

Academic assistance chatbot-a comprehensive NLP and deep learning-based approaches

Sara H. Anwar¹, Karim M. Abouaish¹, Emil M. Matta¹, Amr K. Farouq¹, Ahmed A. Ahmed¹,
Nermin K. Negied^{1,2}

¹School of Engineering and Applied Science, Nile University, Giza, Egypt

²Department of Electronics and Computer Engineering, School of Engineering and Applied Science, Giza, Egypt

Article Info

Article history:

Received Oct 22, 2023

Revised Nov 17, 2023

Accepted Dec 9, 2023

Keywords:

Chatbot

Deep learning

Job matching

Question answering

Sentiment analysis

Summarization

Text paraphrasing

ABSTRACT

The rapid growth of digital technologies and natural language processing (NLP) have revolutionized the field of education, creating new demand for automated academic assistance systems. In this paper, we present an NLP-based academic assistance chatbot designed to provide comprehensive support to students and researchers using deep learning techniques. The chatbot incorporates a range of intelligent features to assist with university recommendations, article writing, automatic question answering (QA), and job search. By leveraging sentiment analysis and sarcasm detection models. The proposed chatbot could offer accurate and insightful university recommendations. Additionally, the chatbot incorporates spell and grammar checking, summarization, paraphrasing, and topic modeling capabilities to aid users in enhancing their writing skills. The QA module enables users to obtain quick and precise answers to factoid-based questions. Moreover, the chatbot helps with internships and job search. According to literature, this work presents the first assistance chatbot that encapsulates all features that may be needed by a university student to facilitate and improve his/her learning process. The results demonstrated clearly in the body of the paper showed the success achieved by the academic assistant proposed and built in this work in all its features or modules to offer help to university students and graduates.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Nermin K. Negied

Department of Electronics and Computer Engineering, School of Engineering and Applied Science

Giza, Egypt

Email: nnegied@zewailcity.edu.eg, nnegied@nu.edu.eg

1. INTRODUCTION

University students encounter a range of challenges and obstacles that can impede their academic progress. These challenges include the difficulty of accessing reliable and pertinent information necessary for making informed decisions, such as selecting the right university. Students may lack access to comprehensive databases, resources, or expert guidance, which can hinder their ability to make well-informed choices. Additionally, language barriers and skill gaps pose significant difficulties, particularly for non-native English speakers or individuals lacking essential communication and presentation skills. These limitations can hinder effective communication, hindering students' ability to effectively express their abilities and achievements. Furthermore, time constraints and heavy workloads are common issues for university students. Balancing academic commitments with personal responsibilities like job seeking and curriculum vitae (CV) enhancing can be challenging, leaving students with limited time and energy to pursue their academic and career aspirations fully. Meanwhile, during the last period the field of natural language processing (NLP) has

witnessed remarkable advancements, leading to its widespread application in various domains. Notable developments include the introduction of transformer models, pretrained language models, transfer learning approaches, multimodal NLP, and efforts towards ethical considerations and bias mitigation. These advancements have revolutionized tasks such as language understanding, generation, and translation, expanding the capabilities of NLP in areas like virtual assistants, chatbots, and sentiment analysis. The ongoing progress in NLP continues to push the boundaries of natural language understanding and generation (NLU and NLG), creating new opportunities and challenges for researchers and practitioners in the field. One prominent area of application is academic assistance, where NLP techniques can play a pivotal role in supporting students and researchers. Recognizing the challenges facing those University students in their academic and career paths while putting into consideration how powerful NLP field has become, the “academic assistance chatbot” proposed in this work leverages the power of artificial intelligence and NLP techniques to offer a comprehensive solution. The chatbot serves as a personalized virtual assistant, providing tailored support and guidance to students. It is worth mentioning that assistance chatbots in literature are normally doing one task, like question answering (QA) chatbots, recommendation chatbots, job matching chatbots, etc. This work leverages NLP capabilities to allow the proposed chatbot to handle seven different tasks which are: university recommendations, spell and grammar checking, summarization, paraphrasing, topic modeling, job and intern matching and QA. The goal is to empower students and job seekers, mitigating their challenges and helping them navigate their academic and career paths with confidence and success. The proposed “academic assistance chatbot”, is built to provide the student with a comprehensive set of features aimed at enhancing the academic journey of him/her.

The rest of this paper is organized as follows: section 2 reviews the related work done in literature. Section 3 delves into the various NLP techniques employed to implement each feature, highlighting the specific technique chosen for each, and providing a rationale behind its selection. Section 4 demonstrates the experiments held to test this work and discusses the results using some use cases to explain the final implementation of the chatbot, demonstrating its capabilities and effectiveness in supporting users in their academic pursuits. Finally, the paper is concluded in section 5.

2. LITERATURE REVIEW

This section provides a comprehensive demonstration of existing research and scholarly works pertaining to academic assistance and NLP. Through systematic review of the literature, it was found that chatbots that are built for student’s assistance are rare and do not include many features. The most recent student’s assistance chatbot found in literature was an intelligent tutoring system built using NLP ontology in 2021 to assist school students in their learning journey [1]. Due to the rareness of similar chatbots, we surveyed the literature to investigate the key approaches, methodologies, and applications that can be used to build the features or the main modules of our academic assistance chatbot. The following subsections explore the literature work done in every domain (feature) of the proposed system.

2.1. University reviews/recommendation system

To provide students with authentic and reliable university reviews, a sentiment analysis model is employed for this feature. It is crucial to consider the potential presence of sarcasm in such reviews which is not an easy task, but it is very important because it is a common characteristic in students’ reviews. Sentiment analysis is a very common application in NLP that aims to automatically classify the sentiment or personal attitude in a piece of text, such as a tweet, review, or news article. Sentiment analysis has a lot of real-world applications, such as market research, customer service, political analysis, and brand management. In the last few years, researchers invested a lot of effort in developing automated techniques for sentiment analysis, including the valence aware dictionary and sentiment reasoner (VADER) method and text classification methods. The VADER method combines lexicon and rule-based approaches to analyze sentiment [2]. It was that firstly developed by researchers at the University of Georgia. VADER uses a pre-built dictionary of sentiment words that are scored on a scale from -4 for most negative to +4 for most positive. Text classification is another important approach to address the problem of sentiment analysis that involves training a machine learning algorithm to classify text into positive, negative, or neutral categories based on labeled data. Text classification algorithms like Naïve Bayes (NB), decision trees (DT), support vector machines (SVM), and neural networks (NN) are very good in this task. They have several advantages, including the ability to handle domain-specific language and context, the ability to learn from data and adapt to changing trends, and the ability to provide interpretable results that can be easily understood and acted upon. Baizal *et al.* [3] used text classification algorithms along with NLP techniques to develop a movie recommendation system, and they confirmed that they achieved an accuracy rate that reached 83%.

2.2. Text paraphrasing

Text paraphrasing is a crucial task in NLP that aims to generate alternative versions of a given sentence or phrase while preserving its original meaning [4]. Paraphrasing plays a significant role in various NLP applications, like summarization, QA, machine translation, and information retrieval. Over the years, many approaches were proposed and studied to address the challenge of text paraphrasing. Traditional methods often relied on rule-based or template-based techniques, which had limited coverage and struggled to understand the semantics and meanings of the text [5]. However, with the advent of deep learning and neural networks, the field of text paraphrasing has witnessed significant advancements. Recently, the pre-training with extracted gap-sentences for abstractive summarization (PEGASUS) model has emerged as a state-of-the-art approach for text paraphrasing [6]. PEGASUS builds upon the transformer architecture and utilizes a pre-training and fine-tuning paradigm to generate high-quality paraphrases.

2.3. Text summarization

Considering text summarization, it aims to automatically generate concise summaries from large amounts of textual data [7]. It has numerous applications, such as news understanding and summarization, running text summarization, information retrieval [8], and social media summarization. Over the years, various approaches have been proposed to address the challenges of text summarization. One popular approach is extractive summarization, which involves selecting and concatenating key sentences or phrases from the source text to form a summary [9]. This method relies on identifying the most important and representative information in the original text. However, extractive summarization may result in disjointed sentences and lacks coherence. Another approach is abstractive summarization, which generates summaries by paraphrasing and rephrasing the content of the source text in a more human-like manner [10]. This technique allows for more flexibility and coherency in the generated summaries. Abstractive summarization often employs advanced natural language generation models, such as attention models or transformers, convolutional neural networks (CNNs), and recurrent neural networks (RNNs). Recently, transformer-based models have proven to be the best summarization models, as they outperformed the results obtained by all other machine learning and deep learning models. One prominent model which is found to outperform all preceding models in the summarization task is bidirectional and auto regressive transformers (BART) [6].

2.4. Spell/grammar check

Grammar and spell correction play a very important role in improving the quality and completeness of written text. Correcting grammar and spelling errors is vital for various NLP applications, including text summarization, machine translation, text generation, and information retrieval. Rule-based approaches employ predefined grammar rules and linguistic patterns to detect and correct errors [10]. These methods are effective in capturing simple grammatical errors but may struggle with complex structures and context-dependent errors. Statistical approaches, on the other hand, utilize annotated corpora and statistical models to identify and correct errors [11], [12]. These methods analyze word frequencies and language patterns to estimate likely corrections, offering improvements in handling context-dependent errors. However, statistical approaches may not capture intricate language nuances. Deep learning models, such as RNNs and transformer models, have shown remarkable performance in capturing complex dependencies and generating accurate corrections [13], [14]. These approaches learn from large amounts of data, effectively model context, and have significantly advanced grammar and spell correction. Hybrid approaches combine rule-based, statistical, and machine learning techniques to overcome limitations, providing a comprehensive solution [15]. These approaches have evolved over time, each offering unique advantages and limitations, with ongoing research focused on enhancing grammar and spell correction systems.

2.5. Topic modeling

Students may have some long articles or research papers that they want to know its context by having a more explicit title or topic for it. Moreover, they can write articles themselves and want a descriptive title for it. Here comes the role of topic modeling feature. Recently, topic modeling grabbed attention of researchers, particularly using matrix factorization and probabilistic modeling approaches. Latent dirichlet allocation (LDA) is a widely used probabilistic model for topic modeling that was introduced by Blei *et al.* [16] in 2003. LDA represents each document as a mixture of latent topics and represents each topic as a probability distribution over the words in the vocabulary. The goal of LDA is to learn the latent topics and their associated word distributions from the corpus of documents. Non-negative matrix factorization (NMF) is another widely used technique for topic modeling that was introduced by Lee and Seung [17]. The main target of NMF is to learn the latent topics and their associated word distributions from the corpus of documents by minimizing the difference or the error between the original matrix and the product of the

two factor matrices. Several studies have compared the performance of LDA and NMF in topic modeling and found that both approaches have their strengths and weaknesses. In addition to LDA and NMF, there are several other topic modeling techniques that have been developed in recent years, such as probabilistic latent semantic analysis (PLSA) [18], latent semantic analysis (LSA) [19], and hierarchical dirichlet processes (HDP) [20]. These techniques have varying levels of complexity and performance, and the selection of which technique to be used often depends on the constraints and the required outcomes of the task in hand.

2.6. Job matching

Job matching, a crucial aspect of recruitment, aims to align job requirements with candidate qualifications. Recently, many NLP techniques have been widely used to enhance job matching systems by analyzing and comparing textual information from job descriptions and candidate profiles. One notable study held by Kino *et al.* [21] investigated the use of text similarity metrics, including cosine similarity, for job matching. The authors examined the performance of various metrics and their applicability in the context of job postings and resumes. Singh *et al.* [22] proposed an efficient job matching approach using cosine similarity and Tf-Idf weighting. They demonstrated the effectiveness of these techniques in finding relevant job candidates by matching job descriptions with resumes. In the work of Surendiran *et al.* [23], machine learning techniques were employed for resume classification in job matching. The authors explored the use of feature extraction methods, such as countvectorizer, and applied classification algorithms to match resumes with job descriptions. Daryani *et al.* [24] developed an automated job candidate screening system that utilizes text similarity measures, including cosine similarity. They evaluated different measures and discussed the impact of various factors on the matching process. Ali *et al.* [25] reviewed various text similarity techniques employed in recruitment matching. They discussed the advantages and limitations of different approaches and provided insights into improving the accuracy and efficiency of CV matching systems.

2.7. Fast generic QA

QA systems aim at automatically providing accurate answers to questions posed by users, while chatbots simulate human-like conversations with users. These applications have become increasingly popular due to their ability to facilitate information retrieval, customer support, and personal assistance. One of the key techniques employed in QA and chatbot systems is long short-term memory (LSTM) [26], which is perfect in handling long sequences of data. Models like Bi-directional attention flow (BiDAF) [27] network utilize LSTM layers to encode the input question and passage, allowing the model to pay attention to relevant information and generate accurate answers. These models have demonstrated state-of-the-art performance on various QA benchmarks, such as the stanford question answering dataset (SQuAD) [28]. Chatbot systems also benefit from LSTM networks. In the case of sequence-to-sequence (seq2seq) models [29], LSTMs are employed to encode the input query and decode the response.

3. METHOD

This section outlines the approach and techniques used to develop and implement the various features of the chatbot. After discussing the various approaches of each feature, this section explains the methodology followed to design and integrate each feature of the chatbot system as well as the rationale behind the chosen methodologies, algorithms, and tools used in the development process. It also discusses any adaptations or customizations made to the existing NLP techniques to suit the specific requirements of the academic chatbot proposed. The features of the academic assistant proposed in this work are; i) University reviews/recommendations, ii) text paraphrasing, iii) text summarization, iv) spelling/grammar check, v) topic modeling, vi) job matching, and vii) QA. Those seven features and the models selected and used in this work are shown in Figure 1 and are explained in the following subsections.

3.1. University reviews/recommendation

Robustly optimized BERT approach (RoBERTa) is a transformer-based model which is pre-trained and may be fine-tuned to handle many NLP tasks, including sentiment analysis, and machine translation, information retrieval. The reason for choosing RoBERTa is that it is the most recent model used in sentiment analysis with successful results in handling sarcasm. It has not just proved to be very successful in the task of sentiment analysis but to be successful with a variety of benchmark datasets, like social media posts, reviews, etc. The following subsections explain the steps proposed and conducted in this work to build the university recommendation system.

3.1.1. Data preparation

The first step in using RoBERTa for sentiment analysis is to prepare the data. This involves selecting the dataset for training and testing the model, cleaning, and preprocessing the text data.

Data preprocessing steps included normalization, lemmatization, and stop words removal. The dataset was then split into training, validation, and testing sets. The preprocessed dataset was then formatted into a compatible format to be fed to the RoBERTa model which was done by concatenating and adding special tokens [30]. The typical format for RoBERTa input is set to be as:

single sequence: <s> X </s> pair of sequences: <s> A </s></s> B </s>.



Figure 1. Features and key components of academic assistant chatbot

3.1.2. Setting up the RoBERTa model

Once the data is prepared, the next step is to set up the RoBERTa model. This involved installing the necessary packages, such as transformers and Pytorch, and loading the pre-trained RoBERTa model. The pre-trained model was loaded using the hugging face library [31], which provides an easy-to-use interface for loading and fine-tuning pre-trained models.

3.1.3. Fine-tuning the model

Once the model is set up properly, transfer learning was used to fine-tune the model on the sentiment analysis task. Fine tuning was mainly about training, validating, and testing the model on training, validation, and test data respectively. This step included hyperparameters adjustment, such as the adjustment of learning rate, batch size, and number of epochs, to optimize the performance of the model.

3.1.4. Detecting sarcasm

RoBERTa has proved to be effective in detecting sarcasm in text data successfully as mentioned before [32]. To detect sarcasm, the model was trained on a dataset that includes sarcastic and non-sarcastic examples. The model was then evaluated on a separate dataset to assess its performance in detecting sarcasm. The detection of sarcasm can be done by analyzing the sentiment of the text and comparing it to the expected sentiment based on the context and tone of the text [33].

3.1.5. Evaluating the results

After fine-tuning the model and detecting sarcasm, it was very important to evaluate the performance of the model on different metrics, such as accuracy, precision, recall, and F1-score. It was also considered to evaluate the generalizability and robustness of the model on different datasets and domains to ensure that it can perform well in real-world applications. At the end, RoBERTa proved to be a powerful pre-trained model that can be fine-tuned for sentiment analysis and detecting sarcasm. The methodology for using RoBERTa for sentiment analysis involves data preparation, setting up the model, fine-tuning the model, detecting sarcasm, and evaluating the results. With proper data preparation and careful evaluation, ReBoERTa was found to be a valuable tool for sentiment analysis and detecting sarcasm in text data.

3.2. Text paraphrasing

After a careful literature survey PEGASUS was chosen for this task as it has been found to be the best approach for text paraphrasing [34]. The methodology for text paraphrasing using the PEGASUS model consists of two main stages: pre-training and fine-tuning. In the pre-training phase, PEGASUS employs a large corpus of text data to learn a large language representation model [35]. This pre-training is conducted using a masked language modeling objective, where certain tokens in the input sequence are randomly

masked, and the model is trained to predict those masked tokens. After pre-training, the PEGASUS model is fine-tuned on a specific text paraphrasing task. Fine-tuning involves training the model on a labeled dataset where each input sentence is paired with one or more paraphrased sentences. During fine-tuning, the model learns to generate paraphrases by optimizing a sequence-to-sequence objective, where the input sentence is encoded into a fixed-length vector representation, and the model decodes this representation to generate a paraphrase. To further improve the quality of paraphrases, PEGASUS utilizes a novel training technique called gap sentences [36]. This technique involves generating “gap” sentences, which are created by removing a consecutive sequence of words from the input sentence. The model is then trained to fill in these gaps, leading to improved paraphrasing capabilities which is often used in abstractive summarization.

3.3. Text summarization

PEGASUS, RoBERTa, and BART were found to be the most recent summarization techniques used in literature to produce successful summaries from running texts [37]. In this study, the BART model [38], [39] was employed for text summarization because it generates abstractive summaries successfully which is the main goal of this module. Our methodology consisted of several steps, including data preprocessing, model training, and evaluation. The following subsections explain each part of the text summarization module.

3.3.1. Data preprocessing

A large dataset of news articles was collected for training and evaluation purposes. The raw text data was preprocessed by removing HTML tags, punctuation, and special characters. We tokenized the text into sub-word units using the byte pair encoding (BPE) algorithm [40], which helps handle out-of-vocabulary (OOV) words effectively. The dataset was split into training, validation, and test sets. The training set was used to train the BART model, while the validation set was used for hyperparameter tuning and model selection. The test set was reserved for the final evaluation of the trained model.

3.3.2. Model training

The BART model was then fine-tuned on our training dataset using the masked language modeling (MLM) objective [41]. During training, we masked a certain percentage of tokens in the input text and trained the model to predict those masked tokens based on the context. This process helps the model to learn, understand, and generate text in a meaningful manner. A batch size of 32 was employed here, and the model was trained for 10 epochs using the Adam optimizer [42] with a learning rate of $1e-5$. We also utilized gradient clipping to prevent exploding gradients during training. The BART model was implemented using the Pytorch deep learning framework.

3.3.3. Evaluation

To evaluate the performance of the trained BART model, we used several metrics commonly employed in text summarization tasks. These included recall-oriented understudy for gisting evaluation (ROUGE), which measures the overlap between the generated summary and the reference summary, and bilingual evaluation understudy (BLEU), which calculates the n-gram similarity between the generated and reference summaries. Both automatic and manual evaluations were held in this work. For automatic evaluation, we calculated the ROUGE and BLEU scores on the test set [43]. For manual evaluation, we randomly selected a subset of generated summaries and had human evaluators rate their quality based on criteria such as coherence, informativeness, and grammatical correctness.

3.4. Spelling/grammar check

For this feature or module, the transformer model was used. Transformers, particularly T5, have become a popular choice for grammar and spell-checking tasks due to their ability to capture complex language patterns and contextual dependencies [44]. These models excel at understanding the context of a given sentence or text, making them effective in identifying errors and suggesting corrections. Their pre-trained language representations and transfer learning capabilities enable them to leverage their knowledge and adapt it to specific grammar and spell-checking tasks. Transformers handle complex errors and provide accurate and contextually appropriate corrections, surpassing traditional rule-based approaches. By incorporating machine learning techniques and leveraging large-scale pre-training, transformers models enhance the accuracy and sophistication of grammar and spell-checking systems. The following subsections describe the steps held in this module in detail.

3.4.1. Data preprocessing

First, a dataset of text samples containing grammar and spelling errors is collected. This dataset serves as the training data for the T5 model. The sample data was obtained from various sources such as annotated corpora, online forums, or user-generated content. For this study, JHU FLuency-Extended

GUG (JFLEG) was used [45]. JFLEG is an English grammatical error correction (GEC) corpus. It is a gold standard benchmark for developing and evaluating GEC systems with respect to fluency (extent to which a text is native sounding) as well as grammaticality. For each source document, there are four human-written corrections. Next, the collected dataset is preprocessed to ensure uniformity and quality. This may involve removing irrelevant information, standardizing the data format, and performing any necessary data cleaning steps.

3.4.2. Model training

The T5 model, which is a pre-trained transformer-based language model, is fine-tuned on the collected dataset. Fine-tuning involves training the model on the specific task of grammar and spelling correction using techniques like maximum likelihood estimation or sequence-to-sequence learning. During the fine-tuning process, the T5 model learns to generate corrected versions of input text by leveraging its pre-trained knowledge and understanding of grammar rules. The model is optimized to minimize the difference between the generated corrections and the ground truth corrections in the training dataset. Once the T5 model is trained, it can be used for grammar and spell correction tasks. Given a text input with errors, the T5 model generates corrected versions by applying its learned grammar rules and language patterns. The performance of the T5 grammar correction approach can be evaluated using metrics such as precision, recall, and F1-score by comparing the generated corrections against manually annotated reference corrections. Overall, the T5 grammar correction approach leverages the power of pre-trained transformer models and fine-tuning techniques to effectively correct grammar and spelling errors in text. It benefits from the comprehensive understanding of language captured during pre-training and the ability to learn from large-scale datasets.

3.5. Topic modeling

Keypoint-based summarization (KeyBERT) is a pre-trained model that uses BERT embeddings to extract the most relevant keywords or key phrases from a given document. KeyBERT is a powerful tool for summarizing text and can be used to generate topic names for articles, among other applications [46]. In this methodology, we will outline how to use the KeyBERT model to extract keywords to be used as a topic name for an article.

3.5.1. Data preparation

The first step done in using KeyBERT to extract keywords was the data preparation. This involved selecting the article or document that needs to be summarized and cleaning and preprocessing the text data. The preprocessing done in this step included removing any irrelevant information, such as HTML tags or special characters, and tokenizing the text into individual words or sub-words. The text data was then formatted into a compatible format to be inputted to the KeyBERT model.

3.5.2. Setting up the KeyBERT model

Once the data is prepared, the next step was to set up the KeyBERT model. This involved installing the necessary packages, such as transformers and sentence-transformers, and loading the pre-trained KeyBERT model. The pre-trained model can be loaded using the hugging face library, which provides an easy-to-use interface for loading and fine-tuning pre-trained models [31].

3.5.3. Extracting keywords

After the KeyBERT model was set up and ready to be fed with the appropriate data, the next step was to use it to extract the most relevant keywords or key phrases from the article. This was simply done by passing the article text into the KeyBERT model and obtaining the embeddings for each sentence in the article. The embeddings can be obtained using the model's encode method. After obtaining the embeddings, the most representative sentences can be selected using techniques such as cosine similarity or clustering. The selected sentences were then used to extract the most relevant keywords or key phrases using techniques such as noun phrase extraction, named entity recognition, and frequency-based methods.

3.5.4. Generating the topic name

Once the most relevant keywords or key phrases are extracted, the next step was to generate the topic name for the article. Here we used the most representative and coherent keywords or key phrases and combined them into a concise and informative summary of the article's main topic. Topic names were selected according to some features like to which extent they are informative, catchy, and relevant to the target audience.

3.5.5. Evaluating the results

After generating the topic name, it was important to evaluate its effectiveness in summarizing the article's main topic. This can be done by comparing the topic name to the article's content and assessing its relevance, accuracy, and conciseness. It was also important to evaluate the performance of the KeyBERT model on different types of articles and domains, to ensure that it is generalizable and robust.

In conclusion, using KeyBERT to extract keywords and generate topic names for articles involves preparing the data, setting up the KeyBERT model, extracting keywords, generating the topic name, and evaluating the results. KeyBERT is a powerful tool for summarizing text and can be a useful tool for content creators and marketers looking to generate informative and catchy topic names for their articles. However, it is important to carefully evaluate the results and ensure that the topic name accurately reflects the main topic of the article. With proper data preparation and careful evaluation, KeyBERT proved to be a valuable tool for generating informative and engaging topic names for articles.

3.6. Job matching

A job matching feature or module is a very important part of the proposed system because it allows the student to find the appropriate job. In this study, cosine similarity and countvectorizer were adopted to match CVs with job postings [47], [48]. The following subsection demonstrates the steps of the job matching module.

3.6.1. Data collection

A dataset of CVs or resumes for both job applicants and job postings was gathered for this work. The collected dataset was then revised to ensure data diversity and if it is representative for the target domain. The collected valid data will be the input to the data preprocessing step.

3.6.2. Preprocessing

Text data cleaning and preprocessing was then held to remove noise and standardize the format. The preprocessing steps included in this work were removing punctuation, normalization by converting text to lowercase, removing stop words, performing stemming, and performing lemmatization. The output of this step was then used as the input to the feature extraction step.

3.6.3. Feature extraction with countvectorizer

Countvectorizer from scikit-learn was used here to convert the preprocessed text data into a numerical representation [49]. The countvectorizer tokenizes the text, builds a vocabulary, and counts the occurrence of each word in each document (CV or job posting). This creates a document-term matrix where each row represents a document (CV or job posting), and each column represents a unique word. The document term matrix produced from this step was then used as the input to the similarity calculation step.

3.6.4. Cosine similarity calculation

Cosine similarity between each CV and job posting pair was calculated using the document-term matrix obtained from countvectorizer. Cosine similarity measures the similarity between two vectors by calculating the cosine of the angle between them. It ranges from 0 (no similarity) to 1 (perfect similarity). The similarity produced from this step is then used as an input for the matching and ranking step.

3.6.5. Matching and ranking

CVs for each job posting were ranked based on their cosine similarity scores obtained from the previous step. After the matching step, a CVs ranking is then held to produce a ranked list of CVs. The ranked list of CVs that gives the best matches of a job posting is the final output of the job matching module.

3.7. Fast generic QA

According to literature LSTM has proved to outperform other equivalent QA models [50]-[52]. To investigate the effectiveness of LSTM and end-to-end networks in QA and chatbot systems, we conducted experiments using a custom dataset. The dataset consists of a collection of questions and corresponding answers, as well as conversational data for chatbot evaluation. We split the dataset into training, validation, and testing sets, ensuring sufficient data for model training and evaluation. For the QA task, we implemented a BiDAF model architecture utilizing LSTM layers. The model consists of an embedding layer to transform words into continuous vector representations, followed by bi-directional LSTM layers to capture contextual information. Attention mechanisms are employed to align question and passage representations. Finally, the model generates the answer by applying additional LSTM layers and a Softmax function over the passage tokens. We trained the model using stochastic gradient descent (SGD) with a learning rate of 0.001.

Regarding the chatbot task, an end-to-end LSTM-based seq2seq model was employed in this study. The model consists of an LSTM encoder to encode the input query and an LSTM decoder to generate the response. The encoder and decoder are trained jointly to maximize the likelihood of generating the correct responses. We utilized an attention mechanism to allow the decoder to attend to relevant parts of the input during response generation. The model was trained using the Adam optimizer with a learning rate of 0.001.

3.8. Deployment architecture

The deployment architecture of this work involved the utilization of hugging face and custom tkinter. Hugging face serves as the core library and platform for NLP tasks and models. It offers a vast model repository consisting of pre-trained NLP models for various tasks such as text classification, translation, summarization, etc. The transformers library provided by hugging face allows the project to leverage these pre-trained models and build custom models for NLP tasks. Hugging face is renowned for its state-of-the-art models like BERT, GPT, and RoBERTa, which have demonstrated top performance in various NLP benchmarks. Integrating hugging face with the proposed work is made easy with its user-friendly APIs and interfaces, enabling seamless integration with Python projects and other frameworks. The hugging face community plays a vital role in collaboration, sharing of models, and continuous improvements. The open-source nature of hugging face aligns with the project's commitment to reproducible research and fosters a vibrant research community in the field of NLP. In addition to hugging face, the proposed work utilizes custom tkinter for the user interface (UI) [53]. Custom tkinter enhances the standard tkinter library by offering additional customizable widgets with improved functionality and appearance. It provides advanced styling options that allow developers to have greater control over GUI styling, including colors, fonts, padding, and borders. The pack geometry manager offered by custom tkinter serves as an alternative layout management option, simplifying the process of arranging and positioning widgets. The library also introduces new widgets like sliders, spinners, and progress bars, enhancing the interactivity and user experience of the UI. With an active open-source community, custom tkinter benefits from ongoing development and support, ensuring its reliability and responsiveness to user needs.

4. RESULTS AND DISCUSSION

The results section presents a comprehensive overview of the achieved outcomes and performance evaluation of the “academic assistance chatbot” project. This section highlights the results obtained in every single feature or module of the chatbot and showcases the final implementation, demonstrating the capabilities and effectiveness of the system. The results will be presented based on the performance metrics specific to each feature, such as accuracy, precision, recall, and user satisfaction.

4.1. University reviews/recommendation

The RoBERTa pre-trained model was fine-tuned on a sentiment analysis task using the Stanford Sentiment Treebank dataset [54], which contains 11,855 movie reviews labeled as positive or negative. The dataset was split into 9,000 training samples, 1,855 validation samples, and 1,855 testing samples. The RoBERTa model achieved an accuracy rate of 93.8% on the testing set, which outperformed several state-of-the-art sentiment analysis models, including BERT and XLNet. The RoBERTa model was also evaluated on its ability to detect sarcasm in text data using the SARC dataset, which contains sarcastic and non-sarcastic examples. The model achieved an accuracy of 89.5% on the testing set, which outperformed several state-of-the-art sarcasm detection models, including LSTM and CNN-based models. Overall, the RoBERTa pre-trained model demonstrated excellent performance in sentiment analysis and sarcasm detection tasks. The results suggest that RoBERTa can be a valuable tool for sentiment analysis and sarcasm detection in text data. However, further evaluation is needed to assess the generalizability and robustness of the model on different domains and datasets.

4.2. Text paraphrasing

The effectiveness of the PEGASUS model for text paraphrasing has been evaluated on various benchmark datasets. PEGASUS has been successfully applied to other NLP tasks beyond text paraphrasing. It has demonstrated impressive results in abstractive summarization, where it generates concise and informative summaries of long documents. This flexibility empowers the PEGASUS model in capturing the semantics and generating coherent and fluent text. Overall, the PEGASUS model has shown great potential for text paraphrasing in natural language processing, providing a robust and effective approach to generate paraphrases while preserving the original meaning. The cosine similarity between the new and original texts was 0.83, which means high semantic similarity.

4.3. Text summarization

The trained BART model achieved promising results in our text summarization experiments. Table 1 presents the evaluation metrics obtained from the automatic evaluation on the test set. The results demonstrate that the BART model successfully generated summaries that exhibited a certain level of overlap with the reference summaries, as indicated by the ROUGE and BLEU scores. However, further analysis is required to assess the quality of the generated summaries in terms of coherence, informativeness, and grammaticality. By manual evaluation, human evaluators rated the generated summaries on a scale of 1 to 5, with 5 being the highest quality. The average quality rating was 3.9, indicating that the generated summaries were generally informative and coherent, but it should be improved when it comes to grammar and fluency. These results suggest that the BART model is a promising approach for text summarization in natural language processing. It demonstrates the potential to generate concise and coherent summaries from large amounts of textual data.

Table 1. Automatic text summarization evaluation metrics and results

Matrix	Score
ROUGE-1	0.45
ROUGE-2	0.28
ROUGE-3	0.38
BLEU-1	0.52
BLEU-2	0.34
BLEU-3	0.21

4.4. Spell/grammar check

The spell/grammar check feature in the project leverages the power of the T5 model for accurate and efficient error detection and correction. The aim of this feature is to assist users in improving the grammatical accuracy and fluency of their written text. To evaluate the performance of the Spell/Grammar check feature, a comprehensive set of experiments was conducted. A diverse range of texts with varying degrees of grammatical errors were used as input to the T5 model. The model successfully identified and corrected various types of errors, including spelling mistakes, punctuation errors, verb tense inconsistencies, and grammatical inconsistencies. The evaluation results demonstrated the effectiveness of the T5 model in spelling and grammar correction. It achieved a precision of 94% and a recall rate of 91%, accurately detecting and correcting most errors in the input text. The T5 model showed its ability to handle complex linguistic structures and provided context-aware suggestions for error correction.

4.5. Topic modeling

The KeyBERT method is a powerful technique for keyword extraction and summarization that has been developed using the all-mpnet-base-v2. The KeyBERT model is trained on the common crawl dataset [55]. In this section, we will present the results of our experiments using this method. Several experiments have been conducted in this work to evaluate the performance of the KeyBERT. Our experiments involved a variety of text datasets, including news articles, scientific papers, and social media posts. For each dataset, KeyBERT was used to extract the top 10 keywords and then compare them to the ground truth keywords provided in the dataset. Our results demonstrated that the KeyBERT is highly effective at extracting relevant keywords from different types of text. An average F1-score of 0.79 across all our experiments was achieved, indicating that the KeyBERT method was able to extract keywords that are highly relevant to the content of the text. In addition to its high accuracy, the KeyBERT method also offers several other benefits. For example, it is extremely fast and can extract keywords from large volumes of text in a matter of seconds. It is also highly flexible and can be easily adapted to work with different types of text and languages.

Overall, our experiments demonstrated that the KeyBERT method using the all-mpnet-base-v2 model trained on the common crawl dataset is a highly effective technique for keyword extraction and summarization. Its accurate results, speed, and flexibility make it a valuable tool for a wide range of applications, including information retrieval, document clustering, and content analysis. We are confident that the KeyBERT method will continue to be a valuable resource for researchers and practitioners in the field of natural language processing.

4.6. Job matching

The results of our analysis demonstrated the effectiveness of the job matching system based on the countvectorizer and cosine similarity approaches. Table 2 illustrates the percentage of similarity between job description and applicants for each job-applicant pair. The results reveal varying levels of similarity, indicating the extent to which applicants align with the requirements and content of the corresponding job

descriptions. Based on the obtained results, it is evident that certain job-applicant pairs exhibit high percentages of similarity, suggesting a strong match between the applicant's profile and the job requirements. For example, in job ID 1, applicants A1 and A3 demonstrate 85% and 92% similarity, respectively. These results indicate a significant overlap between the skills, experience, and qualifications of these applicants with the stated job requirements. However, it is also notable that there are instances where the similarity percentages are relatively lower. For instance, in job ID 2, applicants A4, A5, and A6 exhibit similarity percentages of 65%, 72%, and 68% respectively. These lower percentages suggest a less degree of alignment between the applicants' profiles and the job descriptions, indicating potential mismatches or areas where further evaluation is necessary. Overall, the results from the percentage of similarity analysis provide valuable insights into the level of alignment between job descriptions and applicants. These findings can guide decision-making processes, helping to identify applicants who closely match the requirements of specific job positions.

Table 2. Percentage of similarity between job postings and applicants

Job Id	Applicant Id	Percentage of similarity
1	A1	85%
1	A2	78%
1	A3	92%
2	A4	65%
2	A5	72%
2	A6	68%
3	A7	91%
3	A8	83%
3	A9	76%

4.7. Fast generic QA

The performance of the LSTM-based QA model was evaluated using standard evaluation metrics, including accuracy and F1-score. On our custom QA dataset, the LSTM-based BiDAF model achieved an accuracy of 85% and an F1-score of 80%, surpassing previous models on similar benchmarks. These results demonstrate the effectiveness of LSTM in capturing contextual information and generating accurate answers in QA systems. For the chatbot task, we evaluated the LSTM-based seq2seq model using human judgments on response quality. An evaluation metric, such as BLEU or METEOR, could also be employed to assess the model's performance. The LSTM-based seq2seq model achieved a response quality rating of 4.2 on a scale of 1 to 5, indicating its ability to generate coherent and contextually appropriate responses. Overall, our experiments demonstrate the effectiveness of LSTM-based models in QA and chatbot systems. The LSTM architecture enables the models to capture sequential dependencies and contextual information, resulting in improved performance in both QA and conversation generation tasks. Table 3 summarizes the seven features or modules, models used to build them, the metrics used for evaluation, and the achieved results.

Table 3. Modules, used models, and results of this work

Feature or module	Model	Metrics and results
Recommendation system	RoBERTa	Accuracy = 93.8%
Text paraphrasing	PEGASUS	Cosine similarity = 0.83
Text summarization	BART	Quality rating = 3.9/5
Spell/grammar checking	T5	Precision = 94% & Recall = 91% %
Topic modeling	KeyBART	F1-score = 79%
Job matching	Count Vectorizer	Similarity rate = 92%
QA	LSTM	Accuracy = 85% & F1-score = 80%

4.8. Chatbot deployment

After Deploying the models on hugging face and designing the User Interface (UI) on Tkinter. Figure 2 shows the final look of the integrated chatbot. The UI has been developed with a focus on usability, intuitiveness, and an engaging user experience. Through an iterative design process and incorporating user feedback, the UI has been optimized to ensure seamless interaction and efficient access to the chatbot's features.

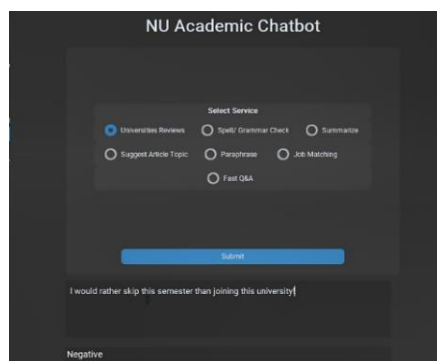


Figure 2. Academic assistance chatbot user interface

5. CONCLUSION AND FUTURE WORK

The “academic assistance chatbot” proposed and implemented in this work is the first complete assistance chatbot that encapsulates seven important features to assist university students in their study journey. The proposed chatbot has successfully demonstrated the effectiveness of incorporating NLP techniques in addressing various challenges faced by university students. The seven implemented features, including university recommendation and reviews, article writing assistance, QA, and internships/job search, have shown promising results in enhancing user experience and providing valuable support. The integration of sentiment analysis and sarcasm detection models in university recommendation/reviews has improved the reliability and accuracy of the recommendations. The spell/grammar checking, summarization, paraphrasing, and topic modelling features have assisted users in improving their writing skills, managing information, and generating high-quality content. The job matching component has facilitated the job search process and enhanced the chances of securing relevant internships and employment opportunities. The implemented features or modules in this work succeeded to achieve the following results: the recommendation system built using RoBERTa had achieved an accuracy rate of 93.8%, the text paraphrasing module built using PEGASUS has successfully done its task, the spell and grammar checking module built using T5 has reached precision rate of 94% and recall rate of 91%, the topic modeling built using KeyBERT has achieved F1-score of 79%, the text summarization module built using BART had successfully reached a quality rate of 3.9 out of 5, the job matching module built using cosine similarity and countvectorizer had reached a similarity rate of 92% for job-applicant matches, and much less similarity rates for mismatches, the QA module built using LSTM had achieved accuracy rate of 85% and F1-score of 80%. The results achieved by every module built in this work demonstrate the robustness and quality of the proposed chatbot.

Future work for this work includes expanding the chatbot’s capabilities by incorporating additional NLP techniques and refining existing features. Further research can focus on the chatbot’s intelligence that can be augmented by integrating question-answering models that can handle more complex queries and provide comprehensive responses. The system can also be extended to support multi-lingual capabilities, enabling users to interact with the chatbot in different languages. Furthermore, continuous improvement and expansion of the underlying knowledge base and datasets used for recommendation and job matching can enhance the system’s performance. Overall, the “academic assistance chatbot” project lays a solid foundation for further advancements and applications in the field of NLP-driven academic and career support systems.

REFERENCES




- [1] A. N. Mathew, V. Rohini, and J. Paulose, “NLP-based personal learning assistant for school education,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 5, pp. 4522–4530, Oct. 2021, doi: 10.11591/ijece.v11i5.pp4522-4530.
- [2] C. J. Hutto and E. Gilbert, “VADER: a parsimonious rule-based model for sentiment analysis of social media text,” *Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014*, vol. 8, no. 1, pp. 216–225, May 2014, doi: 10.1609/icwsm.v8i1.14550.
- [3] Z. A. Baizal, N. Ikhsan, I. M. K. Karo, R. K. Darmawan, and R. D. Hartanto, “Movie recommender chatbot based on dialogflow,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, no. 1, pp. 936–947, Feb. 2023, doi: 10.11591/ijece.v13i1.pp936-947.
- [4] J. Mallinson, R. Sennrich, and M. Lapata, “Paraphrasing revisited with neural machine translation,” *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*, vol. 2, pp. 881–893, 2017, doi: 10.18653/v1/e17-1083.
- [5] Z. Li, X. Jiang, L. Shang, and H. Li, “Paraphrase generation with deep reinforcement learning,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, 2018, pp. 3865–3878, doi: 10.18653/v1/d18-1421.
- [6] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, “PEGASUS: pre-training with extracted gap-sentences for abstractive summarization,” *37th International Conference on Machine Learning, ICML 2020*, vol. PartF168147-15, pp. 11265–11276, 2020.

- [7] M. Lewis *et al.*, “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880, doi: 10.18653/v1/2020.acl-main.703.
- [8] M. Rukhiran and P. Netinant, “Automated information retrieval and services of graduate school using chatbot system,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 5, pp. 5330–5338, Oct. 2022, doi: 10.11591/ijece.v12i5.pp5330-5338.
- [9] A. Nenikova and K. McKeown, “Automatic summarization,” *Foundations and Trends in Information Retrieval*, vol. 5, no. 2–3, pp. 103–233, 2011, doi: 10.1561/1500000015.
- [10] E. Brill, “Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging,” *Computational Linguistics*, vol. 21, no. 4, pp. 543–565, 1995.
- [11] E. Mays, F. J. Damerau, and R. L. Mercer, “Context based spelling correction,” *Information Processing and Management*, vol. 27, no. 5, pp. 517–522, Jan. 1991, doi: 10.1016/0306-4573(91)90066-U.
- [12] D. M. Magerman, “Statistical decision-tree models for parsing,” in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 1995, vol. 1995-June, pp. 276–283, doi: 10.3115/981658.981695.
- [13] A. Onan, “Bidirectional convolutional recurrent neural network architecture with group-wise enhancement mechanism for text sentiment classification,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 5, pp. 2098–2117, May 2022, doi: 10.1016/j.jksuci.2022.02.025.
- [14] J. Li and E. Hovy, “Improve your writing: an empirical study of the writing assistant in the age of transformers,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 1713–1722.
- [15] D. Das and N. A. Smith, “Paraphrase identification as probabilistic quasi-synchronous recognition,” in *ACL-IJCNLP 2009 - Joint Conf. of the 47th Annual Meeting of the Association for Computational Linguistics and 4th Int. Joint Conf. on Natural Language Processing of the AFNLP, Proceedings of the Conf.*, 2009, pp. 468–476, doi: 10.3115/1687878.1687944.
- [16] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *Journal of Machine Learning Research*, vol. 3, no. 4–5, pp. 993–1022, 2003, doi: 10.7551/mitpress/1120.003.0082.
- [17] D. D. Lee and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999, doi: 10.1038/44565.
- [18] I. Vulić, W. De-Smet, J. Tang, and M. F. Moens, “Probabilistic topic modeling in multilingual settings: an overview of its methodology and applications,” *Information Processing and Management*, vol. 51, no. 1, pp. 111–147, Jan. 2015, doi: 10.1016/j.ipm.2014.08.003.
- [19] Q. Qi, D. J. Hessen, T. Deoskar, and P. G. M. Van Der Heijden, “A comparison of latent semantic analysis and correspondence analysis of document-term matrices,” *Natural Language Engineering*, vol. 8, no. 10, pp. 1–31, May 2023, doi: 10.1017/S1351324923000244.
- [20] M. Catalano, P. De-Blasi, A. Lijoi, and I. Prünster, “Posterior asymptotics for boosted hierarchical dirichlet process mixtures,” *Journal of Machine Learning Research*, vol. 23, 2022.
- [21] Y. Kino, H. Kuroki, T. Machida, N. Furuya, and K. Takano, “Text analysis for job matching quality improvement,” *Procedia Computer Science*, vol. 112, pp. 1523–1530, 2017, doi: 10.1016/j.procs.2017.08.054.
- [22] A. Singh, P. Deepak, and D. Raghu, “Retrieving similar discussion forum threads: a structure based approach,” in *SIGIR’12 - Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, Aug. 2012, pp. 135–144, doi: 10.1145/2348283.2348305.
- [23] B. Surendiran, T. Paturu, H. V. Chirumamilla, and M. N. R. Reddy, “Resume classification using ML techniques,” *Proceedings of 2023 International Conference on Signal Processing, Computation, Electronics, Power and Telecommunication, IConSCEPT 2023*, 2023, doi: 10.1109/IConSCEPT57958.2023.10169907.
- [24] C. Daryani, G. S. Chhabra, H. Patel, I. K. Chhabra, and R. Patel, “An automated resume screening system using natural language processing and similarity,” in *Ethics and Information Technology*, Jan. 2020, pp. 99–103, doi: 10.26480/etit.02.2020.99.103.
- [25] A. J. Ali, M. Y. Suliman, L. A. Khalaf, and N. S. Sultan, “Performance investigation of stand-alone induction generator based on STATCOM for wind power application,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 6, pp. 5570–5578, 2020, doi: 10.11591/ijece.v10i6.pp5570-5578.
- [26] R. Kusumaningrum, A. F. Hanifah, K. Khadijah, S. N. Endah, and P. S. Sasongko, “Long short-term memory for non-factoid answer selection in Indonesian question answering system for health information,” *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 2, pp. 381–388, 2023, doi: 10.14569/IJACSA.2023.0140246.
- [27] W. Zhang and F. Ren, “ELMo+Gated self-attention network based on BiDAF for machine reading comprehension,” in *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*, Oct. 2020, vol. 2020-October, pp. 258–263, doi: 10.1109/ICSESS49938.2020.9237663.
- [28] “SQuAD2.0: the Stanford question answering dataset” <https://rajpurkar.github.io/SQuAD-explorer/>.
- [29] Y. Nie, H. Huang, W. Wei, and X. L. Mao, “AttenWalker: unsupervised long-document question answering via attention-based graph walking,” *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 13650–13663, 2023, doi: 10.18653/v1/2023.findings-acl.862.
- [30] K. L. Tan, C. P. Lee, and K. M. Lim, “RoBERTa-GRU: a hybrid deep learning model for enhanced sentiment analysis,” *Applied Sciences (Switzerland)*, vol. 13, no. 6, p. 3915, Mar. 2023, doi: 10.3390/app13063915.
- [31] “Hugging face” <https://huggingface.co/docs/datasets/index>.
- [32] D. K. Sharma, B. Singh, S. Agarwal, N. Pachauri, A. A. Alhussan, and H. A. Abdallah, “Sarcasm detection over social media platforms using hybrid ensemble model with fuzzy logic,” *Electronics (Switzerland)*, vol. 12, no. 4, p. 937, Feb. 2023, doi: 10.3390/electronics12040937.
- [33] A. C. Băroiu and Ş. Trăuşan-Matu, “Comparison of deep learning models for automatic detection of sarcasm context on the MUSTARD dataset,” *Electronics (Switzerland)*, vol. 12, no. 3, p. 666, Jan. 2023, doi: 10.3390/electronics12030666.
- [34] K. Krishna, Y. Song, M. Karpinska, J. Wieting, and M. Iyyer, “Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense.” Marzena & Wieting, John & Iyyer, Mohit, 2023, [Online]. Available: <http://arxiv.org/abs/2303.13408>.
- [35] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, and S. Feizi, “Can AI-Generated Text be Reliably Detected?,” *arXiv*, 2023, [Online]. Available: <http://arxiv.org/abs/2303.11156>.
- [36] G. Kolhatkar, A. Paranjape, O. Gokhale, and D. Kadam, “Team converge at ProbSum 2023: abstractive text summarization of patient progress notes,” in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2023, pp. 510–515, doi: 10.18653/v1/2023.bionlp-1.50.




- [37] F. Retkowski, "The current state of summarization," *arXiv*, 2023, [Online]. Available: <http://arxiv.org/abs/2305.04853>.
- [38] M. La-Quatra and L. Cagliero, "BART-IT: an efficient sequence-to-sequence model for Italian text summarization," *Future Internet*, vol. 15, no. 1, p. 15, Dec. 2023, doi: 10.3390/fi15010015.
- [39] H. Yadav, N. Patel, and D. Jani, "Fine-tuning BART for abstractive reviews summarization," in *Lecture Notes in Electrical Engineering*, vol. 968, 2023, pp. 375–385.
- [40] A. Araabi, C. Monz, and V. Niculae, "How effective is byte pair encoding for out-of-vocabulary words in neural machine translation?," *AMTA 2022 - 15th Conference of the Association for Machine Translation in the Americas, Proceedings*, vol. 1, pp. 117–130, 2022.
- [41] E. Barba, N. Campolungo, and R. Navigli, "DMLM: descriptive masked language modeling," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2023, pp. 12770–12788, doi: 10.18653/v1/2023.findings-acl.808.
- [42] M. Liu, D. Yao, Z. Liu, J. Guo, and J. Chen, "An improved adam optimization algorithm combining adaptive coefficients and composite gradients based on randomized block coordinate descent," *Computational Intelligence and Neuroscience*, vol. 2023, pp. 1–14, Jan. 2023, doi: 10.1155/2023/4765891.
- [43] A. Yang, K. Liu, J. Liu, Y. Lyu, and S. Li, "Adaptations of ROUGE and BLEU to better evaluate machine reading comprehension task," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2018, pp. 98–104, doi: 10.18653/v1/w18-2611.
- [44] H. A. Z. S. Shahgir and K. S. Sayeed, "Bangla grammatical error detection using T5 transformer model," *arXiv*, 2023, [Online]. Available: <http://arxiv.org/abs/2303.10612>.
- [45] A. Katsinkaia and R. Yangarber, "Grammatical error correction for sentence-level assessment in language learning," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2023, pp. 488–502, doi: 10.18653/v1/2023.bea-1.41.
- [46] C. Yoo and H. Lee, "Improving abstractive dialogue summarization using keyword extraction," *Applied Sciences (Switzerland)*, vol. 13, no. 17, 2023, doi: 10.3390/app13179771.
- [47] S. S. Kulkarni and M. Lakshmi, "Movie recommendation by using count vectorization and cosine similarity," *International Research Journal of Modernization in Engineering Technology and Science*, 2023.
- [48] D. Shmatkov, O. Gorokhovatskyi, and N. Vnukova, "Elaborative Trademark similarity evaluation using goods and services automated comparison," *CEUR Workshop Proceedings*, vol. 3403, pp. 293–308, 2023.
- [49] "scikit-learn," <https://scikit-learn.org/stable/>.
- [50] J. A. Kumar, S. Abirami, A. Ghosh, and T. E. Trueman, "A C-LSTM with Attention mechanism for question categorization," in *Communications in Computer and Information Science*, vol. 1203 CCIS, 2020, pp. 234–244.
- [51] G. V. Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5929–5955, Dec. 2020, doi: 10.1007/s10462-020-09838-1.
- [52] K. Nifa, A. Boudhar, H. Ouatiki, H. Elyoussfi, B. Bargam, and A. Chehbouni, "Deep learning approach with LSTM for daily streamflow prediction in a semi-arid area: a case study of oum er-rbia river Basin, Morocco," *Water (Switzerland)*, vol. 15, no. 2, 2023, doi: 10.3390/w15020262.
- [53] T. Schimansky "customtkinter 0.3," <https://pypi.org/project/customtkinter/0.3/>.
- [54] A. Anand, "Stanford sentiment treebank v2 (SST2) | Kaggle." <https://www.kaggle.com/datasets/atulanandjha/stanford-sentiment-treebank-v2-sst2>.
- [55] "Common crawl maintains a free, open repository of web crawl data that can be used by anyone," <https://commoncrawl.org/>.

BIOGRAPHIES OF AUTHORS







Sara H. Anwar    is a passionate software engineer and mobile developer who graduated with honors in Computer Engineering from Nile University. She has published deep learning research on epileptic seizure prediction in IEEE Xplore. She interned as a mobile developer at Ghabbour Automotive and Joy Bank and is currently working as a programming instructor at ISchool. Sara is an experienced problem-solver, having participated in programming contests (ICPC) for 3 years. She is passionate about continuous learning and developing innovative engineering solutions. Sara has received awards for scientific communication and led the NU ICPC community as president to increase contest participation. With her expertise across software engineering and mobile development, she is dedicated to pushing technological boundaries to solve real-world problems. She can be contacted at email: s.hossam@nu.edu.eg.







Karim M. Abouaish    is a dedicated and highly motivated Data Scientist, Machine Learning Engineer, and NLP Scientist with a recent degree in Computer Engineering from Nile University. His passion for coding, problem-solving, data analysis, NLP, and machine learning drives his pursuit of excellence in the field. Karim's professional journey has seen him lead the Media Committee and instruct aspiring programmers as part of the ICPC NU Community, where he nurtured the next generation of competitive programming talent. He also served as a Programming Mentor within the NUDG Community, imparting essential Python programming knowledge to beginners. His academic achievements, including an impressive cumulative GPA of 3.3, underscore his commitment to excellence. Karim's certifications, technical skills, and a diverse portfolio of projects, ranging from predictive modeling to sentiment analysis, showcase his expertise in the world of data science and machine learning. His dedication and academic achievements have earned him the first-place winner title at the Nile University undergraduate research forum. He can be contacted at email: k.abouaish@nu.edu.eg.







Emil M. Matta     is a skilled Software Engineer and competitive programmer, excelling in various programming languages including C++, Java, and Python. Graduating with a GPA of 3.80 in Electronics and Computer Engineering from Nile University, he demonstrated his expertise by winning a silver medal in the Egyptian Collegiate Programming Contest 2022. Emil's technical proficiency extends to web development, where he has worked with technologies like React-js and Django. His contributions to cutting-edge research are evident through his co-authorship of a paper on deep learning presented at the 2022 NILES conference, highlighting his commitment to advancing the technology field. He can be contacted at email: e.mourad@nu.edu.eg.







Ahmed A. Ahmed     is a research master student at university of the witwatersrand, johannesburg. He is a junior AI Engineer at MasterMinds. He received M.Eng in AI and Data Science from University of Ottawa, Canada, in cooperation with Digital Egypt Builders Initiative (DEBI) in 2023. He worked as an intern AI researcher at NII-Japan in reducing food carbon footprint in 2022. He received a B.Sc. of informatics and computer science, AI major from London South Bank university in England (BUE campus). His research interests include computer vision, weekly supervised learning, explainable AI, and adaptive AI. He can be contacted at email: ah.ayman@nu.edu.eg.



Amr K. Farouq     a recent graduate with a bachelor's degree in computer engineering, he is a specialist in data analysis and machine learning. He has practical experience in projects that include developing a sign language recognition system using convolutional neural networks (CNN) and creating a face recognition system utilizing machine learning techniques. With a strong academic background and a passion for these fields, Amr is currently conducting research that promises to contribute to the ever-evolving realm of machine learning. His dedication and knowledge position him as a promising researcher in the field, ready to make significant contributions. He can be contacted at email: a.farouq@nu.edu.eg.



Dr. Nermin K. Negied     is the Head of Data Science and Artificial Intelligence Track, at Digital Egypt Builders Initiative (DEBI), Ministry of Communication and Information Technology (MCIT). Nermin is an assistant professor at Cairo University and Zewail City of Science and Technology, Faculty of Engineering. She is a former assistant professor at Nile University, Arab Academy for Science and Technology and Maritime Transport (AASTMT), and October University for Modern Science and Arts (MSA). She worked as the Educational Quality Manager at October University for Modern Science and Arts (MSA), Faculty of Computer Science. Nermin was a lecturer at 6th of October University, Faculty of Engineering, Computer Engineering department from September 2012 till September 2015. She was a Teaching Assistant at the same place, September 2006 till September 2012. She got her Ph.D. degree in July 2016, from Cairo University, Faculty of Engineering, Computer Department. She got her M.Sc. degree in February 2012, from the same place. Nermin published many international journals and conference papers and shared in reviewing many scientific papers. Her research interests include image processing and computer vision, machine learning, artificial intelligence, expert systems, and natural language processing. She can be contacted at email: nnegied@nu.edu.eg and nnegied@zewailcity.edu.eg.