CONTENT

INTRODUCTION

The development of Natural Language Processing (NLP) is undoubtedly a significant milestone in human history. As a pioneering technique, it has brought about profound changes across every aspect of human industry. From healthcare, education, financial and entertainment sectors, the impact of NLP is far-reaching and transformative. NLP, at its core, is about creating systems that understand, interpret, and generate human language in a valuable way. It's a multidisciplinary field combining computational linguistics, artificial intelligence, and computer science to enable machines to understand human language as it is spoken or written. The goal is to bridge the gap between human communication and digital data processing, and it has been largely successful in this endeavor.

One significant benefit of NLP is its ability to greatly reduce the workload on human resources. By harnessing the power of computers to identify patterns and respond accordingly, NLP eliminates the need for humans to perform repetitive tasks. For instance, in customer service, NLP-powered chatbots could handle common queries, freeing up human agents to deal with more complex issues. This not only improves efficiency but also leads to higher customer satisfaction. Moreover, NLP systems are capable of automating many processes without any human interference. For example, they could analyze and sort through large volumes of data, extract relevant information, and even make predictions based on patterns. This level of automation is particularly useful in fields like market research and social media analysis, where the volume of data could be overwhelming for humans to process efficiently. Another advantage that NLP offers is precision. With the capability of computational power, these systems could achieve a level of accuracy that far exceeds human capabilities. Digital devices could work tirelessly without experiencing fatigue or loss of focus, which leads to fewer mistakes compared to tasks performed by humans. This precision is crucial in sectors like healthcare, where errors could have serious consequences. Furthermore, NLP allows for scalability. While there are limits to how much work a human could handle, NLP systems could process vast amounts of data in a short time. This ability to scale makes NLP an invaluable tool in our data-driven world. With all these advantages and the rapid advancement of technology, it's not surprising that digital supports are becoming preferable to human workers in many areas. They offer cost-effective solutions that could improve efficiency, reduce errors, and deliver valuable insights.

NLP is not a static field. It continues to evolve, driven by advancements in machine learning and deep learning. As these technologies improve, so does the potential of NLP. Digital devices are starting to show that they could not only understand and generate human language but could also understand context, detect sentiment, and even exhibit elements of creativity. Thus, NLP stands as a testament to human ingenuity and ability to leverage technology to solve complex problems. From improving accessibility for people with disabilities to powering virtual assistants and beyond, the possibilities are virtually endless. As exploration and innovation go deeper and wider, there's no doubt that NLP would continue to reshape our world in ways humans could not even imagine yet.

The field of education is currently experiencing a dramatic transformation, thanks to the technological revolution. A key catalyst in this change is the use of NLP. This technology is being leveraged to address various educational challenges, with the admission process for international students being a notable example. The enrollment procedure has become increasingly complex since globalization has begun, resulting in operational inefficiencies due to delays and extended response times. Yet, the incorporation of automation, powered by NLP, has shown promise in mitigating these difficulties. NLP has exhibited its capacity to replace human involvement in many procedures, leading to more streamlined and efficient operations. As a result, an increasing number of

educational institutions are investigating methods to enhance their systems with NLP. The primary goal is to avoid potential student attrition due to avoidable human-related problems.

The integration of NLP into these systems allows institutions to provide swift responses, optimize procedures, and improve overall student satisfaction. However, the influence of NLP isn't just limited to administrative duties. It's being employed to craft personalized learning experiences that cater to the individual needs of students. It's playing a significant role in language learning, adding an element of interactivity and engagement. It's even proving beneficial in grading assignments, ensuring fair and consistent assessments. Furthermore, NLP is demonstrating its worth by improving communication between students and teachers, particularly in the context of remote learning. It's facilitating the development of virtual tutors that could offer 24/7 support to students, promoting continuous learning. It's also proving useful in research, aiding in data gathering and analysis, and assisting in the identification of plagiarism. The potential uses of NLP in education are boundless, and the research works are just beginning to tap into its potential. As this technology continues to evolve and improve, it would further broaden the spectrum of enhancing efficiency and productivity in tasks traditionally handled by humans.

Background

During the annual admissions period, thousands of ambitious students reach out, seeking pertinent information about their desired courses and institutions. However, with limited manpower available in the admissions team, it becomes a challenging task to meticulously scrutinize each question and craft personalized responses. The immediate solution that many institutions have turned to involves the use of web-based systems and social media. These platforms have become popular solutions, serving as one-stop information hubs for prospective students. All the university needs to do is to upload relevant information on the platform, and it becomes a self-help center where candidates could find answers to their academic inquiries. However, due to the exponential increase in the volume of information every day, navigating through these platforms could become a highly time-consuming process for users. This is where the need for more sophisticated, user-friendly systems comes into play. Therefore, the development of graphical user interfaces equipped with NLP capabilities has become the prominent solution in further development phases. Harnessing the power of NLP techniques coupled with a well-trained model, the entire enrollment process could be automated, from pre-admission Q&A to payment processing. This level of automation reduces the need for human interaction significantly, thereby streamlining the process and making it more efficient.

However, as a human-like digital helper, sometimes its emulation ability is doubted for immediate response and response quality. The inability to provide immediate responses with correspondent response could potentially cause institutions to lose prospective students, affecting both the quantity and quality of the student intake. This is a serious concern for any university and it would be required to carefully select the proper method depending on various occasions. By providing real-time, accurate responses, NLP-powered systems could help universities engage better with their prospective students, reducing the risk of losing them due to delayed or inadequate responses. However, implementing such a system is challenging considering the quality of the dataset, model selection and complex workflow. The ideal result is to keep the system with minimal human intervention while simultaneously maintaining the credibility of the dialogue. The system needs to be robust, able to handle a variety of queries, and provide a natural, human-like interaction. This not only involves complex technical development but also requires a deep understanding of the user's needs and expectations. Moreover, the system needs to be adaptive and continually learn from

user interactions to improve its responses over time. This calls for a dynamic learning model that evolves with each interaction, becoming more accurate and reliable.

Aim of the research

The primary objective of this study revolves around the conceptualization and development of enrollment support systems from the ground up. The system is designed to assign appropriate responses to specific questions, leveraging the capabilities and insights from existing state-of-the-art methods and advanced NLP models. This sophisticated system is a hybrid creation, drawing inspiration from a multitude of NLP studies and integrating various components to handle multiple challenges in the development. It combines diverse elements from the realm of NLP, ranging from the intricate design of pre-trained language models and innovative data augmentation strategies to the implementation strategies of applying recommendation systems. Each adopted approach within the system is tailored with slight modifications, carefully made to enhance its functionality without altering the core essence of the original method. These modifications are optimized to cater to the specific requirements of the current work, ensuring that each component is optimized to contribute effectively to the overall performance of the system. The overarching ambition of this study is to construct a robust, automated system with enduring capabilities. The system is designed to respond to any domain-related questions, using a predefined data set to produce relevant and accurate answers.

This set of domain-related questions is related to the program “Big Data and Machine Learning” in the field of "Science in Applied Mathematics and Informatics," provided by the ITMO University The system is not just a mere combination of various NLP techniques. Instead, it represents the culmination of extensive research and careful integration of multiple methodologies to create a comprehensive solution. It's a testament to the transformative power of NLP, demonstrating how the strategic fusion of disparate techniques could culminate in a unified, practical system capable of addressing real-world challenges.

Research plan and contributions

The plans are listed in the format of questions below :

- How to design a system with minimum human interference ?
- How proposed data augmentation methods could perform for expansion of datasets ?
- Could hypercomplex classifiers be more discriminative for different labels than fully-connected layers in pretrained NLP models ?
- How to reduce the mistake of model prediction ?

The main contributions are categorized into 3 parts:

- Sustainable workflow of proposed enrollment support system
- Customized data augmentation (DA) by using pos-tagging
- The implementation of replacing hypercomplex classifier layers with fully-connected layers in pretrained NLP model

In section 2, a comprehensive review of correlated research is delved into, focusing on existing masterpieces of each study. This section serves to provide a solid foundation of understanding for the reader and positions this work within the broader scholarly research. In section

3, it presents the intricate details of implementation of the current system. This includes the data sources, preprocessing steps, model training, and optimization strategies adopted for NLP tasks. In section 4, it aims to provide a transparent overview of experimental setup, enabling the reproduction of this work. In section 5, it showcases the results derived from experiment. This section includes a thorough analysis of the performance of the proposed DA techniques and classifier models, supported by various metrics to demonstrate their effectiveness. Comparative analysis with existing approaches was also provided, highlighting the improvements achieved by proposed methodologies. The discourse concludes in section 6, summarizing the key findings of current work, discussing their implications in the context of current NLP research. It also reflects on the limitations of current study, and suggests possible solutions. Lastly, it outlines future plans and possible development.

1 RELATED WORKS

1.1 Intellectual system in higher school

Chatbots have become an integral part of many institutions, and their functionalities could differ greatly based on the specific requirements of each category. They are adaptable and flexible, able to cater to a wide range of needs. One of the standout features of chatbots is their ability to communicate in various formats. They are not restricted to text-based interactions – they could also engage users through images, and even audio communication. This multiplicity in communication methods allows chatbots to cater to a wide range of user preferences and ensures a more interactive and engaging user experience.

The implementation of intelligent systems such as chatbots doesn't necessarily hinge on location constraints. This is one of the advantages that make them a favored choice as a support system in various institutions. They could be installed and operated from almost anywhere, making them a powerful tool for remote assistance and support. A key area where chatbots have found substantial application is in web systems and social media platforms. With the majority of people spending a significant amount of time online, chatbots on these platforms could provide immediate assistance, answer queries, and even guide users through various processes. They could also gather valuable customer data, helping institutions to understand user behavior and preferences better.

Moreover, there are statistics showing that chatbots have the ability to handle 65% of questions from users for educational purposes[1], in addition the service is able to run 24/7. Due to all these advantages, the usage of chatbot is more preferable each day.

In the realm of practical applications, chatbots have demonstrated their versatility and efficiency in various sectors across the globe. Their application is not limited to financial ability, which enables any institute to be capable of deploying it. It is widely used from the developed countries, such as the United States to developing countries like Somalia, testifying to their universal applicability and affordability. This wide adoption of chatbots also indicates their scalability and adaptability to diverse socio-economic contexts and technological infrastructures.

One noteworthy example of the effective use of chatbots is at Georgia State University[2]. This institution owns one of the world's most advanced chatbots, which plays a crucial role in the enrollment process and supports students throughout their educational journey. The chatbot is designed to provide immediate, personalized responses to queries regarding admission, course selection, academic requirements, and more. This not only streamlines the administrative process but also enhances the overall student experience by providing them with timely and accurate information. The chatbot at Georgia State University also showcases how these intelligent systems could contribute to educational equity and accessibility. By offering round-the-clock support, it ensures that all students, regardless of their schedules or time zones, have access to the information they need. This could be particularly beneficial for non-traditional students, such as those who work full time or have family responsibilities, who may not be able to seek assistance during regular office hours. Furthermore, the affordability of chatbots is a significant factor in their widespread adoption. They could provide cost-effective solutions for customer service, reducing the need for a large support staff and minimizing the waiting time for users. Telkom University (Indonesia) established its own Q&A model for educational issues[3] and Jamhuriya University (Somalia) also customized their own intelligent system to relieve stress in admission period[4], some other related works which inspire current works are listed in APPLICATIONS B.

1.2 Knowledge injection

Large language models (LLMs) are able to capture vast amounts of factual information. LLMs exhibit a remarkable level of knowledge in various domains due to their massive pre-training datasets, however due to the corpus of its pre-training datasets and tremendous resources for training LLMs, here come with two main drawbacks to be improved : (1) knowledge in LLMs could not be updated by time (2) knowledge is not specific to any certain area.

As a result, using external data sets to incorporate new information or improve the LL.M.'s abilities based on previously acquired information becomes attractive researches in the NLP field, especially after the advent of chatGPT astonishing the whole community with its extremely human-like reasoning ability and dialogue generation capacity, but one deadly drawback is that knowledge is limited until 2021.

Some conclusion of why LLMs could fail at answering factual questions are made by[5] with five main factors: (1) Domain knowledge deficit (2) Outdated Information (3) Immemorization (4) Forgetting (5) Reasoning Failure

It is not difficult to imagine that general pre-training is insufficient for business purposes. To overcome this challenge, an additional post-processing step is essential to augment the knowledge of a pre-trained model, which is commonly known as knowledge injection[6]. The assumption of knowledge injection is that if the corpus external knowledge is accessible for LLMs, it could serve as a supporting knowledge base and improve the model's performance on that set of problems related to specific range areas. Two most well-known methods are : fine-tuning and retrieval augmented generation (RAG)

Fine-tuning is a vital part of the machine learning process, in which a pre-trained model undergoes additional training, or "fine-tuning," on a novel, specific dataset or task. The objective of this procedure is to utilize the general knowledge acquired by the model during the pre-training phase and modify it to suit the subtleties of a particular task or domain. This method boosts the model's proficiency in tackling the specific domain, enabling it to carry out the respective task more efficiently, although each step could require substantial resources. It's crucial to differentiate between various types of fine-tuning when discussing it, as each type is suited to different learning environments. These types are usually divided into three categories: supervised, unsupervised, and reinforcement fine-tuning.

1.2.1 Fine-Tuning

Supervised fine-tuning (FT) necessitates the availability of labeled pairs of input and output. This means that for every piece of data in the set, there's a corresponding label that indicates the desired output. One of the most prevalent techniques used in Supervised FT is instruction tuning. This technique has proven to be extraordinarily effective at enhancing the performance of the model. Instruction tuning is a process where the model is given specific instructions in the input, guiding it towards producing the desired output. This method helps in refining the model's predictions, making it a powerful tool for improving model performance. Many current state-of-the-art LLMs have gone through instruction tuning after their pre-training phase. Instruction tuning has demonstrated a strong ability to enhance the overall capability of a machine learning model. This technique is particularly impactful in improving the model's zero-shot learning and reasoning abilities. Zero-shot learning refers to the model's ability to accurately perform tasks that it has not been explicitly trained on, thus demonstrating a form of generalized learning. Similarly, enhanced reasoning capabilities mean that the model could understand complex relationships, infer missing information, and make logical conclusions based on the information it has been trained on. However, it's

important to note that while instruction tuning significantly improves a model's performance in these areas, it doesn't necessarily import new knowledge to the model. The process of instruction tuning refines the model's existing knowledge and understanding, helping it to better apply this knowledge to new situations. However, it does not introduce new information or facts that the model was not previously exposed to during its pre-training phase. Therefore, the model's knowledge remains limited to the information present in its initial training set. Given this limitation, instruction tuning alone could not resolve the knowledge injection problem, which refers to the challenge of introducing new information into a pre-trained model. Addressing this problem requires a more complex solution that may involve a combination of techniques, such as dynamic knowledge retrieval systems or periodic retraining on updated datasets.

Another variant of fine-tuning leverages the potential of Reinforcement Learning (RL) or RL-inspired optimization strategies to refine the model following its pre-training phase. The essence of RL is that it allows the model to learn from its interactions with the environment, thereby continuously improving its performance based on the feedback it receives. Some well-known strategies are reinforcement learning from human feedback(RLHF), direct preference optimization (DPO)[7]. Both of these methods strive to fine-tune the model in a way that aligns it more closely with human values and preferences, leading to improved performance and usability. However, they also pose unique challenges, such as the need for high-quality human feedback and the potential for overfitting to the preferences of the evaluators. Despite these challenges, RL-based fine-tuning methods represent a promising direction for improving the performance and alignment of large language models. These techniques have been shown to be very useful, especially when used in conjunction with instruction tuning. However, similarly to instruction tuning, these methods focus on the overall quality of the response and its expected behavior and not necessarily on its breadth of knowledge.

The last FT strategy we're going to discuss is unsupervised learning, a process where the model doesn't have access to any labels to guide its learning. In this scenario, the model learns to identify patterns or structures within the data without any explicit instructions. A prevalent technique within unsupervised learning is often termed as continual pre-training or unstructured FT. This method sees the FT process as an uninterrupted extension of the pre-training phase. In other words, instead of abruptly shifting gears from the pre-training stage to the fine-tuning stage, this approach treats the entire process as a continuous flow of learning. It starts with a saved checkpoint of the original LLM and trains it in a causal auto-regressive manner, predicting the next token. One major difference in comparison to actual pre-training is the learning rate. Usually, one would need a much lower learning rate when continuing the pre-training of the model to avoid catastrophic forgetting[8]. It is well known that LLMs store vast amounts of knowledge during their pre-training phase. Therefore, it makes sense to continue this process in order to inject knowledge into the model.

1.2.2 Retrieval augmented generation

Retrieval augmented generation (RAG)[9] is a method that significantly broadens the capabilities of Large Language Models (LLMs), particularly in tasks that require a deep understanding and usage of knowledge. This technique essentially incorporates external knowledge sources, like databases or encyclopedias, to supplement the model's internal knowledge acquired during pre-training. It has been shown in some research[10] that using a pre-trained embeddings model could lead to improved performance without the need for additional task-specific training. An embeddings model is a type of machine learning model that represents data by encapsulating information into dense representations in a multi-dimensional space. These vectors, or embeddings, capture the semantic properties of the data. By using pre-trained embeddings, the model could

leverage existing representations of knowledge and apply them to new tasks. The idea is that given an auxiliary knowledge base and an input query, RAG architecture is applied to find documents within the knowledge base that resemble the input query. These documents are then added to the input query, thus giving the model further context about the subject of the query to handle domain-specific tasks. Despite the added complexity of the retrieval process, the use of pre-trained embeddings simplifies the overall training process and enhances the model's performance.

The two methodologies discussed above offer unique benefits in terms of knowledge injection for LLMs. Fine-tuning, as a strategy, has shown to consistently improve the performance of LLMs across all entities. The performance enhancements are particularly demonstrated when there is a moderate amount of data available. On the other hand, insufficient data available may be challenging and incur overfitting, where the model learns to perform well on the training data but fails to generalize to new, unseen data. While RAG allows the model to access and incorporate a broader range of information than what is available in its pre-training data by integrating an external knowledge retrieval step into the model's generation process,. This ability to tap into external sources of information could significantly enhance the model's performance on knowledge-intensive tasks. Interestingly, it has been observed that the combination of RAG and fine-tuning could lead to even greater performance boosts for LLMs. The merge of both strategies could help the model leverage the more outstanding outcome, the task-specific refinement offered by fine-tuning while broader external knowledge access provided by RAG. This powerful combination could enable the model to handle a wide range of tasks more effectively, from general queries to highly specific and complex questions. One thing worth noting is that the critical importance of synthetic data quality is mentioned as a great factor to sway LLMs performance with any of the mentioned strategies[11][12].

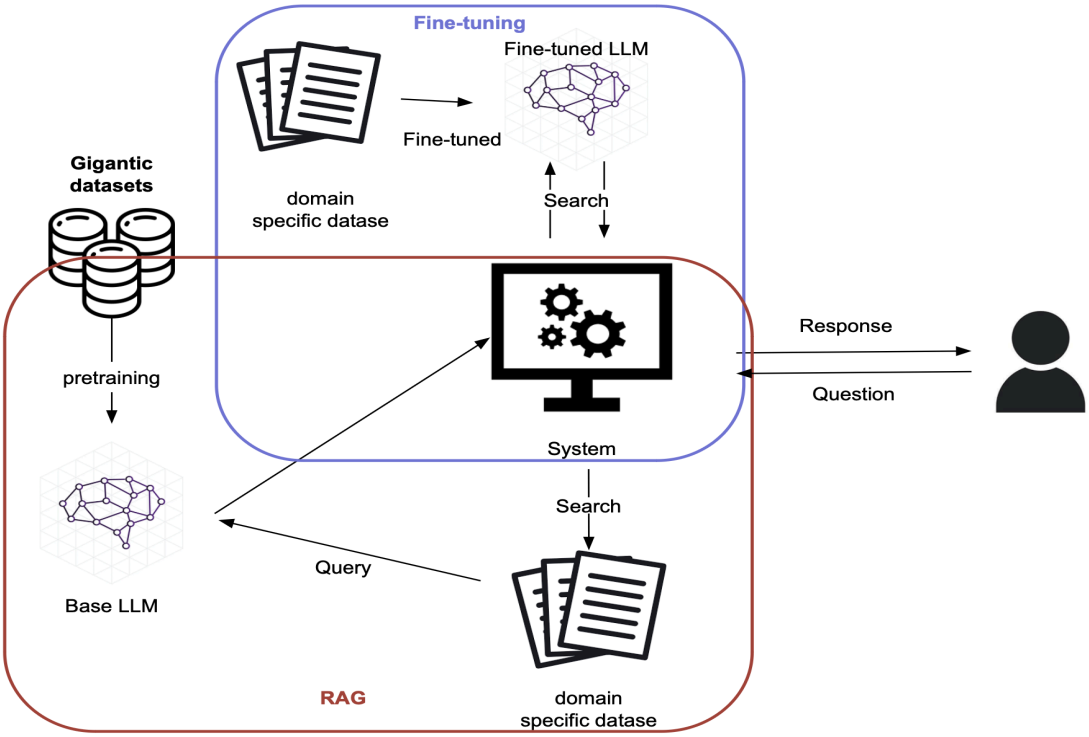


Figure 1 – Illustration of fine-tuning and RAG in the application

The selection of approach between fine-tuning and RAG could greatly influence the system on the user side and developer side, illustration is shown in fig.1 :

- The selection of the wrong approach could result in: poor model performance on specific tasks, leading to inaccurate outputs.
- The increase of costs in computational costs for both training and inference if the chosen method isn't well-suited for your specific use case.
- Extra time and challenge spent on development and modifications if it is required to implement additional techniques.
- Postponements in deploying your application and reaching to users.
- A lack of model interpretability if an overly complex adaptation approach is applied.
- Challenges in transitioning the model into a production environment due to constraints related to size or computation.

1.3 Data augmentation

Limited labeled data often leads training models to overfitting and prevents models from understanding unseen datasets. Data augmentation shows the potential to alleviate this problem by automatically or manually manipulating the data to create additional augmented data. The first research for data augmentation could be traced back to the famous LeNet. Using random distortion on the training images, the MNIST dataset was expanded ninefold, allowing for better recognition of handwritten digits. It is significant to observe the "label preservation" attribute in the outcome of data augmentation, which describes a transformation of training data that preserves class information. Such methods have been extensively studied in the field of computer vision using techniques such as geometric/color space transformations[13][14], mixup[15], and also random deletion. Similarly, in NLP, there is certain difficulty due to the discrete nature of contextual meaning to follow the same rule to succeed in label-preserving textual transformation. In general, augmentation methods in NLP could be roughly sectioned into some phases: character level, token level, sentence level, document level and adversarial augmentation. In this work, character level, token level and sentence level approaches are mainly implemented.

1.3.1 Character-Level augmentation

The idea of adding synthetic noise to text-based applications is not new; however, it is mainly used for adversarial attacks or to develop more powerful models. Four types of synthetic noise are introduced at the orthographic level: character deletion, insertion, substitution, and substitution. Additionally, they presented a mixture of all noise types, which it is commonly called All Characters, by sampling from a distribution that is 60% pure (no noise) and 10% each type of noise. It has shown that adding synthetic noise to training data improves performance on naturally noisy test data (text with real misspellings, keyboard misspellings) without degrading performance on clean data. In these research[16][17], neural machine translation was used as a main tool to test on multiple languages as source such as French and Czech languages and English language was used as target language. It raised a profound conclusion that using appropriate amounts of noises and synthetic errors could enhance the performance and robustness of the model to a certain level.

1.3.2 Token-Level augmentation

Intuitively, if some tokens of a sentence are replaced by other tokens with the same meaning, the semantic meaning of the sentence remains unchanged. An easy way is to get synonyms as replacement words. Several methods based on single word replacement have been explored for data

enhancement. In particular, demonstrated simple single word replacement using a knowledge base such as WordNet[18][19] or in embedding space[20]. However, the improvement from this approach is usually small, and in some cases performance may even worsen. The main disadvantage is the lack of contextual information when retrieving synonyms, especially for words with multiple meanings and few synonyms[21][22]. To solve this shortcoming, pre-trained language models have shown an excellent work of grasping the contextual meaning, which began to be popular among data augmentation research, starting by[23] utilizing language models to generate versatile datasets from training datasets. However there is a struggle for this method to preserve the meaning of label, which have negative potential to change the contextual meaning, like "a excellent game with unbelievable result" could have new variant switching two nouns "a excellent result with unbelievable game" with correct grammatical structure but improper contextual meaning. With these kinds of mistakes in augmented datasets due to the confusion of the contextual meaning or sentimental meaning could be twisted, in result, it could break the robustness of the model and weaken the performance.

Further, in order to handle this problem, conditional BERT (cBERT), which extends BERT, using predicted labels to generate the next word token is proposed[24][25]. Along with cBERT, GPT2 augmentation method is proposed by using training datasets with predicted labels in fine-tuned models, and generating new sentences based on confidence score[26]. However, due to the use of a proprietary extension mechanism, generalization problems arise, meaning that this method is difficult to apply to different language models. Additionally, different sampling strategies with single-word replacement were examined. For example, [27] suggests computing a weighted average of possible word embeddings predicted by LM as replacement input, rather than selecting a specific word from candidates using a language model, since the average representation could improve the text by adding richer information.

While well-designed local modifications could preserve the syntax and semantics of a sentence, random local modifications, such as Removing certain tokens, inserting random tokens, replacing unimportant tokens with random tokens, or randomly replacing tokens in the same sentence could preserve meaning in practice. Different types of operations could be further combined, with each paradigm randomly enriching augmented sentences through insertions, deletions and substitutions. These noise injection techniques could be used effectively for training and show improvement when generating new samples with simple models trained on small training sets[28][29][30].

1.3.3 Sentence-Level augmentation

Paraphrasing, as a data enhancement technique, has been widely used in various NLP tasks. Because it typically provides more varied rich text with different word choices and sentence structures, while maintaining the meaning of the original text. The most well-known method is back-translation, which follows a pipeline that first translates the sentences into some intermediate language and then translates them back to create a paraphrase. Translations in different languages with different vocabulary and language structures could produce original meanings but with different lexical contents. In order to enhance the diversity of augmented data, sampling and noisy beam search could also be adopted during the decoding stage[30].

The conditional generation method generates additional text from the tag-based language model. Once the model is trained to generate source text for a given label, the model could generate new text. To ensure high-quality augmented data, an additional filtering process is often used. For example, in the text classification process, with the original examples prepended with their labels, and then augmented examples are created by assigning specific labels to a finely tuned model. Only

valid examples evaluated by a base classifier trained on the original data are kept for further use[31]. New answers are generated based on questions asked in Q&A and filtered by custom metrics such as accuracy and n-grams, the former metric is used for evaluation of perseverance of original phrase's meaning and the latter one is a collection of multiple successive items in a text document which is useful for text analytics such as evaluation of diversity of contextual text .

1.3.4 Adversarial data augmentation

Adversarial data augmentation[32] is a common technique in machine learning to improve the robustness and generalization capabilities of a model by introducing adversarial examples such as grammatically wrong sentences, misspelling or common keyboard mistakes during the training process. Adversarial paradigms are inputs deliberately designed to cause the model to make incorrect predictions that could not be detected by human observers. Adversarial examples are generated by perturbing the input data in a way that is imperceptible to humans but could lead the model to make mistakes. This is often done by adding small, carefully crafted perturbations to the original data. The generated adversarial examples are then added to the training dataset, effectively expanding the dataset with these challenging instances. This step is crucial for the model to learn to be more robust in the face of diverse and potentially misleading input. The model is trained on the augmented dataset, which now includes both the original and adversarially perturbed examples. The training process aims to make the model more resilient to subtle variations in the input, thus improving its generalization performance. Eventually, The trained model is evaluated on adversarial test sets, which consist of data intentionally designed to deceive the model. This evaluation helps assess the model's ability to handle unforeseen variations and enhances its robustness in real-world scenarios.

1.3.5 Few shot learning (FSL)

Deep learning has made significant progress in various data-intensive applications. However, the performance of deep models may be affected by limited datasets in downstream tasks. FSL, which also could be considered as meta learning, is a branch of science focused on developing solutions to problems with small samples. A typical use case for FSL is when it is difficult or impossible to obtain enough examples due to privacy, security, or ethical concerns, such as medical purposes. FSL is a vital technique in transfer learning in which research aims to use previously obtained knowledge to quickly generalize to downstream tasks containing only a small number of labeled samples[33]. By leveraging this technique, it has shown many promising results, among them the pre-trained model has been becoming the most popular way along with this approach by fine-tuning it with limited label datasets. Some other advancing methodologies are prompting-tuning[34] and meta learning.

Despite recent developments in these promising approaches, they still suffer from some serious limitations. For example, prompting design is a laborious task that requires expertise and experience with a great amount of trials and errors[35]. On the other hand, FSL, meta-learning, is suffering from problems such as training instability and the long-term training process depends on the quality and quantity of training datasets, and also hyper-parameters easily sways the training quality and final outcome[36][37]. Moreover, these approaches require deep machine learning knowledge, knowledge of complex model architectures and training strategies that are not available to conventional practitioners and general developers. In the conclusion, these pipelines may not be suitable to be generalized and common in community, while data augmentation is way more instinct and straightforward method for general practitioner, in addition, data augmentation could be combined with above methods and relieve the difficulty, such as limited source datasets and possibly outperform outcome by solely using one of these pipeline.

1.3.6 Data augmentation using pre-trained language model

Data augmentation using pre-trained language models has become a trend in the field of NLP. These language models have been extensively studied over the past few years, resulting in the development of several effective models. The main models fall into three categories: Auto Encoders like BERT, Autoregressive models like GPT family, and sequence-to-sequence (seq2seq) models like BART[38] and T5[39].

The pre-training process for these models could be split into two main settings: AE and AR. In the AE setting, certain tokens within a sentence are masked, and the model is trained to predict these masked tokens. This is done using the context provided by the unmasked tokens, thus enabling the model to understand and learn from the contextual meanings within the sentence. On the other hand, in the AR setting, the model is trained to predict the next word based on the previous context. This allows the model to generate coherent and contextually accurate sentences, making it particularly useful in tasks like text generation, text completion, and translation. Recently, pre-trained seq2seq models have gained significant attention. Unlike traditional AE or AR models, seq2seq models are designed to transform an input sequence into an output sequence. This makes them extremely versatile and capable of handling a wide variety of tasks. In the context of data augmentation, seq2seq models are often trained for AE denoising tasks. This involves adding noise to the input data and training the model to recover the original, noise-free data. In several data augmentation studies, seq2seq models have shown even stronger creative and precise abilities than other model structures. By generating diverse and contextually accurate augmentations, these models could significantly improve the robustness and performance of downstream NLP tasks.

1.4 Hypercomplex classifier

There has been a research trend of how to increase the performance of representation learning in multiple domains for a couple of years. As a consequence, the model is getting better by time, however in the meanwhile, the parameter size of the model is exponentially increasing at cost. Nowadays, training an efficient model in any domain has been impossible in a single device due to the large corpus of training datasets.

To solve the current problem, the research for minimizing parameters is getting more attention year by year. Among them, hypercomplex is also paid attention to for its flexibility and potential. Most of the related jobs are still at infancy level, but with strong theoretical support[40], work of batch normalization and initialization based on quaternion rules to convert to an equivalent mechanism of real-valued neural networks has been proven to be achievable. Along with this trend, speech recognition is presented by using LSTM and RNN with quaternion algebra[41], in the meanwhile, Quaternion CNNs was applied to image classification, denoising tasks and sound detection[42][43]. In the NLP aspect, there have been magnificent works for quaternion embedding of knowledge graphs[44]. All mentioned works so far are based on quaternion domains in quaternion algebra. In some levels, by using real-value representations, it could be hard to capture value statistically, while for quaternion representations it is more accessible to gain these properties. Moreover quaternions domains effectively fix the problem of vanishing gradient which has been a fetal drawback for RNN. Also, Asymmetry, as one property in quaternion, has shown to be useful in areas such as relational learning and question answering[45].

In recent advanced research, transformer, as most common and popular model architecture in community nowadays, was presented as quaternion version[46], followed by research of

parameterized hypercomplex multiplication(PHM)[47] to overcome the struggles of quaternions domains which has only limited capability to handle dimension with possession of factor 4 in hypercomplex space, which hinders application and research in NLP research. PHM layers are generalized in any model, frankly speaking, it could be considered as a light-weight fully-connected layer with support of Kronecker products. With support of Kronecker products, parameters are decomposed without losing the value of itself.

Proposal research is intended to be a connection between previous works and the enrollee support system in this work, which could be regarded as a hybrid of existing pre-trained model and potential lightweight approach in current research. By using the property of sharing weights via Kronecker products and Hamilton products during training in hypercomplex neural networks, it is expected to be able to optimize the performance of pre-trained discriminative NLP models such as BERT models in the meanwhile parameter size could be greatly reduced by replacing the fully-connected layers with PHM layers.

1.4.1 Background on hypercomplex algebra

Hypercomplex neural networks rely on a hypercomplex number system based on a set of hypercomplex numbers H and their corresponding algebraic rules to form addition and multiplication. These operations must be carefully modeled due to the interaction between imaginary units, which may not behave like real numbers. For instance, with n = 4 there are quaternions whose multiplication is not commutative, while with n = 8 there are octonions and dual quaternions whose multiplication is not commutative and associative, etc. Different multiplication rules are applied due to the heterogeneous interaction of imaginary units based on Cayley–Dickson construction. There are various subsets of the hypercomplex field, well-known subsets are complex numbers, quaternions, octonions, etc. The quaternion field is one of the most popular fields in neural networks due to the Hamilton products. In hypercomplex linear transformation layers there could be fewer degrees of freedom, allowing the model to be compressed, for example, in quaternion domains parameters are reduced to a ¼ of original parameters due to total size of different parameters, four distinct parameters are applied in weight matrix while in real-valued domain all parameters are distinct which could result in 4x4 of different parameters, moreover it has been widely proved in many research showing that it has potentials to have better weights sharing in order to gain better connections between embedding spaces[48]. However, only numbers with factor 4 limits the flexibility of NLP model design, which hinder the further research of hypercomplex neural networks in this field, despite there have been many excellent results which show their own applicability in many areas.

1.4.2 Hypercomplex algebra

Hypercomplex neural networks use a system of hypercomplex algebra based on a set of hypercomplex numbers H and their corresponding algebraic rules to proceed mathematical operations.

Generic hypercomplex formula could be written as below :

$h = h_0 + h_1 i_1 + \dots h_n i_n$ n = 0,1, … , n.

h denotes real value and i denotes imaginary unit. Straightforwardly by having different imaginary units in the formula, there could be different domains of hypercomplex algebra.

1.4.3 Quaternion algebra

A Quaternion Q ∈ H is a hypercomplex number with three imaginary components, which is mostly widely employed domain as follows:

$Q = r + xi + yj + zk$,where $ijk = i^2 = j^2 = k^2 = -1$ and noncommutative multiplication rules apply : $i * j = k, j * k = i, k * i = j, j * i = -k, k * j = -i, i * k = -j$.
From the above expression of Q, r stands as real value and on the counterpart x, y, z are real numbers that represent the imaginary components of the Quaternion vector Q. By adding more imaginary units, different hypercomplex domains could be implemented. It should be careful to do modeling due to different mathematical rules among imaginary units compared to real-valued numbers.

Briefly speaking, Quaternion algebra adheres to most mathematical rules similar to real-valued algebra, but there are certain nuances and patterns that need careful attention during implementation. Starting from addition, Pairwise calculations are proceeded component-wise according to the expression. For example, Q and P are two quaternion vectors, intuitively addition are done by pairs based on imaginary components. By swapping plus sign to minus sign, subtraction could be also expected following the same rule.

Multiplication in the context of quaternion or hypercomplex numbers is a bit more complex than addition and subtraction, primarily due to the non-commutative nature of quaternion multiplication. The multiplication operation is governed by the associative and distributive laws among expressions, but special attention must be paid to the order of multiplication due to a predefined rule in hypercomplex domains. On the other hand, Scalar multiplication in the realm of quaternion algebra is straightforward and parallels the concept in real-valued algebra. It involves multiplying each component of the quaternion by a scalar.

The conjugate of Q is by flipping the sign between plus sign and minus sign of imaginary units while sign of real value remains same

Normalization of quaternion is intuitively easy like real-value algebra by dividing matrix with square root of dot products taking all values without quaternion expression.

The Hamilton product, which represents the multiplication of two Quaternions Q and P , it is responsible for information sharing between real-value unit and imaginary units, representing as a core of mechanism, formula defined as:

$$\begin{bmatrix} Qr & -Qx & -Qy & -Qz \\ Qx & Qr & -Qz & Qy \\ Qy & Qz & Qr & -Qx \\ Qz & -Qy & Qx & Qr \end{bmatrix} \begin{bmatrix} Pr \\ Px \\ Py \\ Pz \end{bmatrix}, \quad (1)$$

Q in matrix are learnable parameters, and P are input layer input. However, practically, it is hard to limit input size or output size as 4. Therefore in order to apply Hamilton products with quaternions units. It is necessary to divide all inputs into 4 segmentations, which in another word input size must be divisible to 4 to leverage quaternions units with Hamilton products into neural networks. In fig.2, it shows the interaction between inputs and outputs. Compared to real-valued matrices, quaternion shows the advantage of how to do parameter saving, which allows matrices to reuse the same weight parameters by changing plus and minus signs.

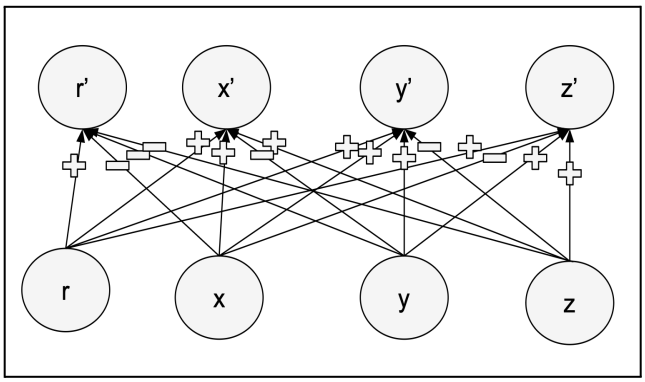


Figure 2 – Weights interaction between input and outputs denoted by arrows for the each pairwise connections, positive or negative weight are shown with + and - signs

1.4.4 Parameterized hypercomplex multiplication layers (PHM)

PHM layer as a core of mechanism to reduce parameters in model construction, which is constituted by constructing the sum of Kronecker products of the weight tensor, which encapsulates and organizes generalization of the vector outer product in real space. With support of Kronecker products matrix, the parameter could be decomposed (parameters reduction) without losing original value. The formula of this layer is defined below with comparison of traditional fully-connected layer:

$$Y = W * x + b, \tag{2}$$

$$Y = H * x + b, \tag{3}$$

Each symbol denotes as follow $H \in R^{O*I}$, $W \in R^{O*I}$, $x \in R^{I}$, $Y \in R^{O}$, $b \in R^{O}$. Fully-connected layer has been fundamental part to many neural network architecture for years, it is necessary to be noted that size of the weight parameters in this layer is $O(O * I)$ in Fully-connected layers, formula shown in (2). Formula of PHM layer is shown in (3), H is constructed from the sum of Kronecker products shown in fig.3, which stand as the most vital part in this mechanism in order to help generalization of outer product that could be parameterized by n, a hyperparameter predefined by user before the process,

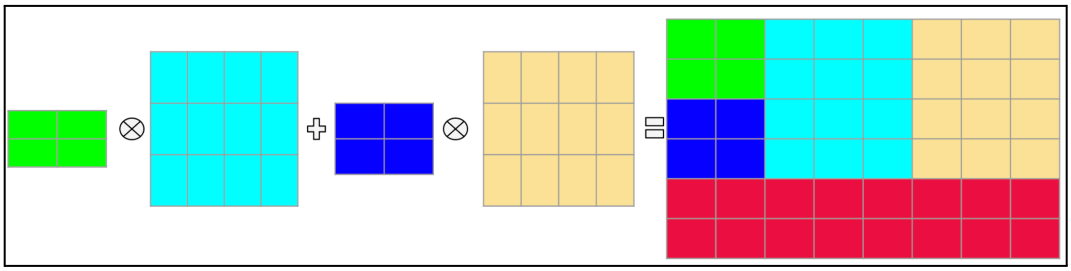


Figure 3 – Abstract illustration of PHM layers, predefined n = 2, input is 8 and output is 6, red part is reduction size to be expected

$$a \otimes Y = \begin{bmatrix} a_{11}Y & a_{12}Y & \cdots & a_{1n}Y \\ a_{21}Y & a_{22}Y & \cdots & a_{2n}Y \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}Y & a_{m2}Y & \cdots & a_{mn}Y \end{bmatrix} \in \mathbb{R}^{mp \times nq}, \quad (4)$$

H in (3) could be denoted as (4), basically is a block matrix which is represented by $a \otimes Y$, where defined as $Y \in R^{p \times q}$, $a \in R^{m \times n}$ and a_{mn} stands for elements by m^{th} column and n^{th} row. The n could be defined by user's intention to transform to real-value domain or hypercomplex domain, by setting it to 2, it would turn to complex domain, with n = 4 it could become quaternion domain, such flexibility extend its application based on different dimension, moreover it solves the restriction possessed by quaternion neural network, which could only handle limited dimension. However one thing should be noted that n could only be set to a number which should be divisible to all hyper-parameters in the same time, in short the dimension of input, hidden layers and output layers should be divisible to n. In case there is no divisible number between hyper-parameters, n would be assigned 1, which means the PHM layer would be transformed to real-valued domain by setting n to 1, in this circumstance, a real-valued unit is not feasible to share information with hypercomplex units.

Also it is possible to switch to domains, which do not have algebra rules such as 2^x $x \in 1, 2, \ldots \infty$, PHM layer has the capacity to learn information from datasets in the training process[49] to adapt patterns. Abstraction is shown in APPLICATIONS F for further understanding of the utilization.

2 OVERVIEW OF IMPLEMENTATION

Implementing the proposed system is indeed an intricate, multi-faceted process. It involves various stages of design, development, and optimization. Each stage is of paramount importance and demands meticulous planning, strategic execution, and continuous evaluation to ensure the overall integrity and functionality of the system. In the initial stages, the system's design involves refining classifiers to boost performance. This is a crucial step that involves a deep understanding of the data, the ability to identify key patterns, and the application of suitable algorithms to create models that could accurately classify information. This tweaking process is done iteratively, with the models being tested and refined based on their performance accuracy. Following this, the data augmentation phase is initiated to increase the depth and diversity of the dataset. This involves creating new data points from the existing data, either by incorporating minor changes or by generating synthetic data. This process is crucial to ensure that the models are trained on a comprehensive dataset that could cover a wide range of possible scenarios, thereby enhancing their predictive capabilities. The recommendation model is implemented for system optimization. This model uses machine learning techniques to analyze user behavior and preferences, and provide personalized recommendations. Implementing such a model is a complex task that requires a deep understanding of both the data and the users, and it plays a crucial role in enhancing user experience and engagement. This entire process, while complex and demanding, is imperative to ensure that the system is robust, efficient, and capable of handling the tasks that were previously managed manually by the admissions team. These tasks involve user interaction, knowledge enrichment, and maintaining a sustainable workflow. Automation of these tasks not only reduces the workload on the admissions team but also enhances efficiency of interaction, improving the overall quality of the service provided. The end goal of this entire process is to develop a system that is innovative, feature-rich, and capable of addressing the challenges of handling large volumes of queries and tasks in the education sector. While the system has been designed with a specific dataset, the approaches and methodologies employed have been developed with a broader perspective. They have been designed to be generalized and adaptable so that they could be applied to other similar models and datasets.

2.1 Cycle and structure of system

In general, the universal life cycle could be designed as follows: requirement, specification, development, testing, deployment and maintenance shown in fig.4. The requirements phase is a critical stage in the development process of a system. At this phase, it's essential to gather comprehensive and precise requirements from the service providers who would be using the system. These requirements form the foundation on which the entire system would be built, so understanding them clearly and completely is crucial. The primary objective of this phase is to comprehend the underlying business problems that the system is intended to solve. This involves detailed discussions and brainstorming sessions with the service providers to identify the system's necessary features and functionalities. Understanding what the system should do forms the basis of its functional requirements. Moreover, it's also important to predefine the potential end-users of the system. Who would be using the system? What are their needs and expectations? Answering these questions would help in defining the system's usability requirements and ensure that the system would be user-friendly and meet the users' needs. Once the requirements are clearly defined and understood, the project moves into the specification phase. This phase involves transforming the collected requirements into a detailed system design. It's like creating a roadmap for the development process, where the flow of the system is mapped out from the database to the user interface. During the specification phase, the structure of the database is defined, taking into account

how data would be stored, retrieved, and managed. The user interface is designed with a focus on usability and a positive user experience, ensuring that it is intuitive and easy to navigate. The interactions between different system components are also planned out during this phase.

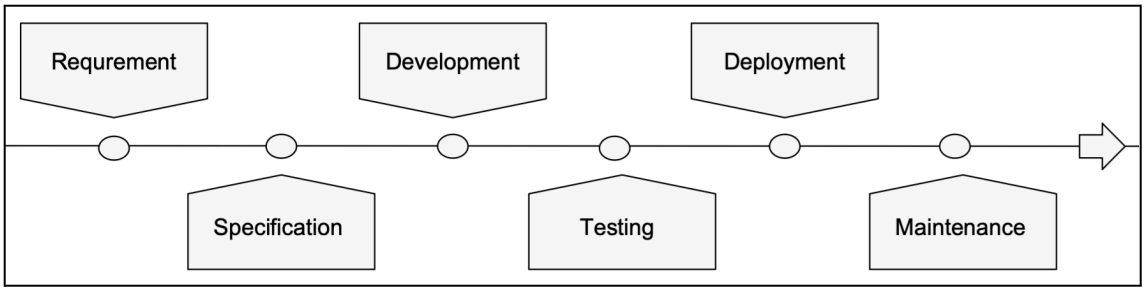


Figure 4 – Life cycle for current work

In a knowledge-based system, the database serves as the vital core that ties all functionalities together. It's the repository where the system's knowledge resides and from where it draws responses to user queries. Therefore, the design of the database should be paid extra attention. It should be structured in a way that facilitates efficient storage, retrieval, and management of data. The content of the database should be thoughtfully curated to include knowledge that is both interesting and useful to the system's users. This could encompass a broad spectrum of information, depending on the system's domain. It could range from frequently asked questions and their answers, to complex data sets for analysis. The goal is to ensure that the system has at its disposal, a wide array of information to generate relevant and engaging responses to user queries. Following the specification phase, the development phase is to implement the design into reality. This phase involves coding and building the system based on the design outlined in the specification phase. It's a meticulous process where the system begins to form the shape. The developers translate the designs into functional code, creating the various components of the system and integrating them together. Once the system has been built, it enters the testing phase. This phase is crucial to ensure the quality and reliability of the system. The system is tested for defects and discrepancies, with the goal of identifying and fixing any issues before it is deployed. Testing could be a time-consuming process, given the complexity of the system and the multitude of scenarios it needs to handle, automated testing is preferred over manual testing due to its efficiency and the ability to cover a wide range of test cases. Automated tests could run around the clock without human intervention, making it possible to catch bugs early and fix them promptly. This not only improves the system's quality but also accelerates the development process.

The deployment phase represents a significant milestone in the system development life cycle. In this phase, the system, having been thoroughly tested and deemed ready for use, is moved from the development environment to the host or production environment. This is where the system is put into operation and begins to serve its intended users in a real-world context. This transition from a controlled development environment to the unpredictable real-world environment could be quite complex. It involves careful planning and execution to ensure that the system functions work as expected. It's not uncommon for potential issues to only surface when the system is exposed to the full variety of real-world use cases. For this reason, once deployed, the system needs to be monitored constantly. This ongoing monitoring allows developers to spot any issues early, understand usage patterns, and gauge system performance under real operational conditions. Following deployment, the system moves into the maintenance phase. It involves regular monitoring

and modifying the system as needed to correct any abnormalities and to ensure its continued operation. The maintenance phase is not just about fixing bugs or issues that arise; it's also about updating the system to meet evolving user needs, accommodating changes in the operating environment, or improving the system's performance based on feedback and usage data. In essence, maintenance is about ensuring that the system remains relevant, useful, and efficient in the face of changing needs and conditions.

Table 1 – Initial evaluation in requirement phase for current work

User Identification	Prospective students, admission staff, and administrators
Purpose	Automatic response
Scalability	Yes
Sustainability	Yes
Dataset	82

In current work, the mentioned life cycle was strictly followed, only first three phases are mentioned in the content, further phases would be mentioned in the further research for evaluation purposes to make judgment of further development and adjustment based on various conditions. Details of the requirement for the implementation of the intellectual enrollee system are listed in table 1 to reflect the implementation.

In the specification phase, the structure of the system is roughly sectioned into 3 parts as follows: user module, conversational module and knowledge module. In the user module, telegram platform is chosen due to the fact that it is a global preferred application for socializing and information retrieval, moreover it could meet the requirement where the target users of this system is set to foreign students. For better user's experience of conversational intellectual systems, telegram, as originally designed for conversational purposes, could be friendly to use for users. Users have choices between textual message and verbal message to have dialogue with the system.

Conversational module is where the users and system interact by processing any kind of message as textual message. In this module, a recommendation model is implemented as a strategy to guide users to ask questions from previous user's options, therefore less mistakes could be made by the system in the meanwhile unexpected but interesting questions could be found by users, which could possibly trigger more interest of users.

The knowledge module is considered to be the core of the system, it contains the knowledge base which enables the discriminative capability to distinguish the different inputs. The knowledge module does not just handle the classic CRUD process, the main task is to harness a semantic engine where NLP models are trained with storage datasets in cloud service where a sustainable life cycle could be achieved by relentlessly processing CRUD, augmentation module and retraining NLP model. Beside mentioned modules for users purpose, modules for administrative purpose are also implemented for any modification and system monitoring. Also there is a mechanism to directly correct the unrecognized response and prevent distorting the knowledge base and users could receive corrected answers directly when they activate the chat on the platform next time.

2.2 Baseline datasets

The selection and utilization of datasets play a crucial role in developing an effective and intelligent university admission system. They serve as the backbone for training the system, ensuring its accuracy, and continuously improving its performance. It's important to choose the right

datasets and use them effectively to achieve the maximum benefits. When it comes to selecting datasets for an intelligent university admission system, categorization could be broadly divided into two: those used for system development and those used for experimental purposes.

The first category, system development datasets, should contain information related to academic topics. These are instrumental in training the system's domain-specific models. They serve as the foundation for the system's understanding and could include data like course descriptions, program details, and university information. While the second category, experimental datasets, are used to test and compare the performance of the proposed with existing approaches. By analyzing the system's performance on these datasets, developers could benchmark the proposed methods against existing methods and make necessary improvements, datasets are listed below :

(1) ITMO-dataset is the main dataset to this research work for development of intellectual systems. It is a private dataset, collected from the official website for admission. Datasets are as a form of question-and-answering with 23 labels at the moment and mainly focus on information of admission and enrollment for foreign students. However due to its extremely scarce number of datasets, it has been a struggle to train a robust model from scratch or fine-tune with existing language models. Therefore it has become an additional task to motivate this research work. The topics contained are related to admission exam, scholarship and other educational related topics.

(2) SNIPS Natural Language Understanding test is a dataset of over 16,000 crowdsourced queries distributed across 7 user intents of varying complexity. These intents vary in complexity, and include tasks such as SearchCreativeWork, GetWeather, BookRestaurant, and PlayMusic. The task of determining an intent from a group of established intents based on a single input is a key element of NLP and is often approached as a classification challenge. This dataset serves a similar purpose as ITMO database does, which demands a model to have capability to look for intention of text from input contexts.

(3) SST-2 is a dataset for emotion classification in movie reviews, marked with two labels (positive and negative). It's a corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language, which is pivotal for understanding and building models for sentiment analysis. With neutral sentences discarded. This binary classification gives the SST-2 dataset a unique place in text classification problems, making it an effective tool for training much more robust models to understand and classify sentiments.

(4) TREC is a question classification dataset containing 5500 labeled questions from the training set and an additional 500 questions from the test set with 6 different classes, providing extensive, varied, and rich resources for text classification and information retrieval tasks. They have been instrumental in various research studies and practical applications, from question classification to pandemic-related research and clinical trials. Their diverse labels and wide range of sources make them a valuable tool for researchers in the field of NLP and beyond.

Datasets of (2) (4) are provided by[24] and datasets (3) originate from Kaggle. In the experiment, slight modification has been made compared to the previous works, numeric labels are used instead of textual labels in consideration that it is unrealistic to assign one related vocabulary to all new topics.

Datasets from (2) (3) (4) are prepared for experiments in data augmentation and hypercomplex classifiers in order to validate that the proposed methods could show comparable performance compared to existing methods. Thus, customized approaches could show persuasive results to handle unique circumstances in this work. The size of each dataset is listed in table 2.

Table 2 – Counts of each original datasets sectioned into 3 groups in the experiment (not including ITMO datasets). In extrinsic evaluation, these datasets are sampled 10 to simulate low-resource regime, while in intrinsic evaluation, train and test are combined as training datasets, test datasets are replaced with augmented datasets

	TREC-6	SST2-2	SNIPS-7	ITMO
Train	5406	6228	13084	82
Dev	546	692	700	
Test	500	1821	700	

2.3 Knowledge injection approaches selection

As mentioned in section 2.2, knowledge injection approaches are classified into fine-tuning and RAG, while both RAG and fine-tuning are effective techniques, their suitability would depend on the specific application, the nature and size of the data set available for model development.

RAG, known for improving accuracy in large models, is highly effective in instances where data is contextually relevant, such as in the interpretation of unstructured dataset. The low initial cost of creating embeddings, which are vector representations of data, makes RAG an attractive option. However, it is important to consider that the input token size could increase the prompt size, and the output token size tends to be more verbose and harder to steer.

While fine-tuning offers a precise, succinct output that is attuned to brevity. It is highly effective and presents opportunities to learn new skills in a specific domain. However, the initial cost is high due to the extensive work required to fine-tune the model on a new dataset. In addition, fine-tuning necessitates minimal input token size, making it a more efficient option for handling large data sets.

Apparently, both approaches could succeed to develop one decent intellectual helper in this specific-domain of knowledge. The factors for approaches selection are summarized by taking into consideration all conditions with given datasets and evaluation in the requirement phase before developing enrollee intellectual support. From the aspect of datasets, although both approaches could handle such well-structured labeled datasets, fine-tuning could offer more tailored model behavior. In scenarios where such data is limited, a RAG system provides a more robust alternative. On the other hand, considering the perspectives of low knowledge update frequency, fine-tuning offers an advantage due to the fact that it is not necessary to retrieve relevant information from external knowledge bases, additionally it could produce results more quickly in deployment phase, making it ideal for immediate response with fewer latency.

There is another issue worth noting, fine-tuning is scalable, both in terms of dataset size and computational needs. Due to its inherent design, the model could be easily expanded or reduced to fit specific hardware requirements, The scalability of RAG is a bit of a mixed bag. While the generation component could be quite scalable, the retrieval component often requires significant computational resources, particularly when dealing with large and growing databases.

Concluding all factors above, following sections of this work would focus on details of implementing a fine-tuning approach of LLMs, including system workflow design, text augmentation approach and model modification optimization by using hypercomplex, detail is shown in table 3 below.

Table 3 – Approaches comparison between fine-tuning and retrieval-augmented generation, condition of current work is outlined in red, which would be used for making final decision of selecting more optimizing approach for system development

Methods	**Fine-tuning**	**RAG**
Use case	1. To pre-train model to a specific task or domain	1. To incorporate specific, up-to-date, or domain-specific knowledge from large datasets
Model adaptation	Yes	No
Knowledge flexibility	No	Yes
Inference speed	Fast	Slow
Data format	Structured datasets	Any
Update frequency	Less	More

2.4 Used tools and libraries

The system's architecture, as discussed earlier, includes three essential components. The primary interaction point between the system and the user is the user interface, which constitutes the first section. To fulfill this role, Telegram, a worldwide well-known conversation platform, is selected. It is more than just a messaging application; it's also a robust platform with a user-friendly, clear graphical interface, making it simple for users to navigate through and interact with the system. For developers, it provides a collection of adaptable, user-friendly tools and functions through the "telebot" library. This library is vital in realizing the system's workflow within the Telegram framework, enabling seamless communication between the user and the system.

The system's second component is the dialogue module. This is the section where interaction takes place between users and digital assistants. It's the essential framework that allows the system to engage in significant conversations with users, comprehend their inputs, and deliver suitable replies. The design and execution of this module call for thorough consideration of multiple aspects, such as the diversity of user inputs, the complexity of responses, and the requirement for adaptability in various dialogue contexts. This module attests to the system's resilience and versatility, equipping it to tackle a broad spectrum of conversational obstacles.

At the core of this system is the knowledge module. This module is designed to facilitate the automatic circulation for the retraining process, supporting the efficient operation of the system while minimizing the need for manual intervention. It leverages NLP techniques, a branch of artificial intelligence that enables computers to understand, interpret, and generate human language. To construct the deep learning model, such as the NLP model, it requires a robust computing framework.

Table 4 – Lists of all applied libraries in system with purpose and reason

Name	Purpose	Reason
telebot	To serve as front-end interface	Graphical interface is clean and intuitive for users
Pytorch	Deep learning framework	GPU acceleration
Transformers	Seek NLP models	Platform containing multiple pre-trained NLP models with multiple kinds of tasks
NLTK	Pre-processing, such as tokenization	Well-written function for english text
scikit-learn	Post-processing for evaluation	Efficient tools for machine learning
matplotlib	Visualziation for numeric statistics, such as visitors and model accuracy	Library for creating static, animated, and interactive visualizations
apscheduler	Activation of retraining periodically	Simple, human-friendly syntax
SpeechRecognition	Conversion from audio to text	Enable audio message input for NLP model
Pandas	Understanding the insight from collected datasets of visitors and its query	Fast, flexible data analysis and manipulation tool
Pymongo	The bridge to database	To retrieve the data with dictionary-like syntax

“PyTorch” is selected as an open-source machine learning library that is ideal for this purpose. One of PyTorch's distinguishing features is its dynamic computation graph, which offers flexibility in constructing complex structures, unlike the static computation graphs found in other frameworks such as TensorFlow. The dynamic computation graph allows the network behavior to be programmatically altered during runtime. This provides a significant advantage in terms of flexibility and user-friendliness. PyTorch also offers a comprehensive array of tools and libraries for deep learning, crafted to exploit the computational abilities of Graphics Processing Units (GPUs), essential for managing the complex computations involved in processing and analyzing unstructured textual data. It supports an extensive array of neural network architectures and offers functionalities for data loading, gradient computing, and etc. Furthermore, PyTorch is created to seamlessly integrate with the rest of the Python ecosystem, including the scientific computing libraries NumPy and SciPy, rendering it an excellent tool for scientific computing tasks. PyTorch's efficiency is another vital aspect. It's designed to be effective in both research and production environments. It offers an imperative and Pythonic programming style that supports code as a model, simplifies debugging, and aligns with other popular scientific computing libraries, while remaining efficient and supporting hardware acceleration.

Besides PyTorch, the system also leverages "HuggingFace", renowned for its dedication to progressing and democratizing artificial intelligence via open-source. This is reflected in the accessible design of their library and the availability of their resources to the broader data science and AI community. One of the prominent features is the simplification of data handling. HuggingFace also supports a broad range of models and datasets for NLP tasks. This includes models for tasks such as text classification, named entity recognition, and question answering. The library offers pre-trained models that could be fine-tuned for specific tasks, thus saving researchers a considerable amount of time and computational resources. By utilizing the "transformers" API, the

fitting NLP model could be downloaded to a local device, making it portable and easy to incorporate into the system.

In addition to the aforementioned libraries, "NLTK" and "scikit-learn" are two major libraries in the Python ecosystem frequently utilized in the data science and machine learning field. NLTK serves as a premier platform for creating Python applications that handle human language data. It offers user-friendly interfaces to over 50 corpora and lexical resources such as WordNet. Moreover, NLTK encompasses a collection of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning. It also provides Wrappers for robust, industrial-grade NLP libraries. NLTK's functionality and modularity make it an excellent tool for research and prototyping. NLTK allows users to easily extend its functionality and customize existing modules, making it adaptable for various NLP-related tasks. On the other hand, scikit-learn is a machine learning library for Python, built on top of NumPy, SciPy, and matplotlib. It is an industry-standard for most data science projects due to its efficiency and ease of use. Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface. It includes modules for classification, regression, clustering, dimensionality reduction, model selection, and preprocessing, among others. Scikit-learn is mainly focused on model building and is a powerful tool for predictive data analysis and statistical modeling. While both libraries could be used for similar tasks, the community around scikit-learn focuses more generally on machine learning. In comparison, NLTK is more specialized, with an emphasis on tasks related to human language data. It's important to note that while scikit-learn has good documentation and resources, the coverage of NLP-related topics may not be as comprehensive as in NLTK.

Beside the mentioned core library, For better user's experience purposes such as voice message, statistical graph demonstration, some extra libraries also are implemented, table 4 shows the details.

2.5 Flow of systems and diagrams

The system's flow, shown in fig.5, is initiated by the built-in "start" button available in the telebot API, which, when clicked, activates the main menu of the system. This interaction is first contact to the whole system which facilitates a seamless transition into the system's core functionalities. It's designed to be user-friendly and intuitive, providing users with an easy-to-navigate platform right from the first sight. The system's functionalities, particularly those pertaining to the Q&A segment, are categorized into two critical sections. These sections are referred to as "Main Questions" and "Writing Questions". The "Main Questions" section is designed to provide answers to the most frequently asked questions in the admission period. It's a quick and efficient way for users to get answers to their queries without the need for typing input or making questions. All it requires is a simple click on the corresponding buttons, and the system fetches and presents the needed information. On the other hand, the "Writing Questions" section serves a different purpose. It acts as an artificial interlocutor, capable of processing text in the format of natural language. This section is designed to mimic a human-like interaction, providing a more personalized and engaging user experience. The system is coded to understand and respond to user queries effectively, ensuring that every interaction feels natural and human-like. After the user interacts with the system, either by selecting a question from the "Main Questions" or by typing a question in the "Writing Questions" section, the system proceeds to generate a response. The response generation process involves the use of the knowledge module, integrated with NLP models, currently the RoBERT model is used. The system uses these models to predict the most suitable response to the user's question. However, the system doesn't stop at providing a predicted

response. It goes a step further by soliciting feedback on the quality of the prediction. This is a crucial part of the self-correction mechanism, where the system learns from the user's feedback. If the feedback is positive, the current input is updated in the system's database. If not, the system provides two more possible predictions for the user to choose from. This feedback loop could be continued by selecting a button until the system provides a response that the user finds satisfactory. If the system fails to provide a satisfactory response, the user's question is stored in an "unknown" database. This database serves as a repository for unanswered questions, which could then be addressed by the admission group in the future.

Regardless of the nature of the feedback received by the system, the activation of the recommendation mechanism is a critical next step. This mechanism, powered by the RoBERT model or sequential models like SASrec[50], plays a pivotal role in guiding users to formulate their subsequent queries. It presents users with options that have been meticulously calculated and predicted by these models. This process is continuous, with the system persistently providing options until either there are no more options left or the user opts to customize their questions. By implementing this approach, the system aims to minimize mis-prediction by the models, thereby enhancing the overall reliability and credibility of the entire system. When discussing the use case difference between the two models for recommendation, it's essential to understand the context in which they operate. Specifically, in the early stages of system usage, when there isn't sufficient user interaction data to train a sequential prediction model, NLP discriminative models, RoBERT in this system, take the role. These models are designed to handle the task of recommendation based on the limited data available at this stage, ensuring that the system continues to provide accurate and relevant responses from given customized input from users. The SASrec and NLP discriminative models, though used interchangeably depending on the stage of system usage, have distinct operational differences. SASrec models focus primarily on the sequential behavior of users. They employ self-attention mechanisms to predict the next item in the sequence, taking into account the order of user interactions. This sequential prediction capability makes SASrec models particularly useful in situations where the order of actions or questions is of importance. Contrastingly, NLP discriminative models operate a bit differently. They focus on providing potential items of interest based on the most recently received text from the user. These models prioritize the immediate context of the user's interaction, making them more suited to situations where the current context is more important than the sequence of actions.

On the other hand, in order to access the admin mode of the system involves a different set of steps. This admin mode requires the input of specific commands, followed by the entry of a predefined password. The password is an essential security feature that ensures only authorized individuals could access the admin mode. To further enhance this security measure, the password is modifiable. This means that it could be changed periodically or as needed to maintain the system's integrity and protect against unauthorized access. The admin mode serves several critical functions, the most vital of which is the ability to correct responses to questions that the system could not recognize. This ability to manually intervene and modify responses is a crucial aspect of the system's self-improvement mechanism. Once these modifications are made, they are stored in the system's database. When users interact with the platform the next time, these corrected responses are directly delivered, enhancing the user experience and the system's accuracy. In addition to correcting unrecognized questions, the admin mode also offers the flexibility to modify responses to predefined datasets in the database. This means that the admin could manually adjust the responses to a set of predefined questions based on various scenarios and different conditions. Moreover, the admin mode also allows for the addition of information about new programs. This information is added in

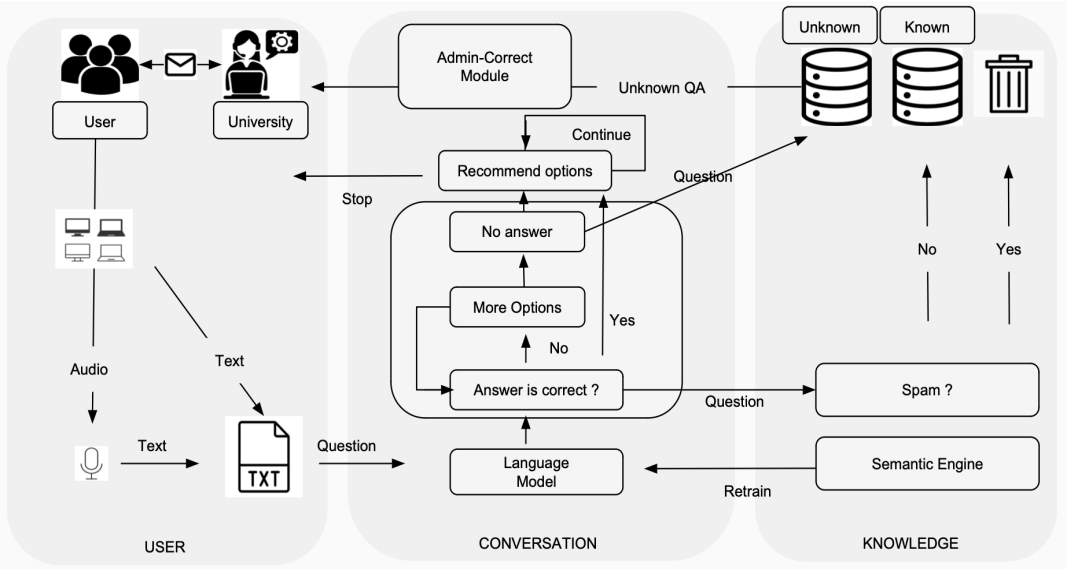


Figure 5 – Diagram of system flow

the format of Json, a popular data interchange format, ensuring that the system stays updated with the latest programs and offerings. The admin mode also plays a significant role in system monitoring. It provides simple visualizations for tracking the model's status outcome and monitoring visitor status. These visualizations offer insights into the system's performance, helping to identify and correct any abnormalities. They are also valuable data that could be used to update admission strategies, ensuring that the system continues to meet user needs effectively.

A point that needs to be paid attention is that there exist two specific scenarios wherein a model would undergo retraining in order to update and enhance its knowledge base. It's important to understand that the model retraining process is situated within the knowledge module, a location that is subject to high computational intensity and potentially time-consuming based on the quantity and quality of the data being processed. The first scenario happens when there is a need for inserting a new label for a correspondent question. In this situation, the model's label could face dramatic shifts when a new label is introduced into the system. The introduction of new labels could cause substantial changes in the latent space, which in turn, retraining is required to complete knowledge insertion for the model. The retraining process is integral to update the knowledge encapsulated in the parameters of the RoBERT model. The data used for this retraining consists of augmented datasets derived from the GPT augmentation process, as well as the datasets currently stored in the database. The amalgamation of these datasets ensures that the model is updated with the most comprehensive and up-to-date information, thus allowing for a more accurate and robust model. The second scenario where retraining becomes necessary is for the sustainable and automatic development of the system. In order to maintain the accuracy and relevance of the system, a retraining process is set to occur after every specific period. This regular update process is designed to incorporate the knowledge derived from the user's texts into the model. This constant updating of knowledge ensures that the model remains informative and is able to provide accurate responses based on the latest user interactions. However, during the retraining process, the model continues to remain as the previous pre-trained model until the new training process is fully complete. This is a

critical step to prevent any knowledge inconsistency between the model and databases. This step ensures that the model maintains consistency with the database and prevents any potential conflicts or errors from arising due to discrepancies among them. Moreover, it also ensures that the system is able to support uninterrupted service to the users. Even though the model is undergoing retraining, the services provided to the users are not disrupted. This continuity in service provision is crucial to maintaining user satisfaction and ensuring a seamless user experience. The diagram of semantic engine flow is shown in fig.6.

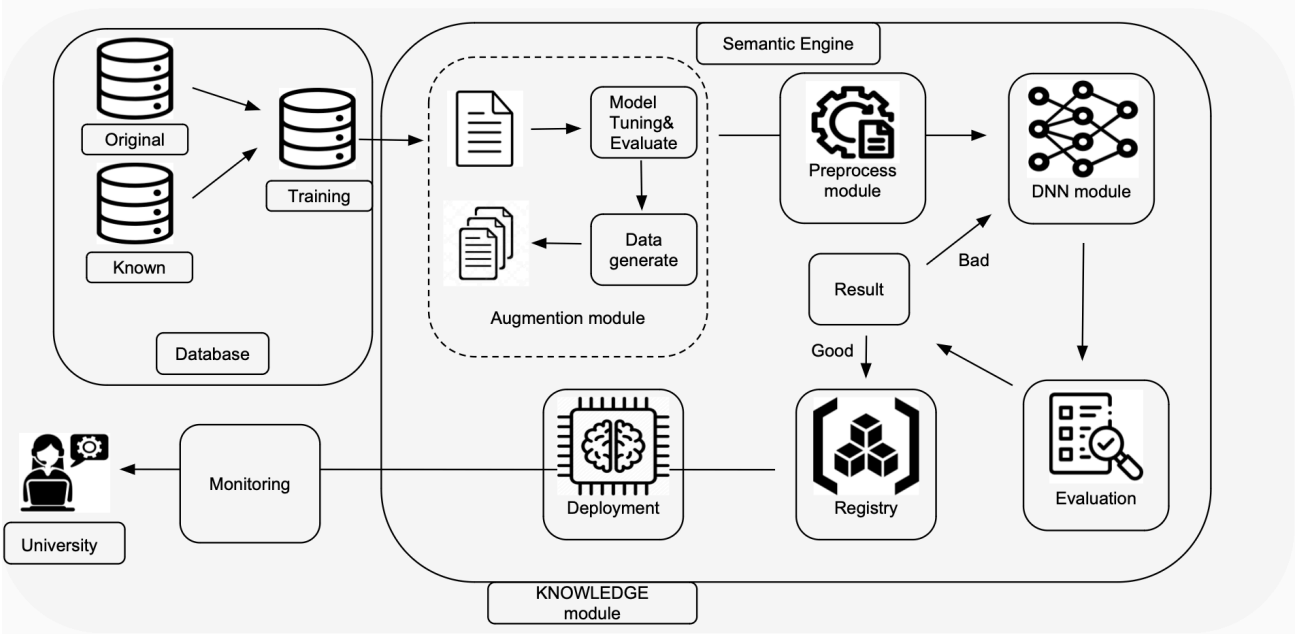


Figure 6 – Flow visualization of semantic engine

2.6 Recommendation system and self-correction system

Recommendation system is built upon fine-tuning a pre-trained RoBERT model from previous inputs of users to relate potential questions which could be intriguing for future users to minimize the necessity of manual input from users to prevent inaccurate prediction from the model due to currently insufficient datasets. Additionally, as alternatives of RoBERT, the SASRec model is preferable compared to another well-known recommendation model BERTrec[51] due to its intuitive way of training unidirectionally. However, at the moment without enough datasets to train models, current recommendation systems utilize user input to query potentially interesting topics for them by predicting with a fine-tuned RoBERT model. It is expected to help users to find out more detail by extending the meaning of current input, which enables the model to maximize knowledge of current databases in the meanwhile reaching to the knowledge that users might not expect. On the other hand, it decreases the chance of making mistakes by the NLP model, therefore preserving the reliability and practicality.

Self-correction relies on robust NLP models to enable predictions to be maintained in top-k predictions, which could succeed in providing few options for users to be corrected without presenting many options to increase the complexity of browsing for users and decrease the comfort of the usage.

2.7 Database design

Databases play a central role in current system and are currently divided into two key clusters based on their function. The first type of databases is designed to store original datasets and provide predefined standard datasets. These databases serve as a reservoir of original data, ensuring that there is a constant supply of high-quality, standard datasets for various operations. The second type of databases is employed for dynamic modifications. These databases are designed to be highly adaptable, accommodating changes such as the storage of new input datasets from users, or the recording of index data after model training. This flexibility makes these databases ideal for tasks that require regular updates and modifications.

In the latter cluster, three main tables are implemented: the “known” table, the “unknown” table, and the “train_data” table. Each of these tables serves a unique purpose and contributes to the system in a different way. The “train_data” tables consist of original datasets sourced from the former cluster. These datasets act as standard responses provided to users when they pose questions. This standardization ensures that users receive consistent and reliable responses, enhancing the overall user experience.

On the other hand, the “known” and “unknown” tables store inputs from users after these inputs have been evaluated by the feedback of the current user in the self-correction mechanism. This process allows the system to learn from user interactions and improve over time. It is important to note that only the “train_data” and “known” tables are used to retrain models. This strategy of data for retraining ensures that the models are always learning from the most reliable and relevant data, thereby improving their accuracy and efficiency.

Each new input record in our database incorporates basic user information, such as name and ID. These pieces of information serve as distinguishers, allowing us to differentiate between various users and their respective interactions with the system. In addition to this, the record also stores key data points such as the user’s input and the system's predicted response. This information is crucial as it allows us to track and analyze the system's performance over time, providing insights into its accuracy and effectiveness. Another vital piece of information that each record holds is the time of interaction.

Time serves as a critical factor, primarily for statistical purposes. It helps in tracking user activity over specific periods and identifying patterns or trends in usage. Moreover, time could play a pivotal role in constructing recommendation systems. By recording the order of questions asked by each user, it could be easier to build a chronological sequence from their interactions. This sequence could then be analyzed to identify patterns or preferences based on the user's past behavior. For instance, if a user frequently asks about a particular topic after another specific topic, the system could leverage this information to anticipate future questions and provide more targeted responses. This level of personalized interaction could significantly enhance the user experience, making the system more responsive and user-friendly.

2.8 Cloud service and database

Docker, a popular platform used for containerization, encapsulates an application along with all of its associated dependencies into a standalone unit known as a 'container'. This ensures the application functions consistently across different computing environments. The primary advantage of using Docker lies in its scalability and portability. By containerizing applications, developers could ensure that they would function identically across a number of different instances, regardless of the host environment. This makes Docker an ideal choice for developing, testing, and deploying

applications across varied platforms and systems, ensuring both reliability and efficiency.

AWS ECS (Amazon Web Services Elastic Container Service) is a cloud-based service designed to manage and deploy containers at scale. It is highly efficient for large-scale container deployments, offering features like Auto Scaling and Application Load Balancers. Auto Scaling is a feature that automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost. It adjusts the computational resources based on the demand, thus ensuring optimal resource utilization. Application Load Balancer, on the other hand, distributes incoming application traffic across multiple targets, such as Docker containers, and routes traffic to targets within Amazon Virtual Private Cloud (VPC) based on the request content. These features make AWS ECS a robust solution for managing and scaling containerized applications.

MongoDB was chosen as the platform for storing datasets. It's a NoSQL database, which differs significantly from traditional SQL databases. The primary advantage of MongoDB, and NoSQL databases in general, is their flexible schema. This means that unlike SQL databases, which require the structure of the data to be defined before anything could be stored, MongoDB allows for the storage of data with a dynamic, flexible schema. This flexibility allows for faster and more agile development, as well as the ability to store complex data structures. Additionally, MongoDB supports a rich query language, allowing for complex queries and data manipulation. The combination of these capabilities makes MongoDB a powerful tool for building intuitive and complex applications.

3 EXPERIMENT

Given the limited preliminary training datasets, with only 82 total data points spread across as many as 23 different topics, it's reasonable to question whether the training model would be capable of generalizing all related unseen contexts during the initial stage of deployment. This concern arises from the fact that having a small and diverse dataset could make it challenging for the machine learning model to learn the nuances of each topic and may lead to overfitting, where the model fails to capture the underlying structure of the data. To overcome this potential issue, customized NLP methods are proposed in this work based on the unique situation: data augmentation and classifier optimization. These techniques are known for their effectiveness in enhancing the performance of models, particularly in discriminative tasks where the goal is to classify input data into one of several predefined categories.

Data augmentation is a technique that increases the size and variety of the training data without the need for new data collection. This is achieved by creating new data instances through various transformations, such as synonym replacement, random insertion, random swap, or random deletion. By augmenting the training data, the model is exposed to more diverse examples, which could help it learn more robust representations and improve its generalization ability.

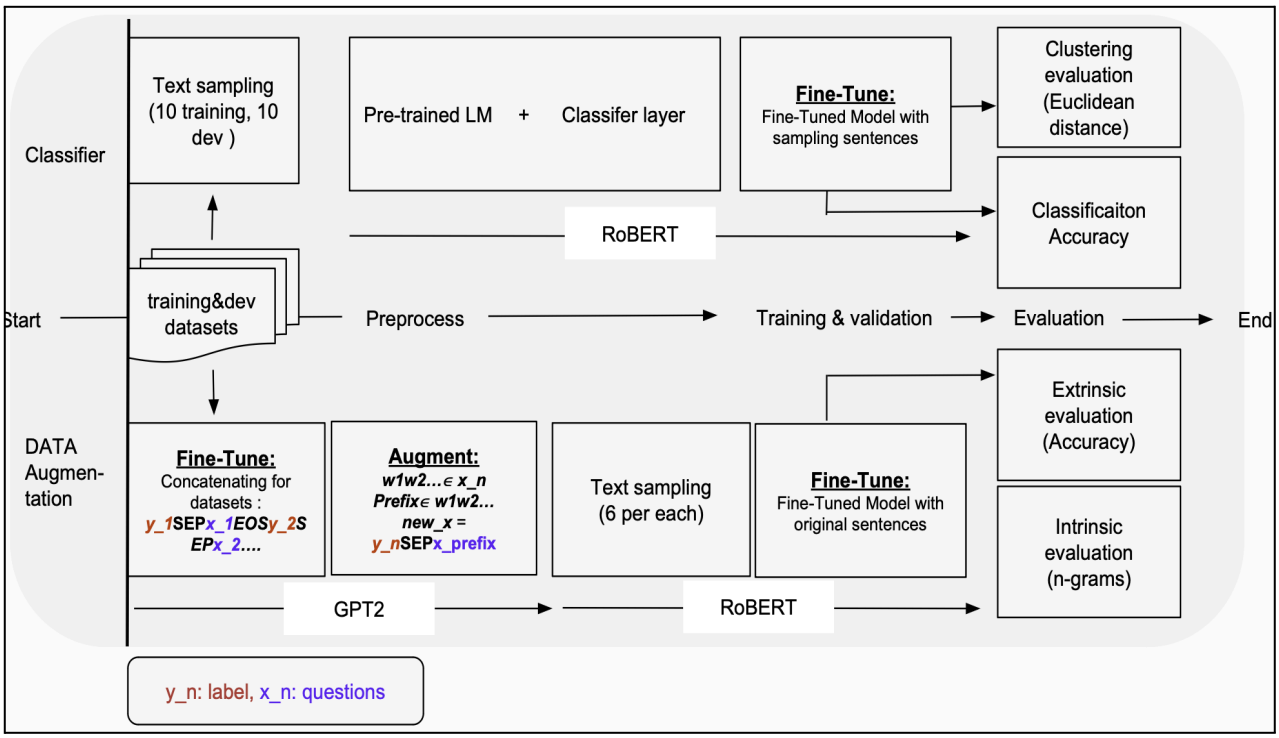


Figure 7 – The illustration of experiment workflow for proposed classifier approach and proposed data augmentation approach

Classifier optimization, on the other hand, is a complex but necessary process in machine learning, which involves fine-tuning the parameters of the classification model to improve its performance. This could include selecting the best features for classification, tuning the hyperparameters, or choosing the most suitable classification algorithm. With proper hyperparameter setting, optimizing the classifier could lead to more accurate and efficient models. The fig.7 attached below illustrates the flow of experiments for both of these proposed methods,

providing a better understanding of the following section. This visualization helps to better understand the sequence and interdependencies of the steps involved in the optimization process. Also, The pseudocode of algorithm and the summary of experiment setting are listed with table format in APPLICATIONS C, APPLICATIONS D respectively for further details.

3.1 Data augmentation

3.1.1 Augmentation method

In this work, approach is an inspiration and extension from previous work implementing GPT2, which belongs to AR model as a generator model following the conditional method from[24] and originally appends the prepending method in order to solve the struggles in preserving label information. By providing an extra contextual sentence, GPT2 could follow its nature to make predictions of the next work token. With inspiration from these magnificent works, the idea of proposal approach is to simply use pos-tagging as a stop word for prefix before generating new sentences to create versatile sentences based on lexical class. The stop word is set to be noun as primary priority and second priority is verb, if there are multiple choices for stop words, the ultimate prefix would be decided stochastically. The ideal result is supposed to have different length of text by combining token level and sentence level augmentation with powerful GPT2's enriched lexical knowledge; conceptual images are shown in Fig 4.

Second proposal is to use a combination of multiple methods, the purpose is to enrich the context with token-level augmentation before fine-tuning the GPT2 model in order to relate words with lexical similarity by clustering similar vocabularies together in the fine-tuning stage. By combining EDA methods and GPT2, it is expected to generate versatile sentences from original sentences. The ultimate hypothesis is to enrich latent space of language models with augmented datasets from EDA in order to generate similar meaning vocabularies and context, thus versatile training datasets are expected to be gained before model training stage.

The main factor of choosing GPT2 as a pipeline model instead of newer versions like GPT3 and GPT3.5 is unavailability due to payment service under consideration of common practitioners and researchers with limited financial resources for research and another reason is the massive size of the upgraded model to local device. But the proposed augmentation approach is generalized, thus replacement of pre-trained models is achievable.

There are some other alternatives that could be utilized, such as GPT-Neo and GPT-J. They are two variants of GPT3, and have potential capacity as simpler versions of GPT3. Therefore in experiment step GPT2 is implemented in order to have fair comparison with BERT model due to similar parameter size and when training the intellectual support system, GPT-Neo is utilized to enrich contextuality in augmentation datasets in order to maximize the knowledge of NLP model after data augmentation from EDA methods.

3.1.2 Filtering method

While data augmentation is a promising technique for enhancing the robustness of machine learning models, especially when dealing with limited datasets, it certainly possesses its own potential drawbacks. One of the key challenges associated with data augmentation, particularly in the domain of NLP, is maintaining the fidelity of the augmented data. Fidelity, in this context, refers to the label-conserving ability of the augmented data to the original meaning of the text. When creating new data instances through techniques such as synonym replacement or sentence rephrasing, there's a risk of distorting the original meaning, which could lead to misleading training examples. This distortion could result in the model learning incorrectly and adversely affect its performance. In the worst-case scenario, this could bring 'catastrophic forgetting', where the model

discards previously learned useful information in favor of these distorted examples, ending up with significant drop in performance. Therefore, it's crucial to have a post-augmentation filtering step, where the quality and relevance of the augmented data are evaluated. This could involve manual inspection or automated approaches such as using language models to evaluate the semantic similarity between the original and augmented text. The goal is to ensure that only high-quality, relevant augmented data is added to the training set. In previous work[31], it followed one extremely simple way to evaluate augmented datasets by using a pre-trained discriminative NLP model. To evaluate the fidelity of the augmented text, it is necessary to fine-tune BERT with original datasets. Fine-tuning a discriminative NLP model on the original datasets allows the model to learn the unique characteristics and nuances of each original sentence, thereby enabling it to form distinct clusters in the embedding latent space. The embedding latent space is a high-dimensional space where similar instances are clustered together. When the model is fine-tuned, it learns to map each instance of the augmented text to its corresponding position in this latent space. This mapping is achieved through the application of embedding layers, which transform the text into numerical vectors that could be processed by the model. The key to assessing the fidelity of the augmented text lies in comparing its position in the latent space with the centers of the clusters associated with each label. Intuitively, if the distance between the augmented text and an assigned cluster center is larger compared to other cluster centers, this could indicate a significant semantic or syntactic difference between the original and augmented text. In other words, if the augmented text is located far from the center of its original cluster, it could be considered as not faithful to the original text. This is because the comparatively larger distance suggests that the augmented text may have deviated from the original meaning or context based on its label. Therefore, this measure serves as a useful criterion for filtering out augmented texts that lack fidelity, thereby ensuring that only high-quality, relevant data is used for model training. In fig.8, it displays the whole workflow of proposed data augmentations from augmentation method to filtering method for better understanding.

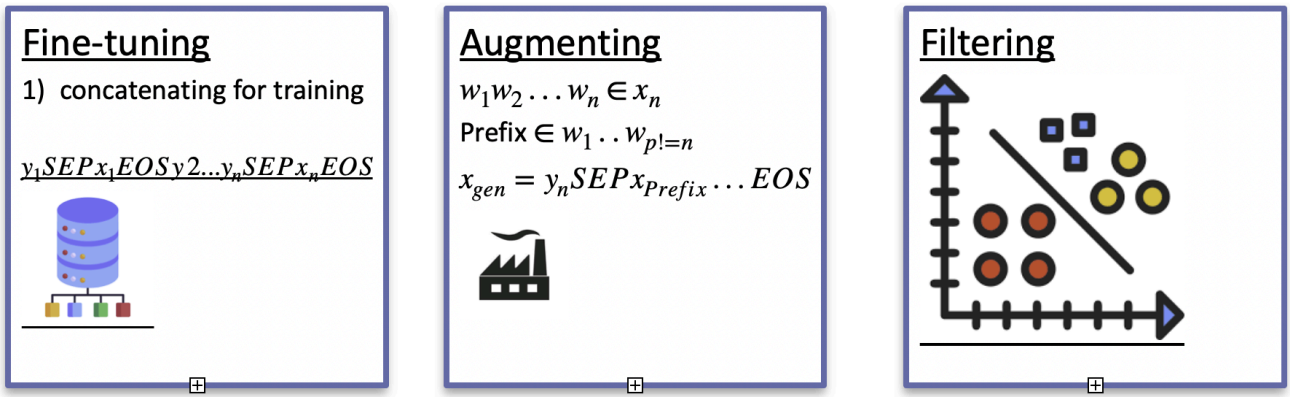


Figure 8 – Conceptual graph of proposed data augmented methods by using GPT2, In fine-tuning stage, all pairs of datasets are concatenated to be used; in augmenting stage, prefix is structured based on where last word is noun or verb

3.1.3 Evaluations

In the field of machine learning, it's vital to evaluate the efficiency and resilience of trained models. To achieve this, an evaluation strategy using a low-resource scenario is suggested. This scenario simulates real-world conditions where substantial labeled data for model training isn't

always accessible. In these situations, models are expected to learn from a limited set of labeled data. Such circumstances present a significant challenge as the lack of labeled data could hinder the model's ability to learn varied patterns present in the larger population. Consequently, the model might experience problems like overfitting, where it performs well on the training data but poorly on unseen data, or underfitting, where it fails to capture the underlying patterns in the data entirely. Implementing the low-resource scenario allows us to evaluate the model's ability to generalize from limited data. It offers valuable insights into the model's learning capacity, performance of accuracy in making predictions when data is scarce, and its resistance to potential overfitting or underfitting. Scenarios like data scarcity are prevalent in real-world applications and pose a unique challenge as they require the model to learn effectively from a smaller data volume.

In setup, augmented datasets are used with original datasets for training. Data augmentation is a method that allows to boost the variety of data available for model training without the need to collect new datasets. This could encompass strategies such as text augmentation, where text instances during the training phase are slightly adjusted to produce new training samples. For example, this could involve rearranging words, substituting words with their synonyms, or even modifying the sentence tense. By training on these enriched datasets, the model is anticipated to comprehend the subtleties of different text instances. This learning process should allow the model to become more attuned to similar instances in the real world, thereby enhancing its capability to generalize from the training data to unfamiliar data.

There are two primary evaluation methods: the extrinsic method and the intrinsic method. To begin with extrinsic evaluation, it necessitates the fine-tuning of the RoBERT model using both the original datasets and their corresponding enhanced versions. The model, through the augmented datasets paired with the original ones, is exposed to a broader range of data instances, thereby boosting its generalization capabilities and capacity to handle various inputs. After the fine-tuning phase, the model undergoes validation through separate validation datasets to identify potential issues such as overfitting. Following this, the model's performance is further assessed by testing its accuracy with test datasets. This stage is crucial as it sheds light on the model's extrinsic performance - its competency in real-world situations. The primary objective of this method is to determine whether text augmentation positively influences the model's capacity to recognize unseen datasets. The goal of increasing the diversity and volume of the training data through augmentation is to enhance the model's resilience and adaptability. Consequently, this should improve its performance on new, unseen data, rendering it more effective and dependable for practical implementation.

On the other hand, intrinsic evaluation begins by using original datasets as training datasets to fine-tune the model. Once the model has been fine-tuned, evaluation of the augmented text’s fidelity is proceeded with the labels of the original data. Fidelity, in this case, refers to how well the augmented phrases maintain the same meaning as the original phrases from which they are derived. This step is significant as it allows us to assess whether the original meaning is preserved after data augmentation. Preservation of original meaning is key in ensuring the effectiveness of the augmentation process, as it ensures the model is learning from accurate and relevant data. In addition to fidelity testing, it's also valuable to verify the versatility of the model by using n-grams. N-grams are contiguous sequences of n items from a given sample of text or speech. In this case, 1-gram and 3-grams are planned to be applied for evaluation. This approach helps verify if semantically different phrases are recreated from the original phrase, which is important for ensuring the model's ability to understand and process language in a nuanced manner. Utilizing 1-gram would involve looking at individual words, which could be informative in identifying the most common words or determining the vocabulary of the text. On the other hand, 3-grams would involve

examining sequences of three words; it could provide insights into the contextual relationship between words and how they are commonly used together.

3.1.4 Baseline methods

In the experiments, several popular methods from previous research are applied to compare with proposed methods. Methods are referred from previous work[31] with inspiration of :

- EDA - short for Easy Data Augmentation, is a straightforward yet effective method that focuses on single-word replacement as a technique to enhance text classification performance, particularly in situations where data is sparse.
  - Randomly insertion
  - Randomly deletion
  - Randomly swapping
- Back Translation is a method using a machine translation model by converting to another language and later translating it back to the original language. In this method, deep back-translation is used to be way more powerful than a single language machine by implementing a machine translation model with multiple languages, more than two conversions of knowledge exchange in order to achieve creative production with adversarial examples due to difference of lexical meaning between languages.
- CBERT - is a language model-based approach and, considered to be the first model-based extension that outperforms other single-word replacement-based approaches.
- CGPT2 - is a variant of using CBERT by using GPT2 as pipeline model

One thing should be noted that CBERT and its variants follow the instruction in[24] by prepending the prefix when generating new text.

3.1.5 Generator and filter model

When it comes to implementing a generator for data augmentation, the ideal outcome is to create diverse and varied phrases that faithfully retain the original meaning of the text. In the field of NLP, numerous models are capable of generating text, but the GPT family of models stands out for their exceptional creativity and versatility. In this work, GPT2 is selected to be the target model to be applied in the data augmentation process. The rationale behind this selection is that GPT2, with its ability to generate diverse and contextually appropriate text, is well-suited to creating augmented data that is both varied and faithful to the original text's meaning.

To implement filter classifiers for evaluating the fidelity of the augmented text, BERT-like model is preferred, due to its potent discriminative abilities. Specifically, RobertaBert with the "roberta-large" model was chosen to assess the generated sentences. RobertaBert is essentially a variant of the original BERT model, sharing the same architecture but differing in a few crucial areas that enhance its performance. One distinguishing feature of RobertaBert is its use of byte-level BPE (Byte Pair Encoding) as its tokenizer. This allows it to handle a broader range of characters and words, enhancing its capability to understand and represent text data. Furthermore, RobertaBert employs a more complex pre-training scheme, utilizing larger datasets and bigger batches during the training process. As a result of these enhancements, RobertaBert has been empirically shown to outperform the original BERT model in various tasks. In terms of hyperparameters for this particular experiment, a drop rate of 0.1 is applied to the sentence representation before the Softmax layer carries out further processing. The Softmax layer is responsible for transforming the output into a probability distribution over predefined classes, providing a basis for making the final prediction. The introduction of a drop rate helps prevent overfitting, thereby ensuring the model generalizes well to unseen data. Adam optimization, a popular choice for training deep learning

models, is used with an initial learning rate of 4e−5 to tune the cross-entropy loss. Adam optimization has capability to adjust the learning rate adaptively for each parameter, leading to more efficient and effective model training. The cross-entropy loss is a commonly used loss function for classification tasks, measuring the dissimilarity between the model's predictions and the true labels. By fine-tuning this loss, the model's performance on the classification task could be significantly improved.

3.2 Hypercomplex classifier

Hypercomplex neural networks differ from traditional neural networks in their higher complex dimensionality. Most current hypercomplex neural network implementations are adaptations of traditional algorithms to high-dimensional situations, and they have achieved positive results, which gives motivations for improvement to future researchers. The design of model structure is straightforward, just replace fully-connected layers in the traditional classifier with PHM layer and it is supposed to have the same functionality to apply a linear transformation to the input vector through a weights matrix, however with hypercomplex domains.

On pairwise text classification tasks, universal benchmark datasets are used to test performance of proposed classifiers with pre-trained language models in order to experiment the compatibility and generalization. In the realm of pairwise text classification tasks, universal benchmark datasets serve as the gold standard for evaluating the performance of proposed classifiers. These datasets, which encompass a diverse range of text instances across various domains and classes, provide an objective measure of a model's ability to accurately classify new, unseen data. Utilizing these datasets allows us to rigorously test and validate the classifier's generalization capabilities with pre-trained language models.

Proposed approach draws inspiration from the recent success in the hypercomplex domain, particularly with regards to the state-of-the-art neural network model architectures. Advantages and limitations of these architectures are taken into consideration as discussed in[46] while developing their own methodology. The primary objective of current research is to scrutinize whether the proposed classifier possesses the capability to decode the latent space of real-value domain representations. It is intriguing to manage modification of representations derived from successful pre-trained language models, such as the BERT-like models and the GPT family of models. These pre-existing models have shown remarkable proficiency in various natural language processing tasks and, as such, offer a solid benchmark for the proposed classifier. It is required to delve deep into the intricate workings of these models, focusing on how they represent and manipulate data in the real-value domain space. By doing so, it is expected to have deeper insights that could help us optimize proposed classifiers to better interpret these representations. Interestingly, this research could be the pioneer in conducting an experiment that bridges the gap between hypercomplex classifiers and pre-trained language models. It's an uncharted territory that holds considerable potential for the advancement of machine learning and NLP. By comparing and contrasting the performance of proposed hypercomplex classifiers with these pre-trained models, there could be an expectable leap in the field and contribute towards the development of more robust, efficient, and effective machine learning models.

3.2.1 Model selection

As the foundation for current research, two well-established models serve as baseline classifiers, namely RobertaBert and GPT2. For RobertaBert, "roberta-large" variant was selected, and for GPT2, "gpt2-Medium" was implemented. Both these models are implemented using the “Huggingface” library in conjunction with Pytorch, a popular open-source machine learning library.

The rationale behind the choice of these specific sizes for both models is to ensure a fair and balanced comparison. The goal is to minimize any discrepancies that could arise from differences in the number of parameters between the models, thus opting for versions that had a similar parameter size. Roberta-large, for instance, has approximately 355 million parameters, while gpt2-Medium has a similar count of around 354 million parameters. By maintaining a similar parameter size for both models, it could be ensured that any differences observed in their performance are not merely a result of one model having a larger capacity to learn due to a greater number of parameters. Instead, any observed differences would more likely be attributable to the inherent characteristics and capabilities of the models themselves.

Utilizing these models as baseline, it could offer a solid foundation for research and ensure that findings are robust and reliable. The goal is to shed light on the strengths and weaknesses of these classifiers in decoding the latent space of real-value domain representations and, in the process, contribute valuable insights to the field of NLP.

3.2.2 Functionality and difference

Although both Bert and GPT models utilize the transformer architecture, they exhibit significant differences in their structural components and functionalities. Bert is an encoder-only model, meaning it takes a sequence of input tokens and encodes them into a series of contextualized representations. On the other hand, the GPT family consists of decoder-only models, which implies they generate a sequence of output tokens based on the given inputs. One key difference lies in their handling of sequences. Bert exhibits non-autoregressive properties, which means it processes all tokens in the input sequence simultaneously. This allows Bert to predict masked tokens based on the context of the entire sequence, not just the preceding tokens. This feature allows Bert particularly effective in tasks like text classification or named entity recognition, where understanding the context of the entire sentence is crucial. In contrast, GPT models are autoregressive, implying they make predictions sequentially from left to right, with each prediction dependent on the preceding tokens. This property makes GPT models especially good at generating coherent and contextually appropriate text, as each word generated is influenced by the words that came before it. As a result, GPT-structured models are widely recognized as powerful generative models, often used for tasks like text generation, language translation, and summarization.

3.2.3 Structure and hyper-parameters

RobertaBert is constructed with a total of 12 layers, each possessing 768 hidden layers and 12 attention heads. These attention heads facilitate the model in concentrating on various parts of the input sequence, thereby enhancing its understanding of the context of each word. On the other hand, the GPT2 framework comprises 24 decoder-only transformer blocks. Each of these blocks contains 16 attention heads and 1024-dimensional embedding and bottleneck layers. These features enable GPT2 to produce coherent and contextually suitable text based on the provided inputs. In the experiment setup, RobertaBert leverages the pooled hidden state representation with the initial special token ([CLS]). Padding is configured to focus on the left side in GPT2. This ensures that the sequence is accurately interpreted by the model, regardless of the input's length. Furthermore, a dropout rate of 0.1 is applied to the sentence representation before being forwarded to the Softmax layer. This strategy prevents overfitting by arbitrarily disregarding a portion of the neurons during training, thereby enhancing the model's generalization capability. Adam optimization was employed, as referenced in[52], with an initial learning rate of 4e−5 for adjusting both the cross-entropy loss and the contrastive loss. Adam, an adaptive learning rate optimization algorithm, is widely known for its efficiency and minimal memory requirement. The total epochs are also set to 40 runs. An

early stop rule was implemented, which could trigger the early stop of training if the validation loss is 15 times higher than the current best validation loss. This rule is particularly important considering the comparatively longer time required for a model to converge in hypercomplex neural network training processes. With such a design, it allows the training process to maintain efficiency while ensuring the quality of training model based on validation loss.

3.2.4 Contrastive learning

While supervised learning is a prominent and frequently used machine learning technique, it does have its difficulty and challenges. It largely requires the presence of accurately labeled data, however collecting such data could often be laborious and time-consuming. Additionally, it may encounter problems such as generalization errors, where the model performs poorly on unseen data due to overfitting or underfitting during training. The model may also struggle with spurious correlations, mistaking coincidence for causality and leading to incorrect predictions. Furthermore, supervised learning models are vulnerable to adversarial attacks, where deceptive inputs designed to trick the model could result in incorrect outputs.

On the other hand, contrastive learning has been gaining a prominent place for its ability to circumvent some of these issues. This unsupervised learning method trains models to comprehend robust and significant representations of data by comparing and contrasting different data points. The main advantage of contrastive learning is that it doesn't require extensive label-annotation, which could avoid a significant time and cost, particularly in the situation with large datasets. In real-life situations, manually annotating large datasets could be tedious and unfeasible. Contrastive learning, on the other hand, could work with unlabeled data, making it a more scalable and efficient approach for many applications.

It focuses on drawing the text of the same labels closer in the embedding space while pushing away the embeddings from different samples[53][54]. Contrastive learning enables efficient utilization of unlabeled content; by capitalizing on the relationships between samples, the model could extract valuable information from a vast quantity of unlabeled text, thereby boosting its ability to generalize. Temperature is one vital hyperparameter as an input-dependent variable, a measure of embedding confidence which could control how sensitive the objective is to specific embedding locations, thus influencing the relative relationship between relational text and non-relational text. In this experiment the temperature is set to 0.3. In the implementation, baseline classifiers with and without contrastive learning are tested in order to experiment whether the performance of hypercomplex neural networks could improve with contrastive learning.

3.2.5 Evaluations

Accuracy and the F1 score are two essential metrics used to assess a model's performance. Accuracy quantifies the ratio of correct predictions made by the model, while the F1 score shows the balanced evaluation of a model's precision and recall. These scores offer a comprehensive understanding of the model's ability to make correct predictions and its effectiveness in identifying relevant instances.

Euclidean distance and Kullback-Leibler (KL) divergence are two well-known techniques when it comes to clustering tasks, in the implementation phase both are implemented with the Sklearn library in Python. Euclidean distance denotes the straight-line distance between two points in a space and is common in unsupervised learning techniques, particularly in K-means clustering. The algorithm uses Euclidean distance to form clusters of related data points that share similar characteristics. This feather of grouping similar data points together makes it possible to create distinct, separate clusters. On the other hand, KL divergence is a measure of how one probability distribution diverges or deviates from a second expected probability distribution. This method is

especially pertinent to unsupervised learning problems and applications that require the comparison or approximation of probability distributions. By comprehending the divergence between different distributions within the data, we can gain a deeper understanding of the data's clustering.

3.2.6 Low resource scenario

This experiment aims to replicate a scenario where source datasets are scarce by using a restricted number of training sets on each separate baseline dataset. This method is designed to mimic real-world situations where data may be limited, and hence, the model has to learn from a confined set of examples. In the present assessment, the experiment was conducted using 10 specific datasets for each label during the training phase. This experiment design was tactically selected because, with a larger dataset, the models could easily attain high accuracy, thereby failing to offer a fair and meaningful comparison. If all models are to reach nearly flawless accuracy, it would become difficult to distinguish their individual merits and demerits, and the evaluation would not yield valuable insights.

4 RESULT

4.1 Data augmentation

From the result of extrinsic evaluation and intrinsic evaluation shown, an analysis of both extrinsic and intrinsic evaluations reveals interesting insights into the performance of different language model approaches. The results in table 5 indicate that the back translation approach and all GPT2 approaches have a higher capacity to incorporate new words, extending beyond the existing vocabulary set. This conclusion is drawn from the 1-gram analysis, which essentially measures the presence of individual words in a text. However, when it comes to the 3-gram results, the GPT2 series approaches demonstrate superior performance in sentence reconstruction. The 3-gram analysis, which considers sequences of three words, provides a deeper understanding of the model's ability to maintain the syntactic structure and coherence of sentences. Interestingly, the 3-gram result appears to have a strong correlation with fidelity, shown in table 6, which refers to the model's ability to maintain the original meaning in the generated text, while grammatical correctness pertains to the model's adherence to the rules of grammar in the target language.

Table 5 – Variety comparison based on 1-gram and 3-grams, best variety is outlined in blue and worst one is in red, results of proposed methods are listed in gray rows

model	ITMO-23		TREC-6		SST2-2	
n-grams	1	3	1	3	1	3
EDA	0.1147	0.428	0.0491	0.3864	0.0305	0.3647
BackTra	0.0846	0.3248	**0.0598**	0.3422	**0.05**	0.446
CBERT	0.0529	0.1806	0.0303	0.1772	0.0181	0.1661
CGPT2	**0.1325**	**0.6979**	0.0453	0.5293	0.0353	**0.6121**
CGPT-Pos	0.1044	0.4899	0.0479	0.4669	0.0381	0.5642
CGPT2 + EDA	0.1049	0.6326	0.0445	**0.5756**	0.0264	0.5685
CGPT-Pos + EDA	0.1067	0.576	0.0511	0.5615	0.0337	0.5293

Table 6 – Fidelity (original-meaning-conservative ability) comparison, best result is outlined in blue and worst one is in red, results of proposed methods are listed in gray rows

model	ITMO-23	TREC-6	SST2-2
EDA	0.9631	0.9081	0.9458
BackTra	0.9689	0.9376	0.9302
CBERT	**0.9825**	**0.9592**	**0.9716**
CGPT2	**0.3197**	0.5488	**0.6844**
CGPT-Pos	0.7519	0.6055	0.7586
CGPT2 + EDA	0.3372	**0.5275**	0.6896
CGPT-Pos + EDA	0.7112	0.5824	0.8177

The data suggests that as 3-gram scores increase, fidelity scores decrease. This implies that while the GPT2 series models excel at reconstructing sentences and maintaining structural integrity,

there may be some loss of semantic accuracy. This inverse relationship between 3-gram scores and fidelity indicates that while sentence structure is maintained, the precise meaning of the text may not be perfectly preserved. This observation opens up new avenues for further research into improving the balance between structural coherence and semantic fidelity in language models. The most plausible elucidation for the observed phenomena lies in the quality of the original datasets. If the original datasets offer insufficient or poor-quality information, it becomes challenging for the model to discern patterns from the generated datasets. This issue becomes particularly prominent in low-resource scenarios, where the data available for training and evaluation is limited. Such a situation often creates a significant impediment in the evaluation process. It restricts the ability to effectively validate two critical aspects of the model's output - fidelity and grammatical correctness, measuring the faithfulness of the model's output in maintaining the original meaning of the text. There might be concerns about the model's capability to acquire and apply knowledge in such circumstances. This is despite the fact that the pre-trained GPT2 model, which has been utilized in this case, already possesses a certain extent of lexical knowledge. The model is designed with an extensive range of lexical understanding, yet its performance would be compromised if the foundational datasets used for fine-tuning and training do not offer comprehensive, high-quality information. Therefore, it becomes essential to ensure the quality and comprehensiveness of the datasets used in the fine-tuning and training process. This not only helps to improve the model's performance but also enhances the reliability of the evaluation process. By addressing these concerns, more accurate insights into the model's capabilities, more informed decisions about its applications and improvements are expected to be achieved.

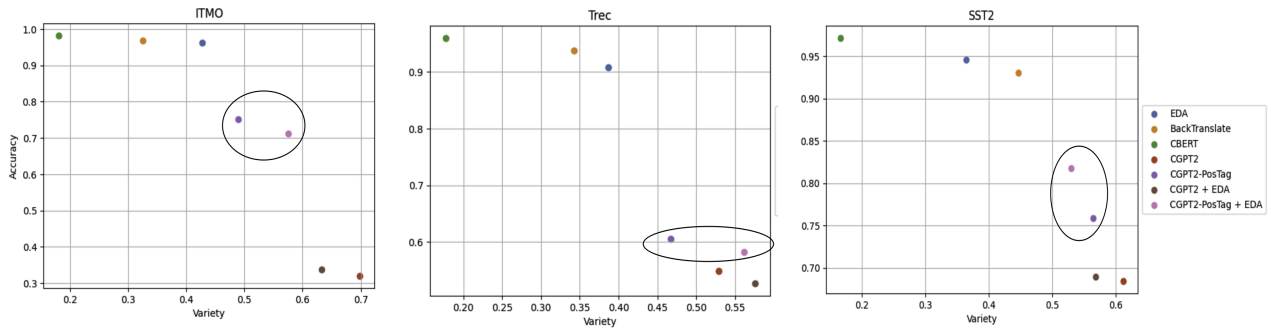


Figure 9 – Variety(x-axis) calculated by 3-grams and fidelity ,denoted as accuracy(y-axis); six sentences were generated per original sentence, proposed methods are circled

The suggested approaches, namely the Pos-tagging GPT2 and EDA-Pos-tagging GPT2, appear to strike a balance between variety and fidelity shown in fig.9. In the evaluation metrics, neither of these methods outperformed the other methods significantly. This lack of a clear superiority indicates that both models only have a comparable level of effectiveness and could be feasibly integrated into the workflow of the current developing system. The trade-off in performance is particularly valuable as it provides flexibility in choosing the most suitable model based on specific use-case requirements. In the system development stage, the EDA-Pos-tagging GPT2 model was applied for further implementation. This decision is likely based on the diversal generated contexts compared with Pos-tagging GPT2, even though it did not show outperforming outcome in any metrics, it showed a more clear explanation with the generated example in APPLICATIONS E.

4.2 Hypercomplex classifier

Table 7 presents a comprehensive comparison of various model configurations and their performance. The table outlines three distinct baseline datasets, two different model structures, two types of loss learning, and models with 16 and 256 hidden layers. This diversity in configurations provides a broad perspective on the performance of various models under different conditions. One of the key findings from this table is the comparative performance of hypercomplex classifiers and traditional classifiers. When applied to the SST2 datasets, hypercomplex classifiers demonstrate a slightly superior accuracy. The improvement might seem marginal at just 1% higher, but in the field of machine learning, even such small increments in accuracy could be significant. It indicates that hypercomplex classifiers are potentially more effective in interpreting and learning from the SST2 datasets. Similarly, in the case of the Trec datasets, hypercomplex classifiers also outperform traditional classifiers. This consistent superior performance across different datasets further validates the efficacy of hypercomplex classifiers. What makes this finding even more compelling is the efficiency with which hypercomplex classifiers operate. As shown in table 8, hypercomplex classifiers achieve these results with approximately 50% fewer parameter sizes compared to traditional classifiers. This efficiency is a significant advantage, as it means that hypercomplex classifiers could deliver better or equivalent performance while using fewer resources.

Table 7 – Accuracy comparison between 2 pre-trained language models with 2 different sizes of hidden layers in a classifier with different benchmark datasets for text classification. Each example is classified with name and hidden layers number, specified as -16 and -256. Accuracy is provided with standard deviation for better evaluation. Experiments were conducted with 5 epochs each and 10 training examples from each label

	SNIPS	TREC	SST2
ROBERTA-16	93.48 (1.61)	72.00 (8.26)	61.86 (11.65)
QROBERTA-16	92.6 (0.52)	**77.44 (10.33)**	62.99 (10.89)
ROBERTA-C-16	74.97 (31.47)	63.68 (22.11)	61.85 (12.47)
QROBERTA-C-16	82.57 (14.57)	41.64 (23.6)	55.75 (7.85)
ROBERTA-256	78.25(28.59)	71.5(3.47)	74.79(10.06)
QROBERTA-256	**93.58(2.51)**	76.27(25.69)	**80.61(9.12)**
ROBERTA-C-256	94.86(1.76)	72.78(20.4)	78.91(8.39)
QROBERTA-C-256	90.69(7.05)	63.57(11.3)	53.42(3.22)
GPT-16	67.69(16.17)	48(17.29)	50.98(2.11)
QGPT-16	59.82(19.71)	44.44(13.86)	51.46(2.58)
GPT-C-16	70.37(3.7)	22.88(7.35)	50.9(0.76)
QGPT-C-16	41.97(14.46)	20.8(6.59)	49.99(1.02)

The performance of the hypercomplex classifier on the SNIPS datasets could not present a better result than the traditional classifier. While it might not achieve the same level of accuracy as seen in the SST2 and Trec datasets, its performance is still comparable to that of a traditional classifier. This indicates that the hypercomplex classifier maintains its efficacy across different types

of datasets with various tasks, showing its substitutability. Moreover, an interesting finding emerges when introducing more complexity into the equation, the complexity of the neural networks increases, by adding more hidden layers, the hypercomplex classifier begins to show significant advantages over the traditional classifier. More specifically, it could achieve substantial parameter savings of more than 50%. This means that the hypercomplex classifier could handle more complex neural networks with fewer parameters, thus maximizing efficiency without compromising performance. This is a crucial finding as it suggests that the hypercomplex classifier is not only able to show comparable performance as traditional classifiers do but also could outperform it occasionally. This efficiency is particularly noticeable when dealing with more complex neural networks, where managing resources effectively could significantly impact overall performance. Moreover, the results also show less deviation in the performance of the hypercomplex classifier with more complex structure, as indicated by the standard deviation values. Lower standard deviation suggests that the results are closely clustered around the mean, indicating a higher degree of stability and robustness in the model's performance.

Table 8 – Parameter size comparison based on different hidden layer sizes. The number next to the model indicates hidden layer size and the number next to the dataset's name indicates the labels. It is obvious to confirm there is reduction of parameters based on different hidden layer size

	SNIPS - 7	TREC - 6	SST2 - 2
ROBERTA-16	33038	33004	32868
QROBERTA-16	**20693**	**12430**	**12394**
ROBERTA-256	528398	527884	525828
QROBERTA-256	**330773**	**198430**	**197914**

While the GPT2 model initially showed promising progress during its training phase, the final outcome was, unfortunately, less than satisfactory. This was primarily due to the inherent nature of the model as a generative model, which is not suited for text classification tasks. Despite this setback with the GPT2 model, the overall experiment still yielded some positive findings. Notably, the performance of the hypercomplex classifiers was found to be comparable as the original classifier does. This is an encouraging result, as it suggests that hypercomplex classifiers could be a generalized option to design model structure, holding their own advantage against more commonly used design. This finding is able to show more evidence of the use of hypercomplex classifiers as replacement of traditional fully-connected layers in various applications, further solidifying their place in the field of machine learning.

When it comes to contrastive learning, an interesting observation could be observed: the performance of the model tends to improve when the number of training epochs is increased, comparatively longer training epochs have been proven as an attribute of contrastive learning in multiple research. From results, it is evident that real-value classifiers exhibit a marked improvement in accuracy when trained using contrastive learning techniques. This suggests that contrastive learning could be particularly beneficial for models that work with real-valued classifiers. However, the same could not be said for hypercomplex classifiers. Unlike their real-valued counterparts, hypercomplex classifiers do not show a similar indication for improvement with contrastive learning. The reasons for this disparity are not immediately clear and further

research is required. It could be due to the inherent complexities of the hypercomplex domain or perhaps certain characteristics of the contrastive learning method itself. Also it's important to note that this result does not diminish the potential effectiveness of contrastive learning.

In table 9, the Euclidean distance metric was employed as a means to evaluate the compactness and distinctiveness of the clusters formed by each label following the fine-tuning process. The analysis from the result revealed some intriguing results, when the evaluation is examined in two out of the three datasets, the least sum of euclidean distances with best accuracy was observed in the hypercomplex classifier without employing contrastive learning, while in the another dataset only the least sum of euclidean distances was confirmed in the hypercomplex classifier training with contrastive learn having huge difference compared to others, however the best accuracy was not confirmed in the same model set. Only from the outcome gained from euclidean distances, this finding could suggest that this particular classifier is able to generate more compact and distinct clusters compared to the other model sets. Compactness and distinctness of clusters are desirable characteristics as they indicate that the classifier is able to effectively distinguish between different categories or labels in the dataset. It simply suggests that in some specific datasets or tasks, hypercomplex classifiers might show more advantages over traditional full-connected classifiers. The effectiveness of contrastive learning could vary depending on modification of the hyperparameter, the complexity of the model, and the particular task at hand.

KL convergence is applied to observe performance of the cluster result, with smaller values indicating better performance gathering the same labels together and vice versa. The results obtained from the pairwise comparisons between models with and without contrastive learning have provided some notable insights. One significant observation is that across all the comparisons, hypercomplex classifiers consistently outperformed real-valued classifiers when it came to clustering datasets with the same labels. This was true regardless of whether contrastive learning was implemented or not. This finding underscores the potential superiority of hypercomplex classifiers in handling complex data structures and relationships, demonstrating their robustness and versatility. However, an interesting exception was observed in the case of SST-2 datasets when models were trained without contrastive loss. In this particular scenario, the real-valued classifiers showed better accuracy than the hypercomplex classifiers. This could be attributed to the unique characteristics of the SST-2 datasets or perhaps certain nuances in the way the models were trained without contrastive loss.

Yet, despite this exception, the overall results suggest that the proposed method, which presumably leverages hypercomplex classifiers, exhibits stronger representation learning skills. This is evident from its ability to effectively distinguish between different label meanings, a crucial aspect of many machine learning tasks. Representation learning is all about learning to identify and extract the underlying structure or patterns in the data. In this regard, the proposed method appears to be quite effective, as it has demonstrated a strong ability to differentiate between the meanings of different labels. However, it's important to note that these results are based on the datasets and conditions used in this study. The performance of different models and methods could vary based on the specifics of the task, the nature of the data, and other factors. Therefore, while these findings provide valuable insights, further research and experimentation may be needed to fully understand the strengths and limitations of the proposed method and to optimize its performance for different applications.

There is observation showing that in training process, hypercomplex classifier has more frequent deviation when small hidden layers are applied, which could lead to a relatively more dramatic fluctuating learning loss compared to traditional classifier, which is main reason to set early stop with special predefined rule in order to have a complete training process for representation learning, thereby there could be fairly enough comparison between different classifier

to achieve a appropriate and faithful research. To achieve a stable training process and reliable result, it is recommended to set hidden layers with bigger value.

Table 9 – Three evaluation metrics are shown above, from top to bottom are sum of euclidean distance of each pair in each label, KL divergence and accuracy. Experiment was conducted with 1 runtime. All least Euclidean distance sums are confirmed in hypercomplex classifiers and 2 of them achieved best accuracy. Pairwise comparison between two classifiers for KL divergence show that most of least value is gained by hypercomplex classifiers except SST2

Euclidean distance	SNIPS	TREC	SST2
ROBERTA-16	125797914	78367717	251057524
QROBERTA-16	**94393822**	**66360602**	266231754
ROBERTA-C-16	192695851	77738803	276453023
QROBERTA-C-16	150421002	76471871	**159591449**
KL divergence	SNIPS	TREC	**SST2**
ROBERTA-16	1.6436	1.4155	**1.9791**
QROBERTA-16	**1.5415**	**1.1768**	2.219
ROBERTA-C-16	1.6231	1.5662	1.7692
QROBERTA-C-16	**1.511**	**1.4904**	**1.5207**
Accuracy	SNIPS	TREC	**SST2**
ROBERTA-16	0.9377	0.5373	**0.7246**
QROBERTA-16	**0.9501**	**0.6973**	0.6119
ROBERTA-C-16	0.9023	0.5118	0.5605
QROBERTA-C-16	0.9388	0.5994	0.5207

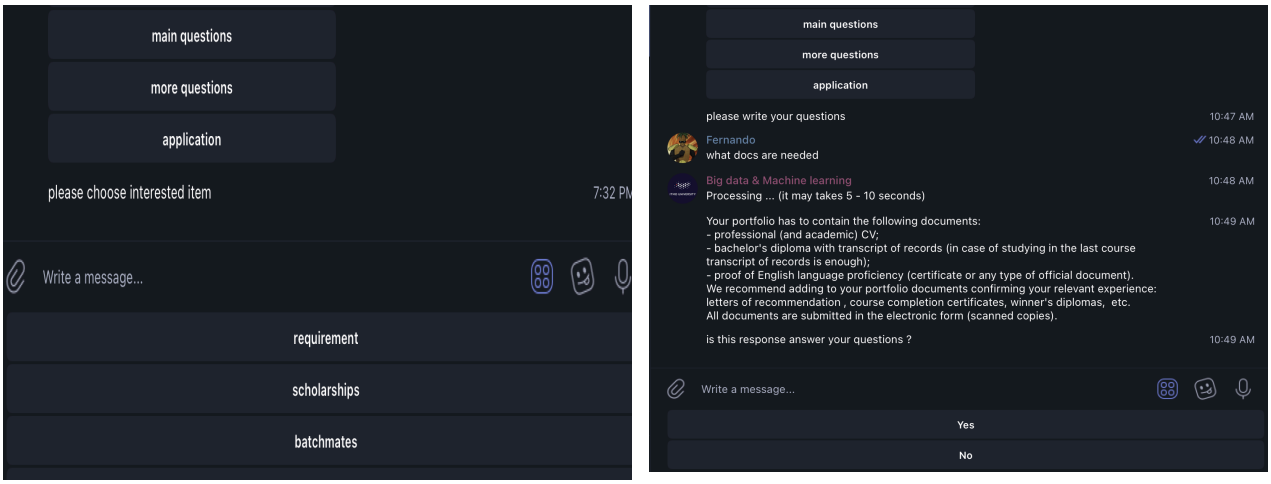
4.3 Demonstration of intellectual support system

The demonstration and functionality of the current system could be best understood through visual clear instructions. Once users initiate the process by clicking the "start" button, they are greeted with a warm welcome message, followed by a personalized menu featuring the user's name. This customized greeting helps establish a friendly and engaging interaction right from the start.

The menu offers two primary options to users for providing information: "Main Questions" and "Write Questions". These options cater to different user preferences and ensure that the platform is versatile and user-friendly. "Main Questions" comprises six predefined questions, carefully chosen based on the extensive experience of the admission group. These questions are deemed to be of most importance and are expected to cover the majority of user queries. The mechanism for this option involves buttons for selection. Users could simply click on the question that best suits their query, making the process straightforward and convenient. A visual representation of this feature is shown in fig.10, demonstrating its user-friendly and intuitive design.

On the other hand, "Write Questions" provides a more open-ended avenue for users who prefer to articulate their queries in their own words. This allows for more personalized and specific

inquiries, catering to the unique needs and concerns of each individual user. This section is currently composed of 23 topics of questions and requires text to be processed in a NLP model to predict the most possible answer.



(a) Main Questions (b) Write Questions

Figure 10 – Demonstration of two types of question-answering

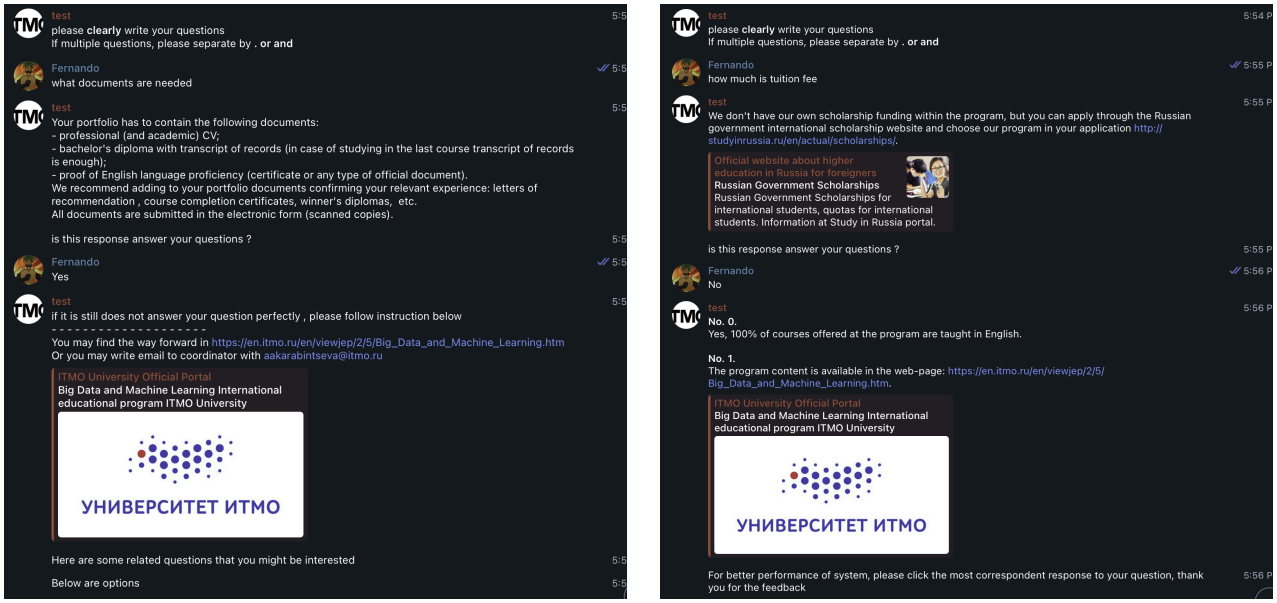
Once users pose a question, they are asked to evaluate the system's predicted answer and provide their feedback. This step is integral to this system's learning process as it helps us understand the effectiveness of prediction from models and make necessary adjustments. This feedback mechanism serves two main purposes. First, it allows the system to gauge user satisfaction with the provided answers, giving more insights into areas where the NLP model excels or needs improvement. Second, it gives users a sense of involvement and control over their interactions, enhancing their overall experience. If a user's evaluation is positive, the system takes this as an indication that the answer met or exceeded the user's expectations.

However, if the evaluation is not positive, the system interprets this as a need for improvement in the provided response. In such scenarios, the system provides possible answers for the user to select and correct. This allows for an immediate rectification, ensuring that the user's query is effectively addressed. Moreover, it provides valuable data for the system to learn from and improve its future responses. This process is visually represented in fig.11. In both cases, after the user's feedback and subsequent interactions, the current inputs are updated into the database. This ensures that all interactions are recorded and used to continuously refine and improve the system's performance.

In any case, the system moves on to provide recommendation options. These are presented in the format of questions, designed to guide the user's interaction and prevent the need for manual question writing. This feature not only saves time for the user but also helps in generating more accurate answers by aligning with the system's predefined question patterns. This process is visually demonstrated in fig.12.

In addition to the primary "Main Questions" and "Write Questions" options, the menu offers a range of other useful features designed to assist and guide users throughout their interaction with the system. The "Help" section is effectively a user instruction, explaining the functionality of each button on the platform. It acts as a guide, providing clarity about the system's operations and

ensuring users could navigate the platform with ease. Whether it is a first-time user or needs a refresher, the "Help" section is there to support you. Next, there's the "Contact" option, which provides users with the email address of the relevant department in the university. This feature ensures that users always have a method of direct communication for more complex queries or when they need more detailed information about the admission process, details are attached in APPLICATIONS A. By providing this contact information, it could be more affirmative that users could receive comprehensive assistance beyond what the system could offer. Lastly, the "Application" option serves a critical function - it redirects users to the application page of ITMO University. This feature simplifies the admission process by providing a direct link to the application form. Users could easily fill in their basic data to complete the first stage of the admission process, making it a seamless transition from Q&A to admission process. In essence, these additional menu options - "Help", "Contact", and "Application" - have been thoughtfully integrated to provide comprehensive support to users. Each feature plays a distinct role, ensuring that users have all the resources they need for a smooth and successful interaction with the system, and ultimately, a successful admission process.



(a) Positive feedback (b) Negative feedback

Figure 11 – Demonstration of self-correction where two actions with two different feedback from users

The admin mode in this system serves a crucial role in ensuring the accuracy and effectiveness of the platform's responses. The primary function of this mode is to correct the questions to which the system is unable to provide satisfactory answers. This mechanism allows us to continually refine and improve our model's performance, ensuring that it remains reliable and accurate, demonstration is shown in fig.13. When a user logs into the system, one of the first things comes into attention is an automatic notification. This notification provides a count of the questions that need correction. This feature is designed to help administrators prioritize their tasks effectively. By knowing the volume of questions that require attention, the administrators could plan their

workload and ensure that all necessary corrections are made in a certain range of time. The correction process involves a review of the system's responses to these questions for administrators to easily assess the appropriateness of the response and make necessary adjustments to improve its accuracy. This could involve modifying the system's response, updating the database with new information, or tweaking the system's algorithms to improve its prediction capabilities.

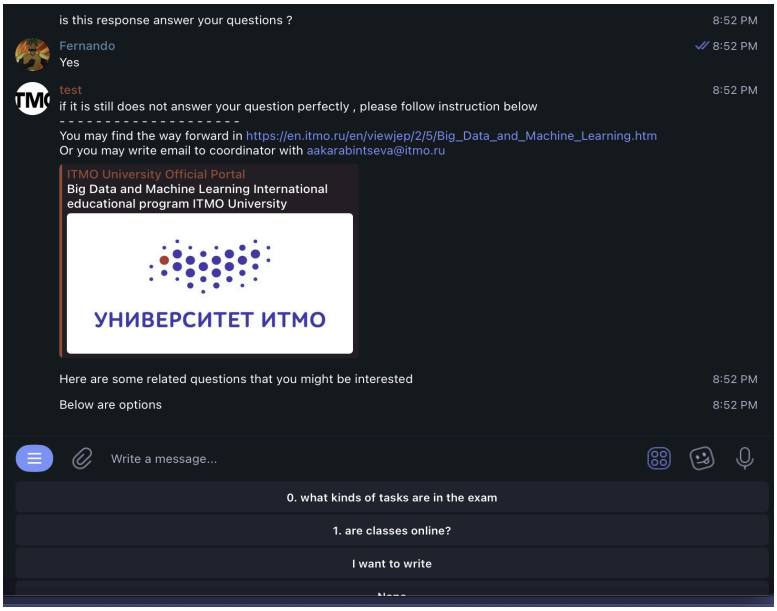


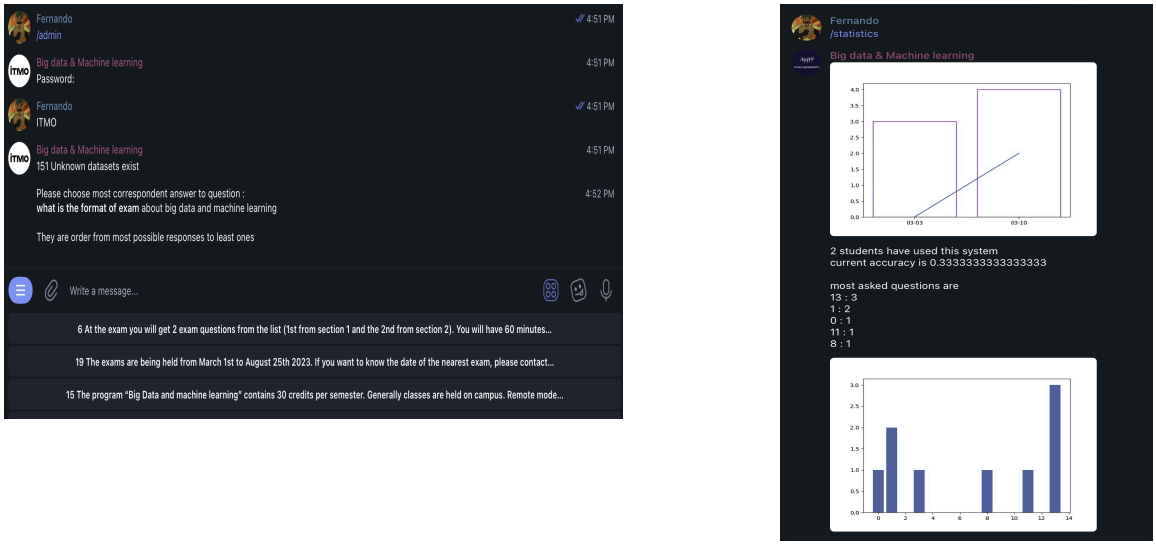
Figure 12 – Demonstration of recommendation mechanism

Focusing the question-answering mechanism on the Telegram platform has multiple advantages. The most significant benefit is the expected reduction in workload for the staff. By automating the process of answering questions, staff time is expected to be freed up and allow them to focus on more complex tasks that require human intervention. This optimized distribution of tasks enhances overall productivity and efficiency. Once a question that was previously unknown to the system is answered, it gets added to the system's knowledge base. This means that the system learns from every interaction it has, continuously expanding its database of known questions and answers.

The advantage of this learning process is twofold. Firstly, it helps in improving the system's capability to handle a broader range of queries. As the system encounters and learns from more questions, it becomes better equipped to provide accurate answers to a wider variety of user queries. Secondly, it enriches the user's experience with the system. When users interact with the system the next time, these newly added questions and their corresponding answers would be displayed on the user interface. This feature is expected to simplify the Q&A process without additional contact with real-human beings for the student side and prevent the university side from forgetting to respond to questions until “unknown” questions finish in the database by providing the notification from the system, moreover enhancing user’s trust and reliability of the system. In summary, by centralizing the question-answering process on the Telegram platform, this feature not only improves efficiency and productivity but also enhances the user experience and the system's capabilities.

Security is paramount in any system, and this system design is no exception. To ensure the safety and integrity of the platform, password authentication has been implemented for the admin interface. This feature is essential to prevent unauthorized access and to protect the system's data

and functionalities. Given the high level of access and control that the admin interface provides, there is a risk that a user might accidentally activate it.



(a) Response correction (b) Statistics report

Figure 13 – Demonstration of response correction function and statistics graph of visitors status and model accuracy confirmation for monitor purpose

To mitigate this risk, a two-step process has been set up to activate the admin interface. The first step is to send a specific command to the system. This command signals the system to activate the authentication step. It's a preliminary measure to ensure that only those who are aware of this command could initiate the process to access the admin interface. Following this, the second step requires the user to enter the correct password. Only upon entering the right password does the system grant access to the admin interface. This double layer of security further enhances the safety of the system. The system also provides admins the flexibility to change their password at any time from within the admin interface. This feature is crucial for maintaining the security of the system, as it allows admins to regularly update their passwords, reducing the risk of unauthorized access.

Looking ahead, further development enhances this system by leveraging the power of machine learning and NLP techniques. Specifically, the ultimate purpose is to develop and train two new models: a spam detection model and a sequence recommendation model. Spam messages could be a nuisance, detracting from the user experience and affecting the system's performance. To counter this, it is required to record all spam messages encountered by the system. Over time, as enough data is accumulated, this information would be used to train a spam detection model. This model would be designed to identify and filter out spam messages, helping to maintain the quality of user interactions and ensure the smooth functioning of the system. In addition to this plan, records of the order in which questions are asked by users would also be kept with datetime. This data is valuable as it could give insights into user behavior and preferences. Once collected data reach a specific amount, the sequence recommendation model would be trained to replace RoBERT model to serve the next recommended option for users. This model would analyze the sequence of

questions and use this information to recommend the most relevant subsequent questions to users from previous records. This feature would make the user experience more intuitive and personalized, effectively guiding users through their interactions with the system.

At present, the current system relies on a fine-tuned RoBERT model to drive the recommendations. This model is capable of detecting possible options related to the current user input. This is a dynamic, context-sensitive approach that allows the system to provide relevant and personalized recommendations. Consider, for example, a user input such as "tell me about an exam". The RoBERT model doesn't just look at this query in isolation, but considers the broader context and related options. It might activate an answer about when the exam is scheduled, but it could also bring up other pertinent information. This could include the format of the exam, resources for exam preparation, examples of past exam questions, and more, as the keyword “exam” is included in other instances. Therefore, even a simple query like "exam" could open up a wide range of related options. The RoBERT model's ability to understand context and recognize these relationships is what makes it so powerful. It could provide users with a comprehensive set of options related to their initial query, helping them explore the topic in greater depth. In this way, the RoBERT model serves as a valuable tool in this recommendation mechanism. It ensures that users are presented with a rich, diverse set of options that are relevant to their interests and needs. By doing so, it enhances the user experience, making interactions with the system more engaging and informative.

CONCLUSION

This paper provides a comprehensive outline of the design process of the NLP intellectual system by using multiple NLP approaches. The work encompasses the topic of how to enhance the performance of digital helper for admission purpose in academic institution, from fine-tuning approach, like classifier modification and data augmentation, platform selection to the implementation of the recommendation model for prevention of knowledge hallucination, all based on limited datasets which could be a critical challenge for any kind of model in any kind of field to train a robust model.

For the optimization of the fine-tuning approach, two proposed strategies were mentioned: data augmentation and classifier modification in pre-trained NLP models. The proposed augmentation approach has proven its effectiveness in retaining the keywords of a sentence while rephrasing the context. This is done to ensure that the meaning remains unchanged, while also enriching the context of the selected keyword. The proposed hypercomplex classifier offers another significant advancement in the system. It has the ability to reduce parameter size compared to the original classifier while not compromising performance. In fact, the proposed classifier has demonstrated comparable, and in some specific datasets, superior classification capabilities as evidenced in section 5.2. The amalgamation of all these modified approaches forms the basis for the proposed system's workflow design. It's important to note that this design is tailored to the provided datasets, which means its efficacy could not be guaranteed if applied to other datasets in different occasions. However, each approach has been designed with the aim of improving upon the original approaches in the meanwhile generalizing in other NLP models as well.

In the phase of developing enrollee intellectual support system, the augmented datasets eventually were increased to 12 sentences per example from original datasets by using proposed data augmentation method, which is expected to expand knowledge maximally from original datasets, while model was modified by implementing the proposed hypercomplex classifier.

The ultimate system is supposed to handle the tasks originally belonging to the admission team, such as conversation with users, enriching knowledge and sustainably proceeding the workflow in the future. In table 10, conclusions are listed with concerns and further development plan, while the following reflects the opening question.

Q. How to design a system with minimum human interference ?

A. "Sustainable workflow of proposed enrollment support system" - This involves creating a streamlined, efficient workflow that could sustain the operations of an enrollment support system. This might involve automating routine tasks, optimizing data pipelines, and developing a self-correction mechanism from user's feedback that could identify and rectify bottlenecks. The aim is to build a system that is consistent, scalable, and adaptable to changing requirements.

Q. How proposed data augmentation methods could perform for expansion of datasets ?

A. Yes, it shows a trade-off between variety and label-conserving ability compared with previous works, in which pos-tagging could keep the keyword to maintain the core meaning while modifying contexts. This contribution involves augmenting the training data using pos-tagging by including keywords in a prefix sentence to prevent the loss of original meaning. This could help the model understand language patterns and syntax better, leading to generalization of unseen datasets.

Q. Could hypercomplex classifiers be more discriminative for different labels than fully-connected layers in pretrained NLP models ?

A. Yes, from the result of the experiment, it shows comparable performance as fully-connected layers do in the meantime it contains less parameter size. This contribution focuses on modifying the architecture of a pre-existing NLP model. By replacing hypercomplex classifier layers with fully-connected layers, the model may gain additional flexibility and scalability. This could potentially improve the model's ability to make accurate predictions and adapt to new data.

Q. How to reduce the mistake of model prediction ?

A. One active and one passive strategy are implemented to conquer this challenge: recommendation system and self-correction mechanism with feedback from users, respectively.

Active contribution involves implementing a recommendation system that combines sequential algorithms and NLP models. Sequential algorithms could capture and predict user behavior based on previous input, while NLP models could understand and interpret user language. Together, they could provide more personalized and contextually relevant recommendations.

Passive contribution involves implementing a mechanism similar to “reinforcement learning” that uses human feedback to improve itself. This continuous learning method allows the system to self-correct by learning from its mistakes and successes, thereby expanding its knowledge and improving its performance over time.

Concerns

While the proposed data augmentation methods used in the workflow have helped mitigate the issues posed by insufficient datasets, it's important to note that GPT2 has some limitations when it comes to rephrasing sentences, particularly in comparison to its more advanced successors, GPT3 and GPT4. GPT2, while impressive in its own right, might not always produce the most diverse or grammatically accurate sentence structures, which could lead to models learning from less-than-optimal data. In my opinion, to truly harness the full potential of data augmentation techniques and handle real-world problems, employing a more advanced model such as GPT3 or GPT4 would be a much better alternative. These models have been refined and enhanced to create more syntactically varied and grammatically accurate sentences. By utilizing these advanced models for data augmentation, it is expected to generate a broader range of high-quality, nuanced sentences. This would not only enrich datasets but also ensure that the models are learning from more accurate and diverse examples. Furthermore, this switch could also help in preventing models from acquiring 'defective knowledge', that is, learning incorrect or misleading patterns from poorly constructed sentences. It's crucial to remember that the quality of training data fundamentally drives the performance of NLP models. By investing in better data augmentation techniques via advanced models, in essence, a more robust and reliable model could be trained to recognize more unseen datasets. However, it's also important to consider the computational cost and resource requirements associated with these advanced models. While they offer improved performance, they might also require more robust infrastructure. Thus, a balance must be struck between the level of model sophistication and the available resources.

Scalability, indeed, poses a significant concern, particularly when the system is set to expand and accommodate similar information but associated with different educational programs. The

challenge lies in ensuring that the information provided remains accurate and relevant to each unique educational program, despite the similarities in their datasets. At present, the proposed solution is to train the models with prefixes of program names. This strategy helps to distinguish the similar datasets by associating each piece of data with a specific program. While this approach works effectively for now, it could potentially lead to complications in the future, especially when multiple new program information is added into the system. Decoding the user's text might not present a significant issue. By prompting all user's text with the prefix of the program name before the original input, the well-trained model should be able recognize the difference and provide responses that are most related to the specified program. However, potential confusion may arise when the system is required to provide recommendation items from the recommendation system. To counter this, the current solution is to define the range of numeric labels before presenting the recommended item. This step is crucial in eliminating the risk of knowledge confusion by providing incorrect information from different educational programs. But while it ensures accuracy, it's important to consider its impact on the system's processing time and, subsequently, the user experience. For long-term development, this method could potentially lead to time-consuming processing by defining the range of numeric labels each time in the database, combined with the time generating a response, could result in delays. Delays in response times could lead to an unpleasant user experience, which is unfavorable for system providers.

Table 10 – The summary of conclusion

Contribution	**Challenge**	**Result**
Minimize human interference	Less human interference	1) Intellectual system with periodical routine & discriminative ability
GPT DA with Pos-tagging	Insufficient training dataset	Trade-off between variety & label-conserving ability
Hypercomplex classifer		More light-weight & better performance
Prevent prediction mistake	NLP model instability	1) Self-correction from user's feedback 2) Recommendation mechanism
Concern	**Problem**	**Solution**
GPT version	GPT2 < GPT3, GPT4	1) Newer GPT 2) Combination of different DA method
Scalabiility	Knowledge confusion	1) To location range of labels in database 2) Combination with RAG approaches
Further develop	**Current limit**	**Expected outcome**
Multi-languages	English Only	Globally friendly
Preliminary checking of admission documents	Not supported	Prevent unnecessary time & resources
Student selection	Not supported	Enhance quality of students

Future development

The future of university admissions is moving towards intelligent, automated systems that could handle complex tasks with ease and precision. These systems are not just about automating processes; they're about improving the quality of interactions, personalizing experiences, and making admissions more efficient and effective. At the current moment, the system is still at the very beginning phase where service only covers the functionality for Q&A of admission and enrollment purposes. One key area of potential improvement could be the implementation of multi-language support. This would address language barriers and facilitate simpler and more effective communication with prospective students and their guardians, who may not be proficient in the language of instruction. By catering to a diverse range of linguistic backgrounds, the system could truly become global and accessible to all. Another feasible enhancement could be the addition of a mechanism for preliminary checking of admission documents and previous educational data. This could process an initial selection based on the received datasets, effectively reducing the number of candidates before the next round of selection by university staff. This would prevent unnecessary time and resources from being spent on candidates who don't meet the minimum requirements, thereby enhancing the efficiency of the admission process. Furthermore, the system could be designed to continuously learn and improve from each interaction, becoming progressively better at predicting student success and making admission decisions. This would result in a more refined admission process, where each decision is backed by data and sophisticated algorithms. As specified in the concerns when datasets expand with personal information or implementation of more tasks in the system, RAG might be better alternatives to complete more dynamical and complex processing. RAG's key strength lies in its ability to pull relevant information from a vast knowledge base, making it highly adaptable to changing data and tasks. As the data expands or becomes more complex, a RAG system could still effectively retrieve and utilize the necessary information. This makes it an ideal choice for scenarios where datasets are continuously growing or diversifying, or where the system has to handle an increasing number of tasks. When personal information is involved, the dynamic nature of RAG could be beneficial. Instead of relying on static pre-training data, which might be challenging to deal with elastic datasets, the RAG approach could retrieve data from a broad and anonymized knowledge without long-terning fine-tuning model. On the other hand, Fine-tuning, by its nature, is designed to adapt pre-trained models to perform better on specific tasks or domains. This could enhance the model's performance in these specific areas.

The integration of RAG and fine-tuning approaches has the potential to drive innovation and foster collaboration across various LLMs in numerous industries. This blend of approaches is a promising step towards increasing the efficiency and effectiveness of these models. Initially, the focus should be on training the models with structured datasets. Structured data is highly organized and easily searchable, making it an ideal starting point for training LLMs. This initial training phase allows the models to comprehend the basic patterns and relationships within the domain-specific data. Once the models have been fine-tuned with structured data, they could then move onto the process of generating questions and answers. This is where the power of RAG comes into play. With its ability to retrieve relevant information from a large corpus of documents and synthesize it into coherent responses, RAG provides a sophisticated approach to question-answering tasks. It allows the models to construct detailed and contextually accurate responses, thereby enhancing their performance. The combination of structured and domain-specific datasets, along with RAG and fine-tuning approaches, could lead a model to become a better problem solver. This blend of methodologies ensures that the LLMs are not only well-trained but also adaptable and capable of handling a wide range of tasks. They would be better equipped to understand and respond to complex queries, making them invaluable tools in a variety of industry settings.

Beside admission purposes, there could be more potential applications for educational purposes. Indeed, there are endless possibilities for further development and expansion of the system's utility beyond just admissions and enrollment aspects. One such potential area could be its implementation as an educational helper, designed to respond to academic queries from students. Considering that staff in each department could be as scarce as in the admission team, yet are required to handle queries from hundreds of students, such a system would be a game-changer in terms of efficiency and responsiveness. The system could be designed to function as an intellectual tutor, capable of engaging in meaningful discussions with students about domain-specific knowledge. It could provide explanations, answer questions, and even engage in stimulating intellectual debates, thereby supplementing the teaching process and enhancing the student's learning experience. Moreover, It could also behave like a mentor, guiding and leading students to overcome academic difficulties by providing valuable advice, suggesting resources, and even offering motivational support in the student's educational journey. Furthermore, the system could be customized according to each student's unique needs and preferences. The system could also assist students with class selection, providing recommendations based on their academic performance, interests, and career goals by analyzing course offerings, prerequisites, and student reviews to suggest the most suitable classes for each student.

REFERENCES

[1] ALBERTI, Laura, et al. The COVID-19 impact on higher education stakeholders and institutional services. Rowman & Littlefield, 2022.

[2] BARRETT, Mandy, et al. Using artificial intelligence to enhance educational opportunities and student services in higher education. Inquiry: The Journal of the Virginia Community Colleges, 2019, 22.1: 11.

[3] CHANDRA, Yogi Wisesa; SUYANTO, Suyanto. Indonesian chatbot of university admission using a question answering system based on sequence-to-sequence model. Procedia Computer Science, 2019, 157: 367-374.

[4] HERSI, Abdikafi Hashi, et al. An intelligent Somali language chatbot serving as an online admission help desk. SORER: Mogadishu, Somalia, 2021.

[5] WANG, Cunxiang, et al. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. arXiv preprint arXiv:2310.07521, 2023.

[6] CHEN, Sanyuan, et al. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. arXiv preprint arXiv:2004.12651, 2020.

[7] RAFAILOV, Rafael, et al. Direct preference optimization: Your language model is secretly a reward model. Advances in Neural Information Processing Systems, 2024, 36.

[8] KIRKPATRICK, James, et al. R. Proceedings of the national academy of sciences, 2017, 114.13: 3521-3526.

[9] LEWIS, Patrick, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 2020, 33: 9459-9474.

[10] NEELAKANTAN, Arvind, et al. Text and code embeddings by contrastive pre-training. arXiv preprint arXiv:2201.10005, 2022.

[11] SOUDANI, Heydar; KANOULAS, Evangelos; HASIBI, Faegheh. Fine Tuning vs. Retrieval Augmented Generation for Less Popular Knowledge. arXiv preprint arXiv:2403.01432, 2024.

[12] OVADIA, Oded, et al. Fine-tuning or retrieval? comparing knowledge injection in llms. arXiv preprint arXiv:2312.05934, 2023.

[13] SIMARD, Patrice Y., et al. Best practices for convolutional neural networks applied to visual document analysis. In: Icdar. 2003.

[14] KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 2012, 25.

[15] ZHANG, Hongyi, et al. mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412, 2017.

[16] BELINKOV, Yonatan; BISK, Yonatan. Synthetic and natural noise both break neural machine translation. arXiv preprint arXiv:1711.02173, 2017.

[17] KARPUKHIN, Vladimir, et al. Training on synthetic noise improves robustness to natural noise in machine translation. arXiv preprint arXiv:1902.01509, 2019.

[18] WEI, Jason; ZOU, Kai. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. arXiv preprint arXiv:1901.11196, 2019.

[19] FELLBAUM, Christiane (ed.). WordNet: An electronic lexical database. MIT press, 1998.

[20] ZHONG, Zhun, et al. Random erasing data augmentation. In: Proceedings of the AAAI conference on artificial intelligence. 2020. p. 13001-13008.

[21] KOLOMIYETS, Oleksandr; BETHARD, Steven; MOENS, Marie-Francine. Model-portability experiments for textual temporal analysis. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies. ACL; East Stroudsburg, PA, 2011. p. 271-276.

[22] ZHANG, Xiang; ZHAO, Junbo; LECUN, Yann. Character-level convolutional networks for text classification. Advances in neural information processing systems, 2015, 28.

[23] KOBAYASHI, Sosuke. Contextual augmentation: Data augmentation by words with paradigmatic relations. arXiv preprint arXiv:1805.06201, 2018.

[24] WU, Xing, et al. Conditional bert contextual augmentation. In: Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part IV 19. Springer International Publishing, 2019. p. 84-95.

[25] ANABY-TAVOR, A., et al. Not Enough Data? Deep Learning to the Rescue! arXiv 2019. arXiv preprint arXiv:1911.03118, 1911.

[26] RADFORD, Alec, et al. Language models are unsupervised multitask learners. OpenAI blog, 2019, 1.8: 9.

[27] GAO, Fei, et al. Soft contextual data augmentation for neural machine translation. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. p. 5539-5544.

[28] NIU, Tong; BANSAL, Mohit. Adversarial over-sensitivity and over-stability strategies for dialogue models. arXiv preprint arXiv:1809.02079, 2018.

[29] MIAO, Zhengjie, et al. Snippext: Semi-supervised opinion mining with augmented data. In: Proceedings of The Web Conference 2020. 2020. p. 617-628.

[30] ARTETXE, Mikel, et al. Unsupervised neural machine translation. arXiv preprint arXiv:1710.11041, 2017.

[31] KUMAR, Varun; CHOUDHARY, Ashutosh; CHO, Eunah. Data augmentation using pre-trained transformer models. arXiv preprint arXiv:2003.02245, 2020.

[32] VOLPI, Riccardo, et al. Generalizing to unseen domains via adversarial data augmentation. Advances in neural information processing systems, 2018, 31.

[33] FEI-FEI, L.; FERGUS, R.; PERONA, P. One-Shot learning of object categories. IEEE Trans. Pattern Recognition and Machine Intelligence.

[34] LESTER, Brian; AL-RFOU, Rami; CONSTANT, Noah. The power of scale for parameter-efficient prompt tuning. arXiv preprint arXiv:2104.08691, 2021.

[35] GAO, Tianyu; FISCH, Adam; CHEN, Danqi. Making pre-trained language models better few-shot learners. arXiv preprint arXiv:2012.15723, 2020.

[36] ANTONIOU, Antreas; EDWARDS, Harrison; STORKEY, Amos. How to train your MAML. In: International conference on learning representations. 2018.

[37] FINN, Chelsea; ABBEEL, Pieter; LEVINE, Sergey. Model-agnostic meta-learning for fast adaptation of deep networks. In: International conference on machine learning. PMLR, 2017. p. 1126-1135.

[38] CHIPMAN, Hugh A.; GEORGE, Edward I.; MCCULLOCH, Robert E. BART: Bayesian additive regression trees. 2010.

[39] NI, Jianmo, et al. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. arXiv preprint arXiv:2108.08877, 2021.

[40] GAUDET, Chase J.; MAIDA, Anthony S. Deep quaternion networks. In: 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, 2018. p. 1-8.

[41] PARCOLLET, Titouan, et al. Quaternion convolutional neural networks for end-to-end

automatic speech recognition. arXiv preprint arXiv:1806.07789, 2018.

[42] ZHU, Xuanyu, et al. Quaternion convolutional neural networks. In: Proceedings of the European Conference on Computer Vision (ECCV). 2018. p. 631-647.

[43] COMMINIELLO, Danilo, et al. Quaternion convolutional neural networks for detection and localization of 3D sound events. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019. p. 8533-8537.

[44] ZHANG, Shuai, et al. Quaternion collaborative filtering for recommendation. arXiv preprint arXiv:1906.02594, 2019.

[45] TROUILLON, Théo, et al. Complex embeddings for simple link prediction. In: International conference on machine learning. PMLR, 2016. p. 2071-2080.

[46] TAY, Yi, et al. Lightweight and efficient neural natural language processing with quaternion networks. arXiv preprint arXiv:1906.04393, 2019.

[47] ZHANG, Aston, et al. Beyond fully-connected layers with quaternions: Parameterization of hypercomplex multiplications with $1/n $ parameters. arXiv preprint arXiv:2102.08597, 2021.

[48] LE, Tuan, et al. Parameterized hypercomplex graph neural networks for graph classification. In: International Conference on Artificial Neural Networks. Cham: Springer International Publishing, 2021. p. 204-216.

[49] TANG, Zedong, et al. Skfac: Training neural networks with faster kronecker-factored approximate curvature. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021. p. 13479-13487.

[50] KANG, Wang-Cheng; MCAULEY, Julian. Self-attentive sequential recommendation. In: 2018 IEEE international conference on data mining (ICDM). IEEE, 2018. p. 197-206.

[51] SUN, Fei, et al. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM international conference on information and knowledge management. 2019. p. 1441-1450.

[52] KINGA, D., et al. A method for stochastic optimization. In: International conference on learning representations (ICLR). 2015. p. 6.

[53] FANG, Hongchao, et al. Cert: Contrastive self-supervised learning for language understanding. arXiv preprint arXiv:2005.12766, 2020.

[54] JAISWAL, Ashish, et al. A survey on contrastive self-supervised learning. Technologies, 2020, 9.1: 2.

[55] GRASSUCCI, Eleonora; ZHANG, Aston; COMMINIELLO, Danilo. PHNNs:

Lightweight neural networks via parameterized hypercomplex convolutions. IEEE Transactions on Neural Networks and Learning Systems, 2022.

APPLICATIONS

A System guidance

User

1. Help
   - This module offers a detailed overview of each feature available on the platform. It has been designed to guide users through the functionality of each section, ensuring a seamless navigation experience.
2. Contact
   - This section provides essential contact information for the various departments at ITMO University. It has been created to assist users with inquiries related to enrollment processes.
3. Main Questions
   - This component is a compilation of the most commonly asked questions from users. It serves as a quick reference guide for queries that have been addressed previously.
4. Write Questions
   - This is an interactive space that allows users to submit textual messages or voice messages. System would encourage users to provide feedback on the given responses. Unanswered or incorrectly answered questions would be forwarded to the "unknown" database for universities to review and rectify.
   - users are presented with recommendations based on their previous interactions and inquiries. These suggestions are tailored to align with the user's interests, offering a more personalized user experience.
5. Application
   - This section provides a direct link to the application form on the ITMO University website. It allows users to transition smoothly to the application process when they are ready to apply.

Admin

1. Correct Unknown Questions
   - This section enables the rectification of system-identified unknown questions. Options for correction would be provided sequentially, based on the prediction by a fine-tuned RoBERT model.
   - Once corrected, these answers would be dispatched directly to users when they open a chat on Telegram next time.
2. Update responses
   - This feature is designed to update predefined responses within the system database, ensuring that information remains new and accurate.
3. Change passwords
   - For security purposes, this section allows the users to update their passwords as needed.

4. Statistics

- This module offers insights into recent visitor activity on the chat platform. It has been designed to provide an understanding of user interaction patterns and model prediction status.

5. Add new programs
    - This option allows the insertion of new programs. Users could input JSON formatted files of datasets containing ordered questions and answers for this purpose.

Extra info

1. NLP model would undergo a retraining process under two specific circumstances:

a. Periodic Retraining: The model would be retrained at regular, predetermined intervals. This is to ensure that the model's understanding and performance are consistently up-to-date, reflective of any changes or trends in the data it processes.

b. Content-Based Retraining: Every time there is an update in program information or a new topic is added to a program, the model would be retrained. This is necessary because the addition of new information could significantly alter the context or content that the model needs to understand and process.

2. The sequence model would not initially be activated due to the absence of a dataset. NLP Model model would be responsible for recommendation items provided before the sequence model is activated, two different models activation used are:

a. RoBERT Model Activation: Despite the inactivity of the sequence model, the NLP model would be activated to provide recommendation options. This model could operate independently of the sequence model, offering valuable insights and suggestions based on its understanding of natural language.

b. SASrec Activation and Retraining: The SASrec model, a sequence recommendation model, would be activated after a predefined period. Following its activation, the model would undergo a retraining process every specific period. Regular retraining ensures that the model's recommendations remain relevant and accurate, considering any new data or changes in user behavior patterns over time.

B Related work list

The inception of the current research work was largely influenced by a comprehensive historical review spanning the last five years. This review focused specifically on intellectual enrollee support systems and digital assistants developed for academic purposes. The objective was to understand the evolution of these systems and the impact of technological advancements on their design and functionality in the meanwhile being inspired by these related works.

Over the past half-decade, there has been a significant shift in the application of technology for the development of academic digital systems. The initial phases were marked by relatively basic systems, often with limited functionality and a heavy reliance on predefined manual patterns in the database. However, with the rapid advancement in NLP technology and the increasing digitization of academic processes, there has been a fundamental transformation in the design and capabilities of these systems. In the early stages, the NLP community was largely dependent on techniques like keyword matching for these systems. Keyword matching is a straightforward method that matches user queries with predefined responses based on the presence of specific words or phrases. While this technique is effective for simple tasks, it had significant limitations in handling more complex tasks that require understanding context or intent. However, with the continual evolution of NLP technology, the community began to shift away from these rudimentary techniques. Instead, people

start to lean heavily on deep learning, specifically LLMs, to handle more complex tasks. These tasks often involve mimicking human conversation, encompassing elements such as reasoning, categorization, and contextual understanding.

Many previous works in this field have unfortunately been somewhat vague in their descriptions and methodologies, often simply referring to broad categories such as 'deep learning' or 'Natural Language Processing'. These unclear descriptions, while not entirely inaccurate, do not provide adequate insight into the specific techniques and processes employed. This lack of information could create a "black box" scenario where the details of the implementation or description of parameter setting are not fully understood, thereby making replication or further development challenging. To address this issue, the current work endeavors to provide a comprehensive and detailed presentation of all methodologies, techniques, and processes used. The intention is to enhance transparency and foster an environment that encourages further development for the intellectual system in the academic field. By providing specifics about the models used, the datasets trained on, the performance metrics evaluated, and the various stages of the development and testing process, the current work aims to set a new standard for future research. The goal is not merely to achieve impressive results, but also to illuminate the path taken to reach those results. This illumination should, in turn, enable other researchers to reproduce the work, build upon it, and potentially even improve it. The related works are listed in the table.

Table 11 – The comparison of related academic intellectual systems, NA shows the value is not specified in the report

Topic	Dataset	Country	Purpose	Techniques	Metrics / Effect
Using Artificial Intelligence to Enhance Educational Opportunities and Student Services in Higher Education	NA	USA	enrollment, completion of education	NA	Prevernt loss of potential studetns by over 20%
Indonesian Chatbot of University Admission Using a Question Answering System Based on Sequence-to-Sequence Model Answering System	2,506	Indonesian	Q&A purpose	Sequence-to-Sequence model & attention mechanism	BLEU Score: 44.68
NEU-chatbot Chatbot for admission of National Economics University	1500	Vietnam	admission counseling	BERT, Dual Intent and Entity Transformer	Accuracy : 90.29% from 50000 questions
Development of an AI Chatbot to Support Admissions and Career Guidance for Universities	1061	Vietnam	Admission,career guidance	TF-IDF Algorithm,SVM model, Easy data augmentation	Accuracy : 65 %

C Algorithm for proposed approaches

Algorithm 1 for Pos-tagging GPT2:

---

fine-tune phase

Input : Training datasets $D_{train} = \{(x_n, y_n)\}_{n=1}^{N_{data}}$, a datasets

with pair of label and sequence
Pretrained model $M \in$ {GPT2, Bert}
Classifier $C \in \{FC, PHM\}$

$Loss \in \{cross\ entropy\ ,\ contrastive\ loss\}$

Input : Θ, initial M parameters

Output : $\hat{\Theta}$, the trained parameters

for i = 1,2,3…N_{epoch} do

for n = 1,2, … N_{data} do

$P(\Theta) \leftarrow C(M(x_n|\Theta))$

$Loss(\Theta) = LossFunc(P(\Theta)_n, y_n)$

$\Theta \leftarrow \Theta - \eta \cdot \nabla Loss(\Theta)$

end

end

return $\hat{\Theta} = \Theta$

augmentation phase

$M_{\hat{\theta}}$, fine-tuning model

Input : N , num of augmentation per sentence

Output : $\hat{D}_{generated}$

for i = 1 … $[X_i, Y_i] \in D_{original}$ do :

$[\hat{X}_i, \hat{Y}_i]$ is combination of label and prefix token after random choosing between noun and verb based on Pos-Tagging

$\hat{D}_{generated} \leftarrow$ Synthesize N example of $[\hat{X}_i, \hat{Y}_i]$ using $M_{\hat{\theta}}$

end

return $\hat{D}_{generated}$

Algorithm 2 for hypercomplex classifier:

Input : Training datasets $D_{train} = \{(x_n, y_n)\}_{n=1}^{N_{data}}$, a datasets
with pair of label and sequence
Pretrained model $M \in$ {GPT2,RobertBert}
Classifier $C \in \{FC, PHM\}$
$Loss \in \{cross\ entropy, contrastive\ loss\}$

Input : Θ, initial M parameters
Output : $\hat{\Theta}$, the trained parameters
for i = 1,2,3…N_{epoch} do
    for n = 1,2, … N_{data} do
        $P(\Theta) \leftarrow C(M(x_n|\Theta))$
        $Loss(\Theta) = LossFunc(P(\Theta)_n, y_n)$
        $\Theta \leftarrow \Theta - \eta \cdot \nabla Loss(\Theta)$
    end
end
return $\hat{\Theta} = \Theta$

D Summary of experiment setting

Table 12 – The details of two experiments setting

Proposed methods	**Hypercomplex Classifer**	**GPT Data Aug with Pos-tagging**
Hypothesis	Stronger discrimination ability to handle little training datasets	Better augmented text to prevent human-annotation process
Model	RoBERT	GPT2
Compared Objects	(1) Fully-Connected Layer	(1) EDA (2) Back-Translation (3) CBERT (4) CGPT2
Evaluation metrics	(1) Accuracy (2) Euclidean distance	(1) N-grams (versatility) (2) Fidelity (meaning-preservation)

E Example of data augmentation

There is result in the previous section showing that proposed data augmented approaches with GPT2 and pos-tagging shows competitive performance compared with previous data augmented approaches with GPT2 by using predefined number of prefixes, table below showcases examples generated in experiments.

Table 13 – Comparison of generated text between proposed data augmentation by GPT2 method and previous data augmentation by GPT2, red text in GPT column shows predefined prefix, while in GPT with Pos-tagging column it shows the keyword captured by pos-tagging as prefix boundary method for further context generation

Original	GPT2	GPT2 with Pos-tagging
What are the requirements to enter the program?	- What are the best ways to improve my work skills at Microsoft - What are the tasks at the workplace and at home - What are the details of the programme	- What are the **requirements** to enter the exam - What are the **requirements** to enter the exam in an academic setting - What are the **requirements** to enter this field
Who will be my batchmates at the program?	- Who will be the host of my website during the tournament - Who will be the starting editor of this project - Who will be submitting the submission requirements	- Who will be my **batchmates** at my program - Who will be my **batchmates** at the camp - Who will be my **batchmates** when you graduate
What are the research topics for the Master's thesis at the program and how do I choose one?	- What are the requirements of a program offered by the State College of the United States - What are the tasks that people at your company are supposed to complete during my job - What are the conditions for certification	- What are the **research** activities that researchers are doing in the area of science and technology at the program - What are the **research** topics for this exam - What are the **research** topics for the Master's thesis

F Hypercomplex multiplication table

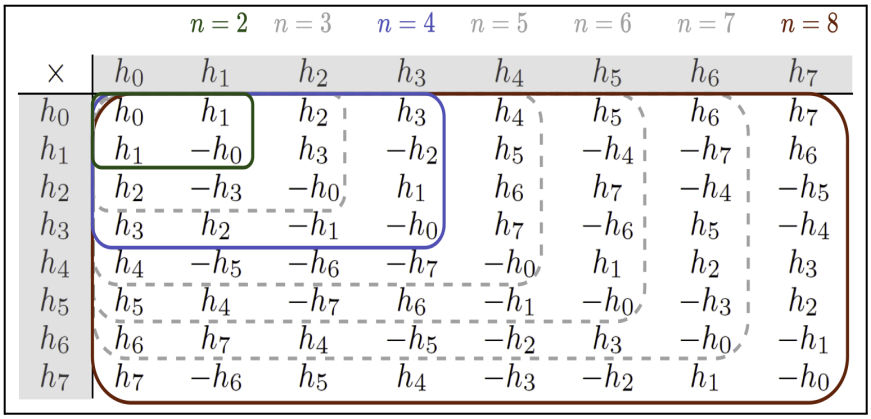


Figure 14 – An abstract table for hypercomplex multiplication. In columns, n = 2 with green line shows complex domain , while n = 4 is quaternion domain with blue line and n = 8 with red line octonion domain . There exist rules for mentioned domains, while others do not. The parameterized hypercomplex approaches have the capacity to learn patterns from domains with no rules. image is taken from[55]