# An Evolutive Knowledge Base for "AskBot" Toward Inclusive and Smart Learning-based NLP Techniques

Khadija El azhari[1], Imane Hilal[2], Najima Daoudi[3], Rachida Ajhoun[4], Ikram Belgas[5]
Smart Systems Laboratory-E.N.S.I.A.S, Mohammed V University, Rabat, Morocco[1, 4]
LyRica Labs, School of Information Sciences, Rabat, Morocco[2, 3]
École Nationale Supérieure des Arts et Métiers, Casablanca, Morocco[5]

***Abstract*—Artificial Intelligence chatbots have shown a growing interest in different domains including e-learning. They support learners by answering their repetitive and massive questions. In this paper, we develop a smart learning architecture for an inclusive chatbot handling both text and voice messages. Thus, disabled learners can easily use it. We automatically extract, preprocess, vectorize, and construct AskBot's Knowledge Base. The present work evaluates various vectorization techniques with similarity measures to answer learners' questions. The proposed architecture handles both Wh-Questions starting with Wh words and Non-Wh-Questions, beginning with unpredictable words. Regarding Wh-Questions, we develop a neural network model to classify intents. Our results show that the model's accuracy and the F1-Score are equal to 99,5%, and 97% respectively. With a similarity score of 0.6, our findings indicate that TF-IDF has performed well, correctly answering 90% of the tested Wh-Questions. Concerning No-Wh Questions, soft cosine measure, and fasttext successfully answered 72% of Non-Wh-Question.**

***Keywords*—Knowledge base; KB; artificial intelligence; AI; chatbot; e-learning; cosine similarity; soft cosine similarity; TF-IDF; FastText; neural network**

## I. Introduction

The e-learning domain has shown impressive growth in applying AI technologies to enhance the quality of learning. e-Learning platforms are increasingly recognizing the importance of integrating AI to perform tasks effectively while saving time, energy, and cost. Based on AI technologies, the e-learning domain succeeds in (1) personalizing educational content according to each learner's needs and capabilities instead of proposing a standard approach to learning. Hence, the learning strategy becomes dynamic, customized, and individualized to encourage learners. (2) Providing specific information about each learner's progress, strengths, and weaknesses, and even attendance issues. Hence, data analysis on e-learning platforms enhances and improves learning experiences. (3) Developing multilanguage course content by integrating automatic translation tools that offer more speed and efficiency in translating huge amounts of content, thus saving time and offloading teachers from such tasks that consume time and energy. (4) Supporting learners by providing AI chatbots to answer learners' questions. Chatbots are available 24/7. Thus, learners are free to learn at their own pace through multiple devices.

Various chatbots have been created for use in multiple areas, including the e-learning domain. Especially, there is an exponential interest in AI chatbots in the last few years, especially from 2016 until now [1]. AI chatbots have been used in the e-learning domain to satisfy several needs such as (1) Promoting learners' interaction with online courses. (2) Providing information about administration [2], courses, and exam regulations (3) Sharing exercises and tips/hints to solve them and assist students to master the course knowledge (4) Answering repetitive and massive questions about the course knowledge (5) Recommending learning materials and educational resources according to the learner's need (6) Assessing learners' knowledge by automating exams and assignments (Sreelakshmi et al., 2019) (5) Encourage collaborative learning between learners (El Azhari et al., 2022).

Therefore, e-learning chatbots address several issues facing the e-learning domain, especially since they have succeeded in: (1) Automating repetitive tasks by performing them efficiently. Thus, helping tutors to focus their energies on more complicated tasks rather than answering repetitive and massive questions (El Azhari et al., 2021). Hence, saving time, energy, and cost. (2) Capitalizing knowledge from several sources and storing them as the chatbot Knowledge Base (KB), Thus, assisting learners to quickly find the reliable information needed without wasting time searching in many information sources. (3) Encouraging learners to ask their questions (Moreno-Guerrero et al., 2023) without being afraid of attracting attention from others, being criticized, and having their opinions misinterpreted. (4) Supporting interaction between learners and the chatbot, hence, learners ask their questions at their own pace through multiple devices without waiting for the course session (El Azhari et al., 2022).

Despite the efforts made to integrate AI chatbots in the e-learning domain, there are some drawbacks related to the manual process of creating e-learning chatbots. Specifically, several researchers (M. Verleger and J. Pembridge, 2018; Herrera, Yaguachi and Piedra, 2019; El Janati, Maach and El Ghanami, 2020; H. C. B. Chan and T. T. Fung, 2020; Tamayo et al., 2020; Deepika, Bala and Kumar, 2021a; Nhut Lam, Nhat Le and Kalita, 2022; Singh and Singh, 2022) propose AI chatbots by manually creating pairs of Questions and Answers (Q&As) and storing them as the chatbot's local KB. They manually create learners' intents rather than automatically extracting them. Thus, consuming time, energy, and cost.

In this paper, we will address these issues by proposing an inclusive chatbot called « AskBot » able to automatically understand learners' requests in text and voice formats without the need for prebuilt services in online platforms. Thus, saving

time, energy, and cost. This work differs from existing chatbots by providing an automatic approach to construct the AskBot's KB using the web scraping tool without wasting time in manually collecting Q and As related to the chatbot's domain.

The remainder of this paper is structured as follows: The next section outlines the context and the problem statement of the study. The third section presents related works. The fourth section provides AskBot's architecture, and the implementation process and the last section provides the conclusion and future works.

## II. Context and Problem Statement

### A. Overview of AI Chatbots and NLP Techniques

AI chatbots are based on NLP techniques, especially Natural Language Understanding (NLU) to understand what the user asks for and Natural Language Generation (NLG) to generate the appropriate response and answer correctly. AI chatbots recognize users' needs by applying the NLU technique. They extract two main elements: (1) The intent which means the user's intention behind asking a question or formulating a message. (2) Entities are values extracted from the user's input to understand the information needed. Fig. 1 demonstrates a use case of using AI chatbots to answer learners' questions.

Before understanding the user's request, it's highly recommended to transform it into a vector by applying the word embedding technique. Specifically, each word is converted into a single vector. It considers that a word is characterized by its context, i.e., by the words that surround it. Thus, words that share similar contexts also share similar meanings.

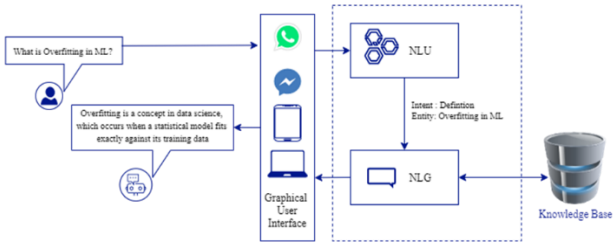


Fig. 1. Example of AI chatbot in an e-learning context.

In this paper, we test four well-known word embedding techniques:

*1) Term Frequency Inverse Document Frequency (TF-IDF) is a text vectorizer method that aims to transform the text into vectors by combining three main concepts: (i) Term Frequency (TF)*: It indicates the significance of a particular term by calculating the number of times it appears in a document. (ii) Document Frequency (DF): It refers to the number of documents that include a particular term. It reveals how frequently a term is used. (iii) Inverse Document Frequency (IDF) determines the weight of a particular term. It aims to reduce the weight of a term if its occurrences are dispersed over all the documents.

*2)* Word2Vec is based on two-layer neural networks and seeks to learn the vector representations of the words composing a text. Thus, words with similar contexts are represented by close numerical vectors.

*3) GloVe works based on two main methods*: (i) Global matrix factorization is the process of reducing huge term frequency matrices using matrix factorization techniques from linear algebra. (ii) Local context window methods are CBOW and Skip–Gram.

*4)* FastText encodes each word as an n-gram of characters rather than immediately learning word vectors. N-grams can be defined as continuous words, symbols, or token sequences in a document.

The chatbot uses the output of the vectorization step to answer given questions. Especially, it measures the similarity between the user's question and all stored questions, then, it returns the response of the most similar question to the user's question. To perform this process, the chatbot needs to measure similarities between questions by using techniques such as Cosine Similarity and Soft Cosine Measure. They are among the techniques widely used. Specifically, they have proven good results in different implementations :

*1) Cosine similarity*: It calculates the cosine of the angle formed by two vectors projected into a multidimensional space. The cosine similarity increases with decreasing angle.

*2) Soft cosine measure*: It assesses the similarity between two publications meaningfully even when they don't share any words. It has been demonstrated that it performs better than several cutting-edge techniques in determining semantic text similarity.

In this part, we introduced an overview of the main concepts used in our work to implement AskBot's architecture. The next part is dedicated to discussing the problem statement of the present work.

### B. Problem Statement

The e-Learning domain suffers from a variety of issues, especially, managing learners' questions. Specifically, tutors spend considerable time and energy manually answering repetitive and massive learners' questions. They are unavailable to manually handle them and satisfy all learners' needs [3], [4]. Thus, learners are forced to seek the information they need from several materials: Books, online links, and search engines [5]. Thus, consuming time and energy in filtering relevant and reliable information [6]. Hiring additional tutors might be a solution to handle massive learners' queries. However, it leads to high costs. For that reason, automatically answering learners' questions is needed to offload tutors from repetitive tasks [7]. Thus, they can concentrate their energies on more complicated tasks.

AI chatbots are highly integrated into e-Learning platforms to automatically respond to learners' questions [8]. They automate handling learners' requests by giving the appropriate answers to given queries. Thus, saving time, energy, and cost. As demonstrated by [9], using an AI chatbot positively impacts learner retention. Specifically, they compared learning using AI chatbots and the Google search engine. Outcomes show that learning through the proposed AI chatbot has a positive impact on memory retention and learning outcomes compared to the

learning through Google search engine. Hence, a chatbot is considered an effective tool [10] to support learners and encourage them to freely ask their questions without being evaluated or misjudged.

Several works have been carried out to automatically handle learners' questions. Many researchers propose AI chatbots to automatically answer learners' questions and satisfy their needs by saving time, energy, and cost. However, results are still insufficient and additional efforts are required to enhance AI chatbots in the e-Learning domain [11]. As shown by [11], [12], educational chatbots in the instant messaging application are still limited and need additional AI features. Especially, chatbots in the E-learning domain suffer from manually creating the chatbot's KB. Many researchers construct the KB by manually collecting pairs of Q&As. Thus, consuming time and energy. For that reason, additional efforts are needed to automate the creation process of the chatbot's KB. In addition to that, researchers prefer using platforms such as Dialog flow, IBM Watson, Wit.ai, etc. They facilitate the creation of chatbots through trained AI models. However, they are expensive and lead to high costs.

In this paper, we will address these issues by proposing a smart architecture based on AI technologies. Especially, we will automatically construct the chatbot's KB by scraping reliable and relevant knowledge. Then, automatically classifying intents by using a neural network model rather than wasting time manually creating training phrases. Specifically, the proposed chatbot can automatically understand learners' needs and satisfy them. Thus, saving time, and energy, and reducing costs. To demonstrate the feasibility of the proposed architecture, we will test the chatbot in the Machine learning (ML) domain. Thus, the designed chatbot will answer questions related to the ML domain.

## III. Related Works

Several works have been carried out to develop e-learning chatbots in different learning contexts to satisfy various needs. They support learners by giving them the information needed rapidly and efficiently. They understand learners' needs and respond correctly based on the local KB. In this section, we will present and discuss recent works focused on automatically constructing AI chatbots to assist learners.

Based on our review [11], most studies propose creating chatbots by manually feeding the KB. They propose constructing the chatbot's KB by manually creating a set of Q&As which is time- and energy-consuming [13]–[21]. On the other hand, some papers automate the process of creating e-learning chatbots by using recent technologies such as web scraping, Optical Character Recognition (OCR), and spider robots. The authors in [22] proposed an educational chatbot to answer questions related to the Data Science domain. They automatically extracted the chatbot KB by using the web scraping technique. Study [23] proposed a chatbot to deliver learning materials. They automatically collected relevant documents, indexed them by Elasticsearch, and stored them in Postgre Database. Researchers in [24] proposed an educational chatbot able to assess and evaluate students' knowledge. They used Apache PDFBox and an overgenerating system to automatically construct the local KB. Study [25] proposed a generative chatbot able to predict answers based on deep learning models. They applied these models to learn from an existing dataset and train the chatbot.

After analyzing previous works, we conclude that few papers proposed automating the creation of the chatbot's KB. They propose the use of web scraping techniques, OCR, or spider robots to automatically collect pairs of Q&As. However, the proposed approaches are still insufficient, and more efforts are needed to automatically understand users' intents without manually creating them. Specifically, they propose using some online platforms such as DialogFlow, IBM Watson, Wit.ai, and Rasa. etc. to facilitate the chatbot's training process without the need for technical knowledge in advanced NLP techniques [26]–[33]. These platforms already integrate NLU and NLG techniques, they reduce the energy needed to understand users' requests and answer them. However, there are some limitations:

*1)* In these platforms, the integration of Q&As is done manually by adding intent for each question with training phrases (Q&As) to help the chatbot learn from them. The manual integration of intents is time and energy-consuming because AI chatbots require a lot of Q&As to function properly. Otherwise, the chatbot will be restricted to the few Q&As included in training phrases because it is challenging to manually gather a huge number of Q&As.

*2)* These platforms are expensive, and the number of queries available in their free editions is constrained.

*3)* These platforms limit the channels to incorporate the chatbot.

The research [34] proposed an approach based on deep learning algorithms, they address the problem of manually creating intents by proposing a generative chatbot. It answers automatically to a user's request by predicting the response via a deep learning model without the need of referring to the KB or classifying the user's intent. Thus, they reduce the time needed to classify Q&As. However, a large amount of data is needed to train generative chatbots. Otherwise, they can result in illogical responses or incorrect answers if the chatbot is trained using little or poor-quality datasets.

In this paper, we aim to automatically handle users' questions through AskBot's architecture. Specifically, we propose a smart architecture to automate the process of detecting intents and understanding the real needs of learners, then, answering correctly without wasting time in classifying intents. Hence, saving time, energy, and cost.

## IV. Proposed Architecture

### A. Chatbot's Architecture

In this section, we will present AskBot's architecture, its implementation process, and the role of its main components. As demonstrated in Fig. 2 there are two main layers in AskBot's architecture: (1) the Back-End layer containing the core components of the architecture, namely: (1) Spell Correction, (2) Data Preprocessing, (3) Vectorization, (4) Similarity Measure, (5) Speech To Text, (6) Text To Speech, (7) Small Talk Verification (8) DialoGPT and (9) The knowledge base. They work together to produce the

appropriate answer for a given question. (2) the Front-End layer representing the graphical user interface (G.U.I). It facilitates the interaction between learners and AskBot. Thus, learners can directly ask their questions using its graphical interface.

AskBot's architecture addresses the problem of manually creating chatbots in the e-learning context either by manually constructing the local KB or by manually creating intents for training phrases in online platforms. Specifically, the present work proposes using the web scraping technique to automatically construct AskBot's KB. Hence, saving time and energy. Furthermore, based on our proposed python script, AskBot can detect users' needs without using prebuilt NLP models in online platforms, thus saving time, and energy, and reducing cost.

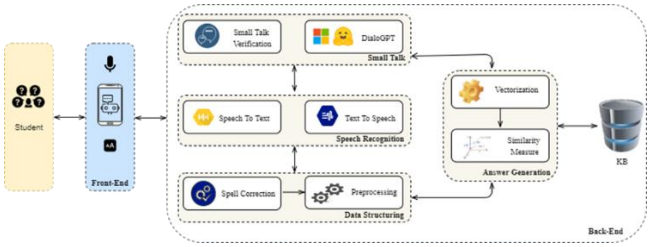


Fig. 2. Askbot's Architecture.

## B. Implementation Process

In this part, we will present the implementation process of AskBot's architecture. To demonstrate the feasibility of our approach, we focused our work on the Machine Learning domain. Thus, we develop an inclusive chatbot able to handle repetitive and massive questions related to the Machine Learning domain. Furthermore, AskBot's architecture can easily perform in different contexts by applying the same implementation process.

In Fig. 3 we present the methodology adopted to train AskBot. As demonstrated in Fig. 3 there are seven main phases in the implementation process of AskBot, namely: (1) Data Collection, (2) Data Pre-processing, (3) Word embedding, (4) Data Splitting, (5) Intent Classification, (6) Similarity Measures and (7) Answer Generation. In this section, we will deeply outline each phase.

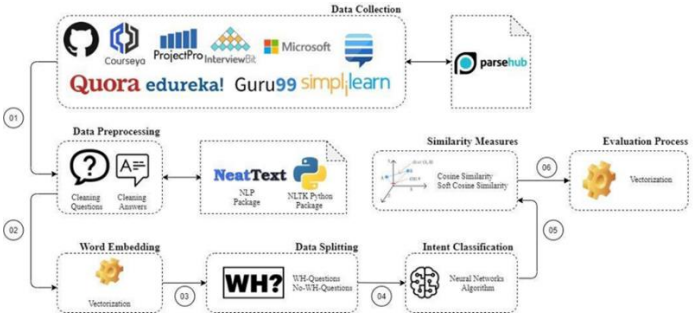


Fig. 3. Methodology adopted to develop AskBot.

*1) Data collection*: The proposed approach automatically collects pairs of Q&As from several online forums and educational sources providing reliable answers for ML questions, including: Interview Bit[1], Simplilearn[2], Edureka[3], Project Pro[4], Guru99[5], Coursera[6], and Quora[7]. Specifically, the present work includes 10075 Q&As retrieved through the ParseHub data extraction tool. It scraps and handles large amounts of data from several sources. However, our solution excludes 8784 of the Quora retrieved Q&As since the chatbot requires direct and concise answers whereas most of Quora's answers are subjective (ideas, experiences, ...), too long, and indirect. Furthermore, this work includes Q&As shared in the Stack Exchange network, especially, 19876 pairs of Q&As are integrated from three reliable sources: Cross Validated, data science, and Artificial Intelligence. They enrich AskBot's KB to 21167 Q&As.

Besides ML questions, the present work integrates Q&As about general conversations (Small Talk) to improve the chatbot's ability to appropriately interact when it receives queries about topics other than technical ones. For that reason, the proposed chatbot integrates an open-source dataset, from the GitHub site. It contains general Q&As, regarding the following topics: greetings, AI, computers, emotion, food, gossip, health, history, humor, literature, money, movies, politics, psychology, science, sports, and stories. Thus, AskBot can imitate human capability to talk about general topics. Furthermore, the chatbot integrates DialoGPT, a sizable pre-trained dialogue response-generating model, developed by MSR AI and the Microsoft Dynamics 365 AI Research team for small talk conversations. The model is created using 147M conversations from Reddit posts. The DialoGPT project lays the groundwork for creating adaptable Open Domain chatbots, able to answer engagingly and naturally a wide range of conversational subjects, tasks, and information requests. Experience demonstrates that the response produced by DialoGPT is comparable to the quality of human response (Zhang et al., 2019).

*2) Data preprocessing*: The present step focuses on applying a series of transformations to preprocess and clean the retrieved Q&As. Hence, successfully performing NLP models. As demonstrated in Fig. 4 there are six main steps in cleaning questions: (1) Remove duplicated rows (2) Delete numbers, and hashtags using the NeatText library (3) Replace punctuations with a space, except the dashes (- and _) since the dataset contains technical words separated by dashes such as the word "scikit-learn". In this case, replacing dashes leads to considering scikit and learn as two different words. The proposed solution is to concatenate these words rather than separate them (4) Remove stopwords: Many words in the English language, such as "I," "the," and "you," are used frequently in texts however they do not offer any significant

[1] https://www.interviewbit.com/
[2] https://www.simplilearn.com/
[3] https://www.edureka.co/
[4] https://www.projectpro.io/
[5] https://www.guru99.com/
[6] https://www.coursera.org/
[7] https://stackexchange.com/

information for NLP operations and modeling. Since it is highly recommended to eliminate them, the proposed approach removes stopwords to increase the efficiency and robustness of the NLP model (5) Fix contractions: A contraction is a word or group of words that has had one or more letters removed and replaced with an apostrophe. The proposed solution replaces contractions with the complete version of the word; For example, "what's" and "you've" become respectively "what is" and "you have". (6) Lemmatization: in this step, the proposed solution consists of converting any kind of word to its base root mode. It is highly recommended to apply Lemmatization when the meaning of the word is important for the analysis.

The second part of the data processing step is to clean responses by removing special characters and HTML tags. In Fig. 5 we present the word cloud for all stored questions in the dataset. As shown in the word cloud, the word occurs in the text more frequently the bigger and bolder it is in the word cloud. Thus, we can ensure that all questions in the dataset are around the Machine Learning domain.

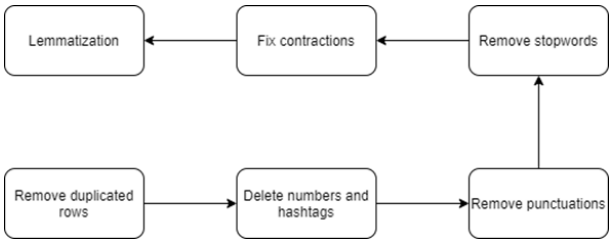


Fig. 4. Methodology adopted to preprocess the questions.

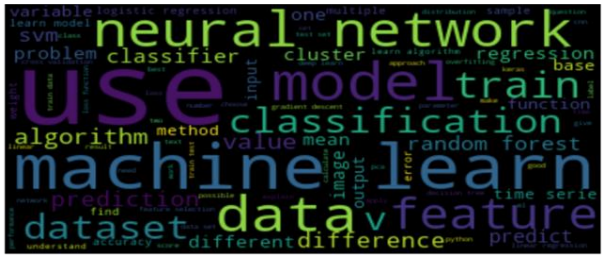


Fig. 5. Word cloud for all stored questions in the dataset.

*3) Word embedding*: The proposed approach focuses on four main word embedding techniques, especially:

- TF-IDF instead of Bag Of Words, because Bag of Words simply creates a set of vectors that contain the number of word occurrences in the document, while the TF-IDF model contains information about the most important and least important words as well. BOW vectors are easy to interpret. However, TF-IDF generally works better in machine-learning models (Pimpalkar and Raj, 2020).
- GloVe captures long-term interdependence by taking semantic meaning and word similarity into account during embedding (Chuah, 2022).
- Fasttext since it can consider the context in textual data and handle out-of-order words by n-gram models (Lestari and Setiawan, 2022).
- Word2vec has proven excellent performance in vectorizing words, and phrases, by producing one vector per word (Sharma and Kumar, 2023b).

Based on the Python Language, the proposed approach succeeds in (1) Developing the TF-IDF model able to vectorize Q&As in numerical vectors by following the TF-IDF process. (2) Developing the glove model by applying "Glove-wiki-Gigaword-50": a pre-trained glove model with no casing, based on 2 billion tweets, 27 billion tokens, and 1.2 billion words. (3) Developing the Fasttext model through the pre-trained model called "Fasttext-wiki-news-subwords-300" which uses one million words vectors and is trained on the statmt.org news dataset, UMBC web-based corpus, and Wikipedia 2017 (16B tokens). (4) Applying the word2vec model by using the "word2vec-google-news-300" that was trained on 100 billion words. It includes three million words and phrases represented by 300-dimensional vectors.

*4) Data splitting*: In this step, the proposed approach automatically classifies questions' intents to group similar questions into the same class. The chatbot's KB includes two types of questions: (1) WH-Questions starting with Wh words such as: what, how, when, why, which, who, and where. They represent almost 27.5% of the KB (2) "Non-Wh-Questions" beginning with different words, they represent almost 72.5% of the KB because most learners prefer asking their questions freely without starting with wh-words. Fig. 6 presents the distribution of WH-Questions and Non-WH-Questions in the KB.

Our work proposes a novel approach to analyze both Wh-Questions and Non-Wh-Questions. The chatbot's KB is divided into two main parts, and each part will be analyzed separately: (1) Dataset for Wh-Questions to automatically classify their intents since all questions start with "WH-words" and (2) Dataset for "Non-Wh Questions": because it is challenging to automatically classify them since they begin with unpredictable words.

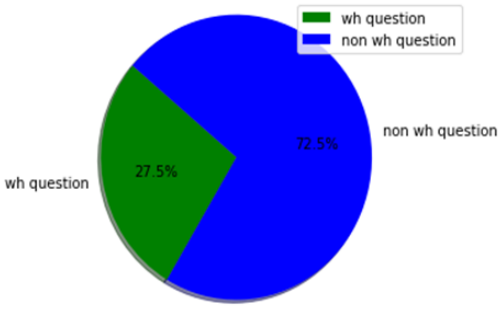


Fig. 6. Distribution of WH-questions and No-WH-questions in Q&As dataset.

*5) Intent classification*: Our proposed approach automatically classifies WH-Questions by developing a python script based on the following conditions:

- If the question starts with "what", the intent class will be: "Defintion_or_asking_for_information", since we noticed that the questions that start with "what", ask for either a definition or a piece of information. Example: what is supervised learning?
- If it starts with "why", the intent class will be "Reason". Example: Why does overfitting occur?
- If it starts with "How", the intent class will be “Method ". Example: How is the f1 score used?
- If it starts with "Who", the class will be "Person". Example: Who invented the concept of overfitting?
- If it starts with "Which ", the intent class will be “Choice ". Example: which is better lstm or gru?
- If it starts with "Where", the intent class will be "Source_or_case", since the questions that contain where ask for either a source or an explanation about a specific case. Example: Where do predictions depend on?
- If it starts with "When", the intent class will be "Situation_to_use_or_to_happen", since the questions start with when asking when to use a situation and when it occurs. Example: When to use group Learning?

As demonstrated by the distribution of intent classes (Fig. 7) the dataset of WH-Questions is unbalanced. Histograms show that most questions fall into two classes: "Method" with 2944 questions and "Definition or asking for information" with 2092 questions. In addition to that, 260 questions refer to "Situation to use or to happen," 387 questions pertain to "Choice," and 873 questions refer to "Reason." However, just a few questions from the class "Source or case" with 51 questions and "Person" with four questions.

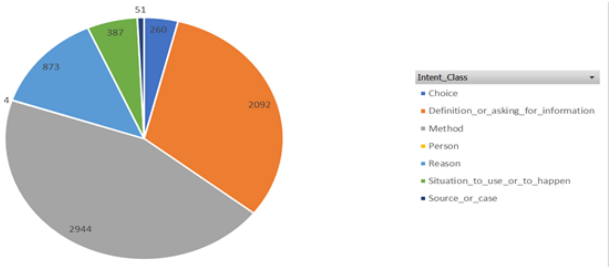


Fig. 7. Distribution of intent classes.

The intent classification step aims to predict the intent of new wh-questions. For that reason, our approach includes a predictive model to forecast the appropriate intent class for a given wh-question. Especially, there are various models to perform the intent classification task, such as K nearest neighbors, Naïve Bayes, Support Vector Machine, Decision Tree, Random Forest, etc. Our proposed architecture adopts Neural network algorithm (NN) because of two main reasons claimed by [35] : (1) it needs less formal statistical training (2) it has a high capacity to detect complex nonlinear relationships between dependent and independent variables, in our case, independent variables refer to wh-questions and the intent class is the dependent variable. The first task is to vectorize wh-questions by using word embedding techniques. Then, develop the neural network (NNs) model that predicts the corresponding intent class in WH-Questions. For that reason, we divide the Wh-Question dataset into two parts: The first one is training data (80% of the dataset) to train the NNs model and the second one is testing data to test the pre-trained model and evaluate its accuracy (20% of the dataset). Then, we use the confusion matrix to evaluate the performance of the NNs model. As demonstrated in Fig. 8 the confusion matrix resumes the intents classification task. Predictions are distinguished by classes and contrasted with real values. Good predictions are presented in the diagonal of the matrix. For example [Method, Method] = 569 which means that 569 questions in the test data present the method intent class, and they were well classified by the NNs model. As demonstrated by Fig. 8 a few questions were badly classified: For example [Reason, Method] = 2 which means that just two questions that present method class, have been classified as Reason class. Thus, we can conclude that the NNs model fits well the Q&As in the test dataset.

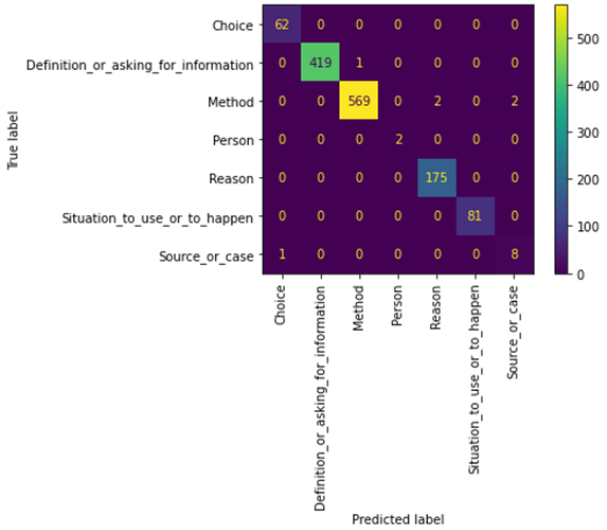


Fig. 8. Confusion matrix of NN model.

Additionally, we display the classification report as a performance evaluation metric to measure the model’s performance. Especially, it presents the classification model's precision, recall, F1 score, and accuracy.

Recall for the class Definition = 1 (419 / 419). Only the classification of the definition intent is important for the recall of the class definition. This doesn't depend on how the other intents are classified. Even if the model mistakenly identified all other intents as definition, the recall will be 100% when it classifies all the definition intent questions as Definition. Precision for the class Definition = 0.9976 (419/420). Many incorrect (Or few correct) classifications for the Definition class lead to low precision. The precision for the Definition class shows how accurate the model is in identifying learners’ questions related to the definition class.

Recall and precision are complementary; they measure the model's performance related to a specific intent. However, the model's performance related to all classes is measured through the accuracy or F1 score, they describe the model performance in forecasting questions' intents. Accuracy calculates the number of times the model was correct overall and (ii) F1 score assesses a model's performance by combining the model's precision and recall scores.

As demonstrated by Fig. 9 the classification report demonstrates that the model's accuracy and F1-Score are respectively equal to 0.995, 0.97. It means that our NNs model fits well the dataset of WH-Question, i.e., the model can distinguish well between the classes. Thus, we can conclude that the model is reliable; it is able to make good predictions.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Choice | 1.00 | 0.98 | 0.99 | 63 |
| Definition_or_asking_for_information | 1.00 | 1.00 | 1.00 | 419 |
| Method | 0.99 | 1.00 | 1.00 | 570 |
| Person | 1.00 | 1.00 | 1.00 | 2 |
| Reason | 1.00 | 0.99 | 0.99 | 177 |
| Situation_to_use_or_to_happen | 1.00 | 1.00 | 1.00 | 81 |
| Source_or_case | 0.89 | 0.80 | 0.84 | 10 |
| accuracy | | | 1.00 | 1322 |
| macro avg | 0.98 | 0.97 | 0.97 | 1322 |
| weighted avg | 1.00 | 1.00 | 1.00 | 1322 |

accuracy: 0.9954614220877458

Fig. 9. Classification report of NN model.

*6) Similarity measures*: The proposed architecture calculates similarity measures to retrieve the most similar question to the learner's question. Then, send its response to the user as the appropriate response. In the case of Wh-Question, AskBot vectorizes the learner's question through word embedding techniques, predicts the learner's intent, then applies cosine similarity. Regarding Non-Wh-Questions, it is difficult to classify the intent class because they start with unpredictable words. For that reason, the proposed solution uses the Soft Cosine Measure to find the appropriate answer directly without classifying their intentions. Soft Cosine Measure is most appropriate for Non-WH-Questions because it can find questions that are related even when they do not share any words. After evaluating several similarity scores, we find that the score of 0.6 has shown satisfactory results in extracting similar questions to the learners' questions. For that reason, we adopt that score as a threshold of similarity. When a user asks a question, we calculate the similarity between the user question and all questions in the local KB, if the similarity is greater than 0.6, then, we send the response of the similar question to the user. Specifically, we choose the question with the highest score of similarity.

*7) Evaluation process*: The proposed approach includes 100 pairs of Q&As related to the ML domain to evaluate each vectorization model. As presented in Fig. 10 for each question in the test dataset, we applied the preprocessing steps and the vectorization models. Then, there are two main cases: (1) Relating WH-question, the proposed solution is to predict the intent class by applying the NNs model, then applying cosine similarity to retrieve the most similar question to the test question. And finally, retrieving the stored answer to the matched question. (2) Regarding the No-WH-Questions, the proposed approach is to apply the soft cosine similarity to retrieve the most similar question with its response.

The proposed approach includes 100 pairs of Q&As related to the ML domain to evaluate each vectorization model. Specifically, for each question in the test dataset, we applied the preprocessing steps and the vectorization models. Then, there are two main cases: (1) Relating WH-question, the proposed solution is to predict the intent class by applying the NN model, then applying cosine similarity to retrieve the most similar question to the test question. And finally, retrieving the stored answer to the similar question. (2) Regarding the Non-WH-Questions, the proposed approach is to apply the soft cosine similarity to retrieve the most similar question with its response.

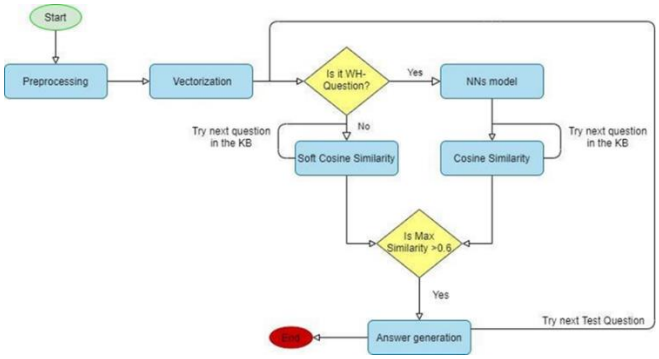


Fig. 10. Flowchart of the evaluation process.

In the evaluation process, we create a dataframe that summarizes the test results. Next, we evaluate the performance of the vectorization methods. Sometimes, even if the scores similarity is high, we find that questions are not very similar. Thus, we cannot automatically evaluate the test results based on the score similarity. Therefore, three team members examine the results to retrieve the number of correct answers in each vectorization method. Each member scores the generated answers by reporting 1 if the questions are similar and 0 otherwise. Since the dataset used in the evaluation process contains Wh and Non-Wh questions, we calculate the percentage of correctly answered questions for each type according to each vectorization technique.

As shown in Fig. 11 all vectorization techniques have demonstrated good results regarding WH-Questions. Specifically, TF-IDF succeeds in responding to 90% of WH-Questions. Based on Fig. 11 we considered TF-IDF as the most appropriate method for WH-Questions for two main reasons: (1) 90% of WH-Questions were well classified. (2) Just 10% of WH-Questions were badly classified, which demonstrates that there is a low chance to make bad classifications for new WH-Questions. Based on Fig. 12, both TF-IDF and Fasttext demonstrated good results in 70%, and 72% of questions respectively. Thus, we adopted Fasttext for Non-WH-Questions.

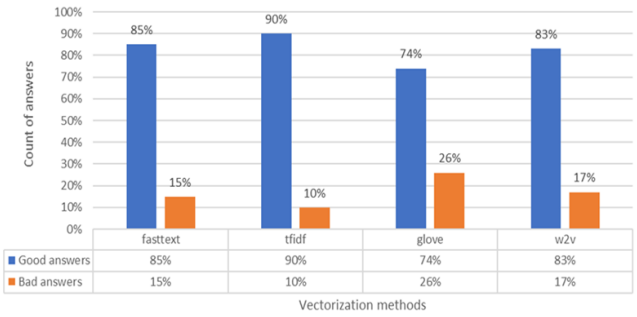


Fig. 11. Comparison between the performances of the vectorization methods in WH-Questions.

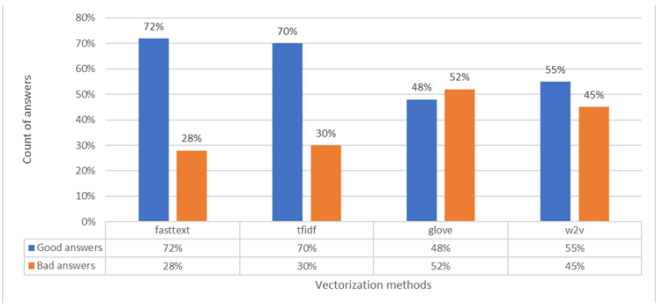


Fig. 12. Comparison between the performances of the vectorization methods in NO-WH-Questions.

### C. *Architecture's Components*

In this section, we will present the main components of AskBot's architecture and the generated information flow to answer learners' questions. In Fig. 13 we present a flowchart diagram that explains how the chatbot's components interact to answer a given question.

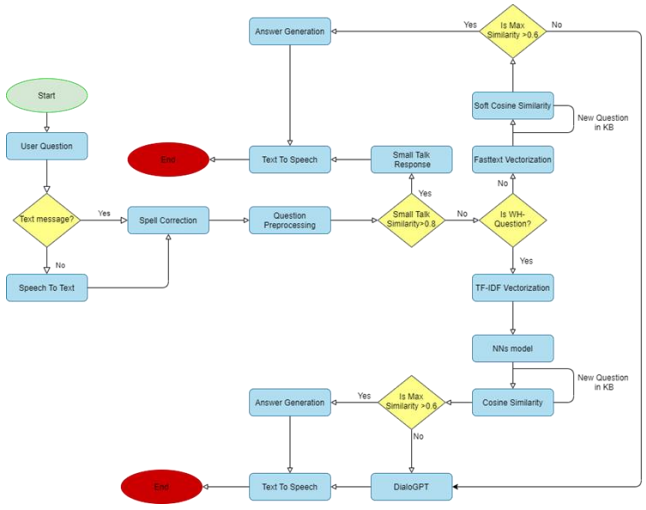


Fig. 13. Flowchart of the Askbot's architecture.

#### 1) Back-end

*a) Spell Correction*: One common task in automatic NLP is spelling correction for many NLP applications, including Web search engines, text summarization, sentiment analysis, etc. When the user misspells search terms or the user's message differs from the spelling in the local KB, the Automatic Spelling Correction function enables search queries to produce the intended results by correcting and rectifying misspelled words through a suggested set of terms that are the closest lexically to the incorrect ones (Hládek, Staš and Pleva, 2020). Python proposes a variety of modules and packages to carry out spell correction such as textblob, pyspellchecker, and JamSpell. However, these Python tools are not specifically designed for correcting technical words in the ML field. For that reason, our approach proposes the my_autocorrect Python function that takes a word as input and outputs a list of terms that are related to it.

First, we start by extracting 10068 keywords from Q&As in AskBot's KB, next, we store them in a python dictionary. Then, we check if a given word exists in the dictionary, thus, the word is correct. Otherwise, we calculated the similarity between the given word and all words in the dictionary. Specifically, after testing several similarity scores, we find that the score of 0.5 has shown good results in retrieving the most similar words. For that reason, we adopt that score as a threshold of similarity. We keep words with similarity scores higher than the threshold of 0.5. And finally, we replace the given word with the word that has the highest similarity score. In Fig. 14 we present two use cases of the developed function "my_autocorrect". In the first example: my_autocorrect('baggng'), the misspelled word 'baggng' will be replaced by the word "bagging" since it represents the highest score similarity (=0.571429). In the other example, the word 'classfer' will be replaced by "classifier" as the most similar word to 'classfer'.

```
my_autocorrect('baggng')
```

| | Word | Prob | Similarity |
|---|---|---|---|
| 2202 | bagging | 0.000133 | 0.571429 |
| 3170 | bag | 0.000133 | 0.400000 |
| 67 | laggy | 0.000133 | 0.285714 |
| 187 | bagof | 0.000133 | 0.285714 |
| 1104 | engage | 0.000133 | 0.250000 |

```
my_autocorrect('classfer')
```

| | Word | Prob | Similarity |
|---|---|---|---|
| 5525 | classifer | 0.000133 | 0.666667 |
| 6939 | classfiers | 0.000133 | 0.600000 |
| 420 | class | 0.000133 | 0.571429 |
| 953 | classier | 0.000133 | 0.555556 |
| 1307 | perclass | 0.000133 | 0.555556 |

Fig. 14. Spell correction function.

*b) Data Preprocessing*: It is the component responsible for cleaning the user's input to transform it into a structured format by applying the data preprocessing steps.

*c) Vectorization*: It is the component that vectorizes the user's input, it takes the output of the "Data Preprocessing component" and verifies if the user's question is a WH-Questions, then, it applies TF-IDF to vectorize it. Otherwise, it applies Fasttext.

*d) Prediction Model NNs*: It is the model that predicts the intent class for WH-Questions asked by the user. It uses the output of the vectorization component to predict the user's intention.

*e) Similarity Measure*: It is the component responsible for generating the appropriate answer for a given question, it verifies the question's type. When the asked question is a WH-Question, then, it uses the intent class and the cosine similarity score to retrieve the most similar question to the user's

question, and finally, it sends the response of the similar question as the appropriate response. Otherwise, if the user's question is a NO-WH-Question, then, it applies the Soft Cosine Similarity to generate the answer.

*f) Small Talk Verification*: It is the component responsible for analyzing the user's message and verifying whether the message falls into small talk data or not. It aims to handle small talk conversation by applying the cosine similarity between the user and the small talk data. if the user's question is similar to a small talk question. (i;e the similarity between the user's question and questions in the small talk data is greater than 0.8). Then, the response will be extracted from small talk data (See Fig. 6) rather than searching in the local kb. We used a high score similarity to ensure that the user's question falls into general conversation and not a technical question.

*g)* DialoGPT: It is the component responsible for generating answers if the user's questions do not match with the stored Q&As in the local kb. After applying the similarity measures, if the similarity score between the user's question and the KB does not exceed the threshold of 0.6. Then, the answer is generated from the DialoGPT component. The small talk and DialoGPT components aim to make the chatbot richer and friendlier. It can interact with learners freely and respond to general questions beyond its local kb. Thus, enhancing the user experience.

*h) Speech Recognition*: It is the component responsible for interacting with voice messages. It aims to make the chatbot more inclusive. Thus, disabled learners can use it to find the information they need. There are two main tasks to perform speech recognition:(1) Speech-To-Text (STT): It is an AI technology that converts voice content into text. It enables a computer program to take a human voice and convert it into text. (2) Text-To-Speech (TTS): It is an AI technology that aims to convert text into voice. It provides the opposite of speech-to-text software's ability to turn voice content into text. The AskBot's architecture supports learners, especially those with disabilities, by efficiently handling their voice queries through SpeechToText and TextToSpeech techniques. Hence, our proposed chatbot is inclusive.

*2)* Front-end

*a) Graphical user Interface*: In order to facilitate the use of the chatbot, we developed a Graphical User Interface (G.U.I).Thus, learners can use it to interact with the chatbot by sending text/voice messages. We used the package "Tkinter" to develop the chatbot interface. It is a standard graphical library for Python to create a GUI-based application.

As demonstrated in Fig. 15 the interface includes a frame for conversation, an input field for the user to enter a message, and two buttons. The first one is labeled "send," to enable sending a message, and the second one enables the STT technology. Students can send messages through two methods: (1) Entering text messages in the input field. (2) Pressing on the "micro" icon to send voice messages.

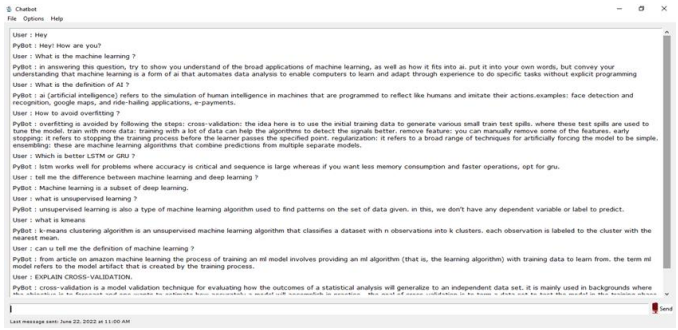


Fig. 15. Spell Use case of AskBot.

## V. CONCLUSION

In this paper, we proposed an inclusive chatbot able to respond to users' questions about the Machine Learning field. It helps learners in their learning processes by providing reliable information the learners need without wasting time searching in many available sources. The present work proposes a smart learning architecture that supports various AI technologies and aims specially to automate answering massive and repetitive questions asked by learners. AskBot offloads tutors from doing repetitive tasks and helps them concentrate and focus their energies on tasks that need more concentration and effort. Furthermore, it converts text to speech and speech to text to facilitate the learning process. Thus, the designed chatbot is inclusive, it can be used by disabled students using speech recognition technology. Disabled students can easily ask their questions by sending vocal messages and receiving their responses in voice format. In addition to that, our chatbot can recognize small talk messages and handle general conversation by using small talk data and the DialoGPT model. Thus, the chatbot adapts its response according to the student's requests. Hence, it encourages learners to interact freely with it and makes them more engaged and motivated. Although this work provides valuable insights to save time and energy needed in chatbot's creation process, it is important to acknowledge some limitations that could be covered by other researchers. Especially, (1) the current version AskBot is not evolutive because the local KB is still limited to already stored knowledge. (2) More advanced models could be tested to all questions without separating them to Wh-Questions and No-Wh-Questions. In future works, we plan to address these limitations and add more advanced features such as (1) Sentiment analysis to recognize the learner's emotions and personalized the chatbot's responses. (2) Deep learning models to create generative chatbots and test their ability to answer students' questions. (3) Distributed storage for the chatbot kB to enhance the speed system and its performance (4) integrate AskBot in the existing Learning Management Systems (LMS).

## REFERENCES

[1] E. Adamopoulou and L. Moussiades, "Chatbots: History, technology, and applications," Mach. Learn. Appl., vol. 2, p. 100006, Dec. 2020, doi: 10.1016/j.mlwa.2020.100006.

[2] S. Meshram, N. Naik, V. Megha, T. More, and S. Kharche, "College enquiry chatbot using RASA framework," presented at the 2021 Asian Conference on Innovation in Technology (ASIANCON), IEEE, 2021, pp. 1–8.

[3] H. T. Hien, P.-N. Cuong, L. N. H. Nam, H. L. T. K. Nhung, and L. D. Thang, "Intelligent assistants in higher-education environments: the FIT-

EBot, a chatbot for administrative and learning support," presented at the Proceedings of the 9th International Symposium on Information and Communication Technology, 2018, pp. 69–76.

[4] E. V. Kuanishbaevna, "The Role of the Teacher in Teaching Students in Accordance with National Values," Eur. J. Bus. Startups Open Soc., vol. 2, no. 1, pp. 79–80, 2022.

[5] W. Ahmed and B. Anto, "AN AUTOMATIC WEB-BASED QUESTION ANSWERING SYSTEM FOR E-LEARNING," Inf. Technol. Learn. TOOLS, vol. 58, no. 2, pp. 1–10, 2017.

[6] E. Kasthuri and S. Balaji, "A Chatbot for Changing Lifestyle in Education," in 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), Feb. 2021, pp. 1317–1322. doi: 10.1109/ICICV50876.2021.9388633.

[7] R. Andersen, A. I. Mørch, and K. T. Litherland, "Collaborative learning with block-based programming: investigating human-centered artificial intelligence in education," Behav. Inf. Technol., vol. 41, no. 9, pp. 1830–1847, 2022.

[8] A.-J. Moreno-Guerrero, J.-A. Marín-Marín, P. Dúo-Terrón, and J. López-Belmonte, "Chatbots in Education: A Systematic Review of the Science Literature," Artif. Intell. High. Educ., pp. 81–94, 2023.

[9] S. Abbasi and H. Kazi, "Measuring effectiveness of learning chatbot systems on student's learning outcome and memory retention," Asian J. Appl. Sci. Eng., vol. 3, no. 2, pp. 251–260, 2014.

[10] N. Mutovkina, "Digital Technologies in the Educational Process and the Effectiveness of Their Use," in Advances in Intelligent Systems, Computer Science and Digital Economics IV, Springer, 2023, pp. 937–946.

[11] K. El Azhari, I. Hilal, N. Daoudi, and R. Ajhoun, "Chatbots in E-learning: Advantages and Limitations," presented at the Colloque sur les Objets et systèmes Connectés-COC'2021, 2021.

[12] P. Smutny and P. Schreiberova, "Chatbots for learning: A review of educational chatbots for the Facebook Messenger," Comput. Educ., vol. 151, p. 103862, Jul. 2020, doi: 10.1016/j.compedu.2020.103862.

[13] M. A. Calijorne Soares, W. Cardoso Brandão, and F. Silva Parreiras, "A Neural Question Answering System for Supporting Software Engineering Students," in 2018 XIII Latin American Conference on Learning Technologies (LACLO), Oct. 2018, pp. 201–207. doi: 10.1109/LACLO.2018.00047.

[14] M. Verleger and J. Pembridge, "A Pilot Study Integrating an AI-driven Chatbot in an Introductory Programming Course," in 2018 IEEE Frontiers in Education Conference (FIE), Oct. 2018, pp. 1–4. doi: 10.1109/FIE.2018.8659282.

[15] M. Kowsher, F. S. Tithi, M. Ashraful Alam, M. N. Huda, M. Md Moheuddin, and M. G. Rosul, "Doly: Bengali Chatbot for Bengali Education," in 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), May 2019, pp. 1–6. doi: 10.1109/ICASERT.2019.8934592.

[16] H. A. Rasheed, J. Zenkert, C. Weber, and M. Fathi, "Conversational chatbot system for student support in administrative exam information," presented at the ICERI2019 Proceedings, IATED, 2019, pp. 8294–8301.

[17] P. A. Tamayo, A. Herrero, J. Martín, C. Navarro, and J. M. Tránchez, "Design of a chatbot as a distance learning assistant," Open Prax., vol. 12, no. 1, pp. 145–153, 2020.

[18] A. T. Neumann et al., "Chatbots as a tool to scale mentoring processes: Individually supporting self-study in higher education," Front. Artif. Intell., vol. 4, p. 668220, 2021.

[19] Y. S. Li, C. S. N. Lam, and C. See, "Using a machine learning architecture to create an AI-powered chatbot for anatomy education," Med. Sci. Educ., vol. 31, pp. 1729–1730, 2021.

[20] K. Nhut Lam, N. Nhat Le, and J. Kalita, "Building a Chatbot on a Closed Domain using RASA," ArXiv E-Prints, p. arXiv-2208, 2022.

[21] S. Singh and S. Singh, "Effective Analysis of Chatbot Frameworks: RASA and Dialogflow," EasyChair, 2516–2314, 2022.

[22] D. Carlander-Reuterfelt, A. Carrera, C. Iglesias, O. Araque, J. Sanchez-Rada, and S. Munoz, "JAICOB: A Data Science Chatbot," IEEE ACCESS, vol. 8, pp. 180672–180680, 2020, doi: 10.1109/ACCESS.2020.3024795.

[23] B. Göschlberger and C. Brandstetter, "Conversational AI for Corporate E-Learning," in Proceedings of the 21st International Conference on Information Integration and Web-Based Applications & Services, in iiWAS2019. New York, NY, USA: Association for Computing Machinery, 2019, pp. 674–678. doi: 10.1145/3366030.3366115.

[24] A. S. Sreelakshmi, S. B. Abhinaya, A. Nair, and S. Jaya Nirmala, "A Question Answering and Quiz Generation Chatbot for Education," in 2019 Grace Hopper Celebration India (GHCI), Nov. 2019, pp. 1–6. doi: 10.1109/GHCI47972.2019.9071832.

[25] Y. Sumikawa, M. Fujiyoshi, H. Hatakeyama, and M. Nagai, "An FAQ dataset for E-learning system used on a Japanese University," Data Brief, vol. 25, p. 104001, Aug. 2019, doi: 10.1016/j.dib.2019.104001.

[26] I. Bohomolova, N. Kushnir, and S. Moshkovska, "Using Chatbots to Approach Individual Learning Trajectories in Physics for Foreign Students," in CEUR Workshop Proceedings, 2021, pp. 253–263. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85121650753&partnerID=40&md5=5a7f89df4f4b10e49c20c8df2e9cfcf5

[27] C. Chun Ho, H. L. Lee, W. K. Lo, and K. F. A. Lui, "Developing a Chatbot for College Student Programme Advisement," in 2018 International Symposium on Educational Technology (ISET), Aug. 2018, pp. 52–56. doi: 10.1109/ISET.2018.00021.

[28] D. Carlander-Reuterfelt, Á. Carrera, C. A. Iglesias, Ó. Araque, J. F. Sánchez Rada, and S. Muñoz, "JAICOB: A Data Science Chatbot," IEEE Access, vol. 8, pp. 180672–180680, 2020, doi: 10.1109/ACCESS.2020.3024795.

[29] N. M. Deepika, M. M. Bala, and R. Kumar, "Design and implementation of intelligent virtual laboratory using RASA framework," Mater. Today Proc., Feb. 2021, doi: 10.1016/j.matpr.2021.01.226.

[30] D. E. Gonda and B. Chu, "Chatbot as a learning resource? Creating conversational bots as a supplement for teaching assistant training course," presented at the 2019 IEEE International Conference on Engineering, Technology and Education (TALE), IEEE, 2019, pp. 1–5.

[31] H. Steinbeck, T. E. I. Zobel, and C. Meinel, "Towards leveraging conversational agents for instructors and learners to find and access learning resources," in 2021 World Engineering Education Forum/Global Engineering Deans Council (WEEF/GEDC), Nov. 2021, pp. 607–611. doi: 10.1109/WEEF/GEDC53299.2021.9657307.

[32] S. Meshram, N. Naik, V. Megha, T. More, and S. Kharche, "College enquiry chatbot using RASA framework," presented at the 2021 Asian Conference on Innovation in Technology (ASIANCON), IEEE, 2021, pp. 1–8.

[33] A. T. Neumann et al., "Chatbots as a Tool to Scale Mentoring Processes: Individually Supporting Self-Study in Higher Education," Front. Artif. Intell., vol. 4, 2021, doi: 10.3389/frai.2021.668220.

[34] L. Wang and W. Wang, "Research and Construction of Junior High School Subject Q&A System Model based on Deep Learning," in 2018 International Conference on Information Systems and Computer Aided Education (ICISCAE), Jul. 2018, pp. 504–508. doi: 10.1109/ICISCAE.2018.8666853.

[35] S. Liang and R. Srikant, "Why deep neural networks for function approximation?," ArXiv Prepr. ArXiv161004161, 2016.