This approach can make the chatbot unavailable to users, degrade performance, or probably make it crash, which may mean a huge loss of service.

Mitigation Strategies:

By limiting the number of requests a user can make and preventing overload, rate limiting and throttling can help to mitigate dos attacks. The DDoS protection services helped by mitigating the heavy attack traffic. It continuously measures the pattern of traffic and deploys anomaly detection systems that identify such DOS attacks at an early stage, hence ensuring the chatbot keeps working and is responsive [47].

5.7.2 General chatbot metrics

There are some general metrics that can be used to monitor if a chatbot is able to meet user expectations [45]:

a) Total Number of Users

The popularity and engagement of your chatbot are determined by the total number of users. For example, if you had 50,000 website visitors in 2021 and 6,000 users of the chatbot, that would put your conversion rate at 12%. This metric enables one to learn about the appeal and efficiency of the bot. A low conversion rate would mean there is something wrong with the design, welcome message, or simply the low visibility of the bot.

b) User Satisfaction

User satisfaction refers to how the user would rate the effectiveness of the chatbot in engaging them. It might be measured with surveys at conversation closure and/or during particular events of interaction. High satisfaction means that the chatbot is very effective at engagingly meeting needs, and thus low scores are indicative of a chatbot that requires improvement.

c) Accuracy of a Chatbot

Accuracy refers to how precisely an AI- and NLP-powered chatbot understands the user query and makes applicable responses. This metric always needs training for high accuracy. In the rule-based chatbot, this metric does not apply because it has a lower chance of errors as it follows specific flows. High accuracy ensures a high level of trust and satisfaction with the bot among users.

5.7.3 Engagement metrics

These are the metric related to engagement [45]:

a) **Active Users:** Active users serve as an engagement metric for chatbots, more specifically in customer support, marketing, and sales contexts. The metric refers to the number of unique users who have commenced interaction with the chatbot within a given period of time. It offers insight, in near real-time, into how frequently the bot engages users. By studying active users, it would be possible to evaluate whether one has effectively deployed their chatbot and the user-engagement strategies adopted. Comparing active users to total users will, however, give a broader perspective regarding the chatbot's reach and popularity with your audience.

b) **New Users:** New users are a measure of those unique visitors who have interacted with your chatbot for the first time in any predefined period. Of all the metrics, this one particularly reaches out to become key in measuring marketing campaigns or strategies spread across a wider audience to make people aware of the set-up of the chatbot. A higher percentage of new users indicates greater effectiveness in marketing strategies and, hence, an increase in adoption of the bot amongst probable customers or users.

c) **Returning Users:** Returning users are those who have interacted with the chatbot previously and come back for more. The larger the returning user base, the more positive user experiences and satisfaction in meeting user requirements. Returning users are an important metric to track with regard to customer service, marketing, and sales, as this directly speaks to whether or not users are satisfied and actually using the chatbot over time.

d) **Average Conversation Duration:** Average conversation duration is the total duration of conversations users have with a chatbot for a certain timeframe. It shows how much the user engagement level is and how effective the chatbot is in answering queries or giving information. Although longer conversation durations could indicate deep user engagement or very complex questions, short conversation times may indicate efficient methods for reprocessing information or very clear conversational flow. Knowing the average conversation duration would optimize chatbot performance toward meeting the user's expectations.

e) **Bounce Rate:** This is the percentage of users who enter the chatbot interface but then leave without any subsequent actions. A high bounce rate simply means that users find the chatbot not helpful, engaging, or relevant to their needs. Bringing down this bounce rate should be important in order to retain users and effectively work out chatbots in general. Tracking the bounce rate will help realize problems with usability, content, or user experience that have to be improved in order to engender adequate user engagement and motivate them to remain and use the chatbot.

f) **Flow Completion Rate:** The flow completion rate refers to the portion of users completing the set conversation flows within a chatbot. As such, it measures how well users can

proceed through and complete interaction steps designed to realize certain outputs. A high completion rate would thus indicate that the conversation flow is easy to follow and useful, hence successful interactions. It could, therefore, be a very useful metric to keep an eye on so that issues or problems in design and user experience that make the completion of flows difficult are recognized and disposed of.

5.7.4 Conversational analytics

These are the metric related to conversational analytics [45]:

a) **Goal Conversion Rate (GCR):** Goal completion rate is one of the vital metrics related to customer service, marketing, and sales. This indicates the percentage of users who have successfully completed predefined goals using bot interaction. These could be anything from completing a transaction to resolving an issue affecting customer service. In this case, a high GCR would then mean that the chatbot is very effective in helping its users realize their respective goals. If this GCR is low, it will involve assessments of the usability and relevance of chatbots to the goals.

b) **Fallback Rate:** One of the metrics can be the fallback rate, which is that part of messages where the chatbot either does not understand the user's intent or can't answer it properly. It comes out to be an important KPI, showing the importance of finding areas for which more training is required by the chatbot to better understand user queries. A high fallback rate may indicate flaws in either the training of chatbots or conversational design. These fallback instances could be further used in training the handling of a wide range of user interactions by the chatbot's NLP.

c) **Human Takeover Rate (HTR):** If the HTR turns out to be high for customer support, then that would further mean that the chatbot is not in a position to support its users. Although a few human interventions might be envisaged or even helpful in complex questions, a high rate of HTR still indicates more training or fine-tuning before it can handle most user queries on its own. This shall ensure a fine balancing act between the automated and human-assisted systems in efforts to come up with user experiences that are efficient and effective by keeping track of HTRs.

5.7.5 Revenue Oriented Metric

Revenue generated or ROI is also used as one of the metrics for chatbots [45]:

a) **Revenue Generated:** Probably the most prominent of all metrics product marketing- and sales-oriented chatbots would be generating is revenue generated, which measures the financial value attributed to a chatbot directly from its activities. Examples include an e-commerce bot driving direct sales while conversations are going on, or it just passes leads to human sales agents for conversation and conversion into revenue. It enables one to understand how effective a chatbot is at driving real business outcomes by tracking the revenues it generates.

b) **Return on Investment/Pay-back Period:** The metric measures the financial returns derived from the deployed chatbot through cost savings, revenue generation, or even customer acquisition efficiencies. It can be used to justify continuous investments in the development and optimization of chatbots since these benefits are explicit, such as reduced operational costs or increased sales revenues. It facilitates tracking the ROI over time and drives strategic decisions about how to improve the chatbot for maximum business value.

6. Insights from field visit

The industry visit to the National Informatics Centre (NIC) provided important insights into the processes involved in the development, usage, and maintenance of chatbots. The development process at NIC is structured into different phases:

6.1 Domain Identification and Requirement Elicitation

The domain of the chatbot should be defined before entering the development phase. In the event that the problem statement is huge and complicated, it will be imperative to break this down into phases or focus on a smaller subset. The requirements regarding the chatbot are then clearly outlined, which may include the number of languages that the chatbot should be able to support and other added features such as RPA or integration into existing databases. At NIC, for any type of software or application development project, the Agile framework is used in keeping tabs on the progress of a project while maintaining all relevant documentation. For more details on the agile framework, refer to [58].

6.2 Data Collection for Chatbots

There is a requirement for data to be used for training to cover all scenarios that a chatbot may come across after it is deployed. At NIC, data is collected using FAQs and documents relevant to the particular department or organization for which the chatbot is being made. If the appropriate data is not available, then custom data is collected from various resources. Data is translated into English in cases where data is in local languages before training the model.

6.3 Building a Conversation

With the requirements and data in place, the next step is to build the conversation using intents, entities, and dialog management. For each use case, different intents are defined, such as application_status, login_status, and approval_status for an application tracking system. Entities mapped to these intents may include words like "status" related to application_status. Multiple entities can have different combinations that are mapped to an intent, like for information related to login and password, if user type “I am unable to login, I want to change password”, this utterance or entities will be mapped to intent having both entities.

While training a model for chatbot, entities in input are annotated so that chatbot is able to learn how to recognize what information is needed to extract and what is the intent for a particular query. NIC uses Google Dialog Flow for model building due to its ease of training and minimal utterance requirements for each intent. Masking techniques are applied to ensure personal information is not learned by the chatbot during training. Dialogs in chatbots developed by NIC are system directed, it helps to ensure that conversion is not open ended and user is replying to all questions asked by chatbots

6.4 Deployment

It trains models on cloud platforms and saves trained models with their weights and other information on local VMs or makes them available to the client organization. All data and transactions are secured by high-level encryption and network security measures. Also, NIC offers a platform called "Build Your Bot" to users for helping them to create chatbots via UI and backend just by connecting their Google Dialog Flow using Jtoken json file with the chatbot UI interface.

6.5 Post Deployment

Long-term analytics regarding the data and conversations are also provided to users through a dashboard by NIC in the post-deployment phase. The conversations are stored with masked user information like account numbers and addresses. These stored utterances are used to retrain the model to improve its performance. In case the chatbot does not perform as expected, the documentation related to the intents and entities is reviewed to resolve the issues. NIC ensures data security and protection through various network security protocols and data encryption. If the pretrained model is provided to client organization, then they need to ensure that they should recognize possible web security issues and perform required mitigation. Possible web security issues can include SQL injection, data security and encryption.

NIC employs the following measures to mitigate any kind of risk and bias:

- **Masking of Data:** Ensures sensitive information is not exposed.
- **Regular Security Audits:** Periodic audits to identify and rectify security vulnerabilities.
- **Retraining of Models:** Continuous model retraining on relevant data to improve performance and security.
- **Data Encryption:** Encryption of data in transit and at rest to protect against unauthorized access.
- **Additional Network Security:** Implementation of robust network security protocols to safeguard data integrity and confidentiality.

These practices collectively ensure that the chatbots developed at NIC are secure, efficient, and capable of delivering high performance. In addition to this NIC, also ensure that their application is not affected by OWASP Top 10 Security Risks. The OWASP (Open Web Application Security Project) Top 10 is a standard awareness document for developers and web application security. For more details related to OWASP Top 10, refer to [57].

7. Conclusion

Standards in general offer alot of advantages and a few of these advantages are listed below:

- Consistency Across Platforms: Guarantees uniform user experience with chatbots across different platforms and devices.
- Security and Data Protection: Guides the implementation of robust encryption and appropriate security measures to ensure adequate protection of users' data from possible breaches.
- Prevention of Developer Mistakes: Enforcing good practices and detailed documentation in chatbot development reduces vulnerabilities and errors.
- Risks on the Platforms: This facility answers to platform-specific security vulnerabilities so that chatbots could be protected against such possible attacks.
- Interoperability and API Security: Security and reliability improvement of interactions between a chatbot and other systems by standardized API protocols.
- User Privacy: It protects personal information by making sure that data is anonymized and protected against possible unauthorized accesses.

All the existing standards related to AI applications, Data quality and software address some aspects of chatbot development, such as ensuring security, data protection and data quality management. However none of these existing standards address in total how to develop chatbots, including data preprocessing techniques for text and dialog management.

Text Data Preprocessing Techniques: Current standards typically fail to consider some very important text preprocessing techniques like lemmatization, stemming, word embedding, and text vectorization. Each of these methods is extremely critical in processing raw textual data into a form that will be useful in training chatbots. Standardizing these methods will ensure perfect interchanging ability and effectiveness of preprocessing—critical factors in the creation of quality training data. Proper preprocessing improves the accuracy and reliability of natural language understanding for the chatbot, therefore making it well-prepared and competent in understanding and generating user responses. A new standard related to text processing could include:

a) **Text Data Preprocessing:** Address lemmatization, stemming, word embedding, and text vectorization.
b) **Tokenization:** Define guidelines for tokenizing text into individual words or subwords.
c) **Normalization:** Address normalization techniques like case conversion, removing punctuation, and handling contractions.
d) **Feature Engineering:** Provide recommendations for feature engineering techniques like n-grams, part-of-speech tagging, and named entity recognition.
e) **Text Representation:** Establish guidelines for representing text data in a numerical format.

f) **Data Cleaning and Quality Control:** Address data cleaning procedures to ensure data quality and reliability.
g) **Evaluation Metrics:** Define appropriate evaluation metrics to assess the effectiveness of text processing techniques.

Dialog Management: The issues with respect to this are those of handling different states and tracking these states, handling ambiguity, and finally, handling interactions with back-end databases and information systems. Existing standards do not comprehensively cover these issues. Such a standard in dialog management would be one umbrella under which to satisfy all these requirements, thus ensuring that chatbots handle conversations in a complex way: handling conversation flow, user intent, and context, and integration with external systems for pulling and updating data. A standardized approach in this respect enhances the ability of the chatbot to respond with coherent and contextually appropriate responses, leading to user satisfaction and more efficient interaction.

There are some key components and processes that could be included in new standard for Dialog management. List of these components and processes are listed below:

a) **State Management:** Clearly define and track conversation states, handle context switches, and prevent context drift to maintain coherence and avoid confusion.
b) **Intent Recognition:** Accurately determine the user's intent using techniques like keyword matching, machine learning, or natural language understanding. Handle ambiguous or complex intents by asking clarifying questions or providing options.
c) **Entity Extraction:** Identify relevant entities using named entity recognition techniques. Resolve ambiguities by using context or additional information.
d) **Response Generation:** Select appropriate responses based on intent and entities. Use response templates or generative models. Ensure responses are coherent, consistent, and aligned with the conversation context.
e) **Error Handling and Recovery:** Handle errors or misunderstandings gracefully. Provide clear and informative error messages.
f) **Evaluation and Improvement:** Establish metrics to measure dialog performance and use feedback loops for continuous improvement.
g) **Integration with Other Components**
- **NLU:** Integrate with NLU components for intent recognition and entity extraction.
- **NLG:** Integrate with NLG components for response generation.
- **Knowledge base:** Access and utilize a knowledge base for information retrieval.
- **External systems:** Integrate with external systems for data access and update.

8. References

1. IS/ISO/IEC 22989:2022, Information Technology — Artificial Intelligence — Artificial Intelligence Concepts and Terminology
2. IS/ISO/IEC 5339:2024, Information technology — Artificial intelligence — Guidance for AI applications
3. IS/ISO/IEC TR 24368:2022, Information technology — Artificial intelligence — Overview of ethical and societal concerns
4. IS/ISO/IEC 24028:2020 Information technology — Artificial intelligence — Overview of trustworthiness in artificial intelligence
5. IS/ISO/IEC 5469:2024, Artificial intelligence — Functional safety and AI systems
6. ISO/IEC FDIS 5259-1, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 1: Overview, terminology, and examples.
7. ISO/IEC DIS 5259-2, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 2: Data quality measures
8. ISO/IEC FDIS 5259-3, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 3: Data quality management requirements and guidelines.
9. ISO/IEC FDIS 5259-4, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 4: Data quality process framework.
10. ISO/IEC DIS 5259-5, Artificial intelligence — Data quality for analytics and machine learning (ML) — Part 5: Data quality governance framework.
11. Aleedy, M., Shaiba, H., & Bezbradica, M. (2019). Generating and analyzing chatbot responses using natural language processing. *International Journal of Advanced Computer Science and Applications*, *10*(9).
12. Huang, X., & CIS, A. (2021). Chatbot: design, architecture, and applications. *University of Pennsylvania: School of Engineering and Applied Science, Pennsylvania*, *1*.
13. Dagkoulis, I., & Moussiades, L. (2022, November). A Comparative Evaluation of Chatbot Development Platforms. In *Proceedings of the 26th Pan-Hellenic Conference on Informatics* (pp. 322-328).
14. Cahn, J. (2017). CHATBOT: Architecture, design, & development. *University of Pennsylvania School of Engineering and Applied Science Department of Computer and Information Science*.
15. Rosario, G., & Noever, D. (2023). Grading conversational responses of chatbots. *arXiv preprint arXiv:2303.12038*.
16. Pandey, S., & Sharma, S. (2023). A comparative study of retrieval-based and generative-based chatbots using deep learning and machine learning. *Healthcare Analytics*, *3*, 100198.
17. Brabra, H., Báez, M., Benatallah, B., Gaaloul, W., Bouguelia, S., & Zamanirad, S. (2021). Dialogue management in conversational systems: a review of approaches, challenges, and opportunities. *IEEE Transactions on Cognitive and Developmental Systems*, *14*(3), 783-798.
18. Harms, J. G., Kucherbaev, P., Bozzon, A., & Houben, G. J. (2018). Approaches for dialog management in conversational agents. *IEEE Internet Computing*, *23*(2), 13-22.
19. Johari, N. M., Nohuddin, P. N., Baharin, A. H. A., Yakob, N. A., & Ebadi, M. J. (2022). Features requirement elicitation process for designing a chatbot application. *IET Networks*.

20. Natural Language Processing (NLP) - Overview - GeeksforGeeks
21. What Is a Chatbot? | IBM
22. What Is a Chatbot? Definition, Types, and Examples | Coursera
23. 5 types of chatbot and how to choose the right one (ibm.com)
24. 30+ Chatbot Use Cases/Applications in Business (2023 Update) - Marketing Scoop
25. Building Smarter Chatbots: Intent and Entity Training Models (libraria.ai)
26. What are Recurrent Neural Networks? | IBM
27. Bidirectional Recurrent Neural Networks Definition | DeepAI
28. Transformers in Machine Learning - GeeksforGeeks
29. What is a Large Language Model (LLM) - GeeksforGeeks
30. Named Entity Recognition - GeeksforGeeks
31. The Beginner's Guide to the Chatbot Development Lifecycle - The Chatbot Business Framework
32. Chatbot Requirements: Technical & Non-Technical Considerations (botscrew.com)
33. Chatbot Requirements: How To Build An Ideal Solution - Springs (springsapps.com)
34. What Is Chatbot Design? | IBM
35. User Stories vs Use Cases: Choosing the Right Technique for Your Software Development Project - Visual Paradigm Guides (visual-paradigm.com)
36. Conversational Flow: Types & Best Practices | BotPenguin
37. From Inception to Implementation — The Process of Building an AI Chatbot | by Lee Hwee | KeyReply | Medium
38. The Complete Guide to Building a Chatbot with Deep Learning From Scratch | by Matthew Evan Taruno | Towards Data Science
39. Text Vectorization and Word Embedding | Guide to Master NLP (Part 5) (analyticsvidhya.com)
40. Word Embeddings in NLP - GeeksforGeeks
41. word2vec | Text | TensorFlow
42. How To Fine-Tune GPT-3 For Custom Intent Classification | Saturn Cloud Blog
43. Mastering Chatbot Testing: A Step-by-Step Guide (kore.ai)
44. Validating Your Chatbot: Essential Procedures and Tips (libraria.ai)
45. Chatbot Analytics: 14 Chatbot Metrics To Track in 2024 (botscrew.com)
46. Chatbot Deployment Strategies: A Step-by-Step Guide to Elevate Customer Experience - (dluxchat.com)
47. Securing Chatbots from Common Threats & Mitigating Risks (chat360.io)
48. Are Chatbots Safe to Use? (kaspersky.com)
49. TF IDF | Padhai Time
50. Word2Vec Explained (israelg99.github.io)
51. 4 — Bag of Words Model in NLP. In this article, we will cover the Bag… | by Aysel Aydin | Medium
52. One-Hot Encoding Explained | Baeldung on Computer Science
53. What is Natural Language Understanding (NLU)? | Definition from TechTarget

54. Artificial Intelligence | Natural Language Generation - GeeksforGeeks
55. Chatbot Data: Picking the Right Sources to Train Your Chatbot (landbot.io)
56. Text Generation Using N-Gram Model | by Oleg Borisov | Towards Data Science
57. OWASP Top Ten | OWASP Foundation
58. What are Agile frameworks? - GeeksforGeeks
59. NIST Artificial Intelligence Risk Management Framework (AI RMF 1.0)
60. What Are AI Hallucinations? | IBM

9. Bibliography

1. IS/ISO/IEC TR 24029-1:2021, Artificial Intelligence (AI) — Assessment of the robustness of neural networks, Part 1: Overview.
2. IS/ISO/IEC 24029-2:2023, Artificial intelligence (AI) — Assessment of the robustness of neural networks, Part 2: Methodology for the use of formal methods.
3. ISO/IEC TS 8200:2024, Information technology — Artificial intelligence — Controllability of automated artificial intelligence systems.
4. IS/ISO/IEC 8183:2023, Information technology — Artificial intelligence — Data life cycle framework.
5. IS/ISO/IEC 23053:2022, Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML).
6. IS/ISO/IEC TS 4213:2022, Information technology — Artificial intelligence — Assessment of machine learning classification performance.
7. ISO/IEC TR 17903:2024, Information technology — Artificial intelligence — Overview of machine learning computing devices.
8. IS/ISO/IEC 5338:2023, Information technology — Artificial intelligence — AI system life cycle processes.
9. ISO/IEC 23894:2023, Information technology — Artificial intelligence — Guidance on risk management.
10. ISO/IEC 42001:2023, Information technology — Artificial intelligence — Management system.
11. IS 11291 (Part 1) : 2023, ISO/IEC 29119-1 : 2022, Software and systems engineering – Software testing Part 1: General concepts