Arab Academy for Science, Technology, and Maritime Transport

College of Artificial Intelligence

[Data Science]

B. Sc. Graduation Project

[IntelliLearn]

Presented By:

- Ahmed Nasser
- Adham Sherif
- George Youhana
- Magda Elroumany
- Sawsan Kassem
- Sohaila Khaled

Supervised By:

Dr. Ahmed El Shaer

Eng. Ahmed Metwali

Eng. Nagy Khairat

DECLARATION

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Bachelor of Science in Artificial Intelligence is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not, to the best of my knowledge, breach any law of copyright, and has not been taken from the work of others and to the extent that such work has been cited and acknowledged within the text of my work.

Student Name	Registration Number	Signature	Date
Adham Sherif	20101590	Adham Sherif	
Ahmed Nasser	20109959	Ahmed Nasser	
George Youhana	20200451	George Youhana	
Magda El-Romany	20109952	Magda El-Romany	
Sawsan Kassem	20200425	Sawsan Kassem	
Sohaila Khaled	20109961	Sohaila Khaled	

Arab Academy for Science, Technology and Maritime Transport

College of Artificial Intelligence

[Data Science]

Academic Year: 2024

Semester: 8

Graduation Project Summary Report

Project Title		
Supervisor(s)		
Team members	**Registration Number**	**Name**
	20101590	Adham Sherif
	20109959	Ahmed Nasser
	20200451	George Youhana
	20109952	Magda El-Roumany
	20200425	Sawsan Kassem
	20109961	Sohaila Khaled
Project Deliverables		

Abstract	In the digital age, transforming educational tools and resources is crucial for enhancing learning experiences and accessibility. This project, "IntelliLearn," leverages advanced AI technologies, including OpenAI's Whisper and xAI's Grok, to develop a comprehensive system for text analysis, summarization, and generation. The system allows users to upload a PDF, audio recording, or video. If the uploaded file is an audio recording or video, it undergoes speech-to-text (STT) processing to extract the text, which is then fed into a large language model (LLM). For PDFs, the text is directly processed by the LLM. The extracted or uploaded text is then utilized for various tasks such as text generation, code generation, summarization, and question answering. By integrating robust speech recognition capabilities with deep learning insights, IntelliLearn provides efficient content processing and user-friendly interaction, making it an invaluable tool for students, educators, and professionals. The system aims to bridge the gap between vast information sources and users, facilitating better understanding and retention of complex information. This project not only enhances the accessibility of educational content but also streamlines the process of information extraction and utilization, thereby significantly contributing to the field of educational technology.
System Constraints	The development and implementation of IntelliLearn are subject to the following constraints: 1. Computational Resources: The system requires significant computational power to process large datasets and run complex AI models. This necessitates access to high-performance servers and GPUs, which can be costly and may limit scalability. 2. Data Quality and Availability: The effectiveness of IntelliLearn heavily depends on the quality and diversity of the training data. Obtaining high-quality, representative datasets for training AI models can be challenging, particularly in specialized educational domains.

<table>
<tr><td></td><td colspan="2">3. Latency and Real-Time Processing: Ensuring low latency for real-time speech-to-text and text processing is critical for user experience. Achieving this requires optimizing algorithms and potentially investing in advanced hardware to meet performance requirements.
4. Integration with Existing Systems: IntelliLearn needs to integrate seamlessly with various educational platforms and tools. Compatibility issues and the need for extensive customization can pose challenges during deployment.
5. User Accessibility: The system must be designed to be accessible to a wide range of users, including those with disabilities. This involves adhering to accessibility standards and potentially developing specialized interfaces and features.
6. Regulatory Compliance: Adhering to data protection regulations, such as GDPR and CCPA, is essential. Compliance requires ongoing monitoring and adjustments to data handling practices to meet legal requirements.</td></tr>
<tr><td>Project Impact</td><td colspan="2"></td></tr>
<tr><td rowspan="3">Team Organization</td><td>Name</td><td>Tasks achieved</td></tr>
<tr><td>Ahmed Nasser</td><td>Lead developer for the speech-to-text module.
Implemented and optimized the Whisper and Wav2vec2 models for accurate audio transcription.
Ensured integration of the speech-to-text module with the summarization component.</td></tr>
<tr><td>Adham Sherif</td><td>Project manager overseeing the development process.
Coordinated team activities and ensured milestones were met.
Conducted performance evaluations and optimization of the AI models.</td></tr>
</table>

	George Youhana	Developed the text summarization module. Integrated transformer-based models like BERT and GPT for high-quality text summarization. Worked on the evaluation metrics to ensure the accuracy and relevance of summaries.
	Magda Elroumany	Responsible for user interface design and user experience enhancement. Designed a user-friendly interface for uploading files and interacting with the system. Conducted usability testing to ensure accessibility and ease of use.
	Sawsan Kassem	Managed the data pipeline and preprocessing tasks. Ensured high-quality data for training and testing the AI models. Implemented data security measures to protect user information.
	Sohaila Khaled	Developed the code generation module. Integrated the module with the text generation component to handle user queries. Worked on maintaining and updating the knowledge base for accurate code generation.
Ethics/Safety	Ethics and safety are paramount considerations in the development and deployment of IntelliLearn. The system is designed with the following ethical principles and safety measures: 1. Data Privacy and Security: IntelliLearn ensures the confidentiality and integrity of user data by implementing robust encryption protocols and secure data storage solutions. Personal information and educational content are handled with the utmost care to prevent unauthorized access and	

	breaches. 2. Bias Mitigation: The AI models used in IntelliLearn are continuously monitored and evaluated to minimize biases in generated content. Efforts are made to train the models on diverse datasets, promoting fairness and inclusivity in educational materials. 3. Transparency: IntelliLearn maintains transparency in its operations and AI decision-making processes. Users are informed about how their data is used, and the system's algorithms are designed to provide explanations for generated outputs. 4. User Consent: Users are required to provide informed consent before their data is processed. Clear and concise information about data usage, retention policies, and user rights is provided to ensure users are fully aware of how their information is handled. 5. Safety in Usage: The system includes safeguards to prevent misuse and ensure safe interaction. Features such as content filtering and moderation are employed to avoid the dissemination of harmful or inappropriate content.

Main Supervisor Signature

……..

ACKNOWLEDGMENT

We would like to express our heartfelt gratitude to the individuals and institutions who have supported and guided us throughout this project.

First and foremost, we extend our sincere thanks to Dr. Ahmed El Shaer for his invaluable mentorship, continuous encouragement, and insightful guidance, which have been instrumental in the successful completion of this project.

We also wish to thank Eng. Nagy Khaiet and the College of AI for providing the resources and support necessary to bring this project to fruition.

Special thanks to Eng. Ahmed Metwalli for his technical expertise and assistance, which have greatly contributed to the development and implementation of this project.

Lastly, we would like to acknowledge our families and friends for their unwavering support and understanding during the course of this project. Their encouragement has been a constant source of motivation for us.

Thank you all for your contributions and support.

ABSTRACT

In the digital age, the transformation of educational tools and resources is crucial to enhancing learning experiences and accessibility. This project, "IntelliLearn," leverages advanced AI technologies, including OpenAI's Whisper and xAI's Grok, to develop a comprehensive system for text analysis, summarization, and generation. The system allows users to upload a PDF, audio recording, or video. If the uploaded file is an audio recording or video, it undergoes speech-to-text (STT) processing to extract the text, which is then fed into a large language model (LLM). For PDFs, the text is directly processed by the LLM.

The extracted or uploaded text is then utilized for various tasks such as text generation, code generation, summarization, and question answering. By integrating robust speech recognition capabilities with deep learning insights, IntelliLearn provides efficient content processing and user-friendly interaction, making it an invaluable tool for students, educators, and professionals. The system aims to bridge the gap between vast information sources and users, facilitating better understanding and retention of complex information. This project not only enhances the accessibility of educational content but also streamlines the process of information extraction and utilization, thereby significantly contributing to the field of educational technology.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

Chapter One

1 INTRODUCTION

In an era characterized by the relentless pace of digital progress and the ever-expanding frontiers of technological innovation, the transformative potential of large language models (LLMs) in the realm of education is steadily coming into focus. As digital tools and platforms permeate every facet of our lives, from communication to commerce, it's becoming abundantly clear that LLMs hold the key to a profound revolution in how we approach teaching and learning.

Importance and Problem Statement

The landscape of education grapples with several challenges, including insufficient learning material, limited interaction engagement, and time-consuming manual review processes. Students often find themselves stuck on complex concepts, relying on static resources, and facing inefficiencies in exam preparation. Intellillearn addresses these issues head-on by leveraging the power of LLMs, aiming to enhance accessibility, comprehension, and overall engagement.

In the ever-evolving educational arena, numerous hurdles persist, spanning from a scarcity of learning materials to the struggle to foster meaningful interactions and the labor-intensive nature of manual review procedures. Students frequently encounter difficulties grasping intricate concepts, often resorting to unchanging resources and experiencing inefficiencies in their exam preparations. Intellillearn emerges as a beacon of innovation, boldly confronting these obstacles by harnessing the transformative potential of LLMs, with the primary objective of significantly augmenting accessibility, deepening comprehension, and fostering a heightened level of engagement across educational platforms.

Overview and Statistics

Statistics reveal a significant shift in user behavior following the launch of innovative technologies like ChatGPT. With over 100 million weekly active users, chatbots have proven their efficacy in information retrieval. However, despite their adeptness in swift problem-solving, their full potential in education remains untapped. In the aftermath of ChatGPT's debut in 2020, insights from the education sector underscored the positive impact of employing chatbots for student assessments. This innovative approach not only fostered increased engagement but also contributed to a notable improvement in students' familiarity and competence with the platform.

The paper draws insights from the positive impact of chatbots on student assessments, highlighting the need for a broader application in addressing diverse educational needs. The statistical results presented in Fig 1 demonstrate the growing reliance on digital platforms and

chatbots, reinforcing the importance of integrating advanced AI technologies in educational settings.

Objectives

The objectives of Intellillearn encompass a multifaceted approach designed to enhance accessibility and comprehension through the implementation of a user-friendly chatbot interface. Intellillearn strives to provide accurate answers to queries, generate relevant questions to stimulate learning, offer concise summaries for effective retention, and incorporate visual aids to facilitate understanding. By addressing the complexities and scope of educational content, Intellillearn endeavors to create an inclusive and engaging learning environment that empowers learners of all backgrounds.

Contribution

This paper contributes to the educational landscape by introducing a sophisticated approach to content processing and accessibility tools. By delving into the integration of LLMs, it strives to improve understanding and engagement with digital content, overcoming the limitations faced by traditional educational chatbots.

In the rapidly evolving landscape of artificial intelligence (AI) and machine learning (ML), natural language processing (NLP) has emerged as a crucial field, driving advancements in how machines understand, interpret, and generate human language. Among the myriad applications of NLP, text generation and summarization stand out for their potential to revolutionize information processing, content creation, and human-computer interaction.

Statistics and Results

Overview of User Engagement Post-ChatGPT Launch:

- ChatGPT has garnered over 100 million weekly active users since its debut in 2020.
- A significant shift in user behavior has been observed, with many opting for chatbots over traditional search engines for information retrieval.

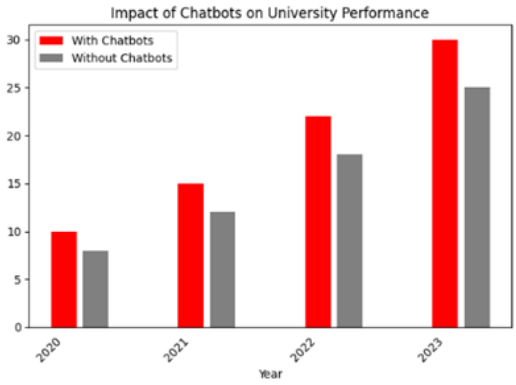
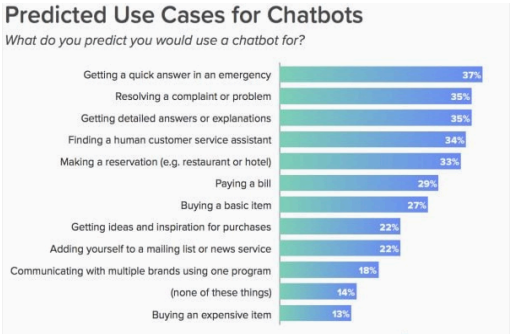


Figure 1 about impact chatbot on university performance

Predicted Use Cases of Chatbots:



- Despite the adeptness of chatbots in swiftly addressing urgent queries during emergencies, their utilization in education remains constrained.
- The prevalent emphasis on immediate problem-solving hinders their potential effectiveness in addressing a broader spectrum of educational needs and contexts.

Figure 2 about predicted uses case for chatbot

Impact on Student Assessments:

- Increased engagement: Students using chatbots for assessments showed higher engagement levels.
- Improved competency: The familiarity and competence of students with digital platforms saw notable improvements.

User Behavior Trends:

- Statistics indicate a growing reliance on chatbots for quick and accurate information retrieval.
- Despite their effectiveness in problem-solving, the full potential of chatbots in educational contexts remains underexplored.

Evolution of Text Generation and Summarization

Text generation, often synonymous with natural language generation (NLG), refers to the process by which machines produce human-like text based on a given input. Historically, text generation systems have evolved from simple rule-based models to sophisticated neural network-based approaches. The advent of statistical methods marked a significant milestone, with n-gram models leading the charge by analyzing the probabilities of word sequences. However, these models were limited by their inability to capture long-range dependencies in text. The introduction of recurrent neural networks (RNNs) and their variants, such as long short-term

memory (LSTM) networks, brought about a significant leap, allowing for better handling of sequential data and more coherent text generation.

In recent years, transformer-based models, most notably BERT and GPT, have set new benchmarks in the field. These models leverage the attention mechanism, which enables them to weigh the importance of different words in a sequence more effectively, resulting in more contextually accurate and fluent text generation.

Core Components of Text Generation Systems

Modern text generation systems comprise several key components:

- **Language Model**: This statistically analyzes large text corpora to predict the most likely next word in a sequence. Transformer models like GPT have revolutionized this component by capturing complex patterns in text.
- **Knowledge Base**: Some systems incorporate a knowledge base, serving as a repository of factual information to ensure coherence and factual accuracy.
- **Task-Specific Encoder**: For specialized tasks, an encoder is used to extract relevant information, tailoring the generated text to the specific task or domain.
- **Decoder**: This translates processed information into human-readable text, ensuring fluency and coherence.

Evaluation Metrics for Text Generation

Evaluating text generation quality remains a challenge. Metrics such as BLEU Score, Human Evaluation, and STS help assess fluency, coherence, and semantic similarity.

Automatic Text Summarization

Automatic text summarization condenses documents while retaining essential information. This task has seen advancements since the late 1950s, aiming to produce concise versions that encapsulate relevant information. Challenges include redundancy and co-reference, especially when summarizing multiple documents.

Speech-to-Text and Summarization

Speech-to-text technology converts spoken language into written text, enabling automatic summarization. By leveraging speech recognition and summarization algorithms, spoken content

can be summarized into concise, readable formats, enhancing accessibility and information retrieval.

By addressing these facets, Intellillearn aims to enhance the educational experience through advanced AI and NLP technologies, making learning more accessible, engaging, and effective.

Chapter Two

2 LITERATURE REVIEW AND RELATED WORK

This chapter reviews existing literature and previous work related to AI in education, text analysis, summarization, and generation. It examines various approaches and technologies that have been utilized in the past, highlighting their strengths and limitations. By critically analyzing these works, this chapter establishes the knowledge base on which IntelliLearn builds and identifies gaps that the project aims to fill.

Text Generation and Summarization

Evolution of Text Generation: Text generation systems have evolved significantly from rule-based models to neural network-based approaches. Early systems, such as n-gram models, were limited in capturing long-range dependencies in text. The advent of Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks marked significant improvements. The recent introduction of transformer-based models, such as BERT and GPT, has set new benchmarks in the field.

Core Components of Modern Systems:

- **Language Model:** Uses statistical analysis to predict the next word in a sequence. Transformer models like GPT are particularly effective.
- **Knowledge Base:** Ensures coherence and factual accuracy.
- **Task-Specific Encoder and Decoder:** Tailors the generated text to specific tasks or domains.

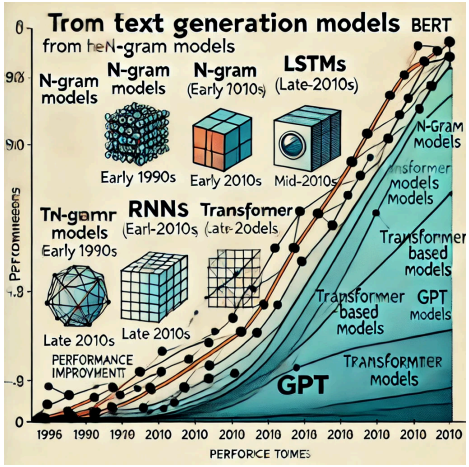


Figure 3 :

This figure illustrates the progression of text generation models from n-gram models to transformers, highlighting the improvements in performance over time.

Summarization Techniques:

- **Extractive Summarization:** Selects key sentences from the original text.
- **Abstractive Summarization:** Generates new sentences, paraphrasing the original content.
- **Hybrid Approaches:** Combine both methods for improved results.

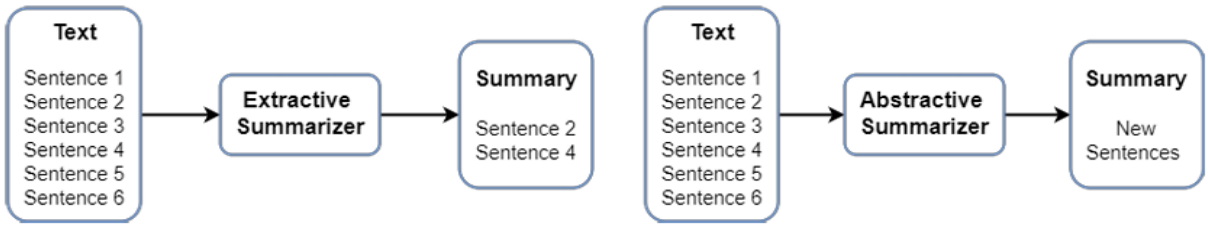


Figure 4 :This Figure Demonstrate difference techniques of summarization

Speech-to-Text Technologies

Modern STT Systems: Systems like Whisper and PyAnnote leverage deep learning for accurate transcription. These systems are crucial for lecture transcription and summarization in educational settings.

Workflow:

1. **Capture:** Recording the lecture audio.
2. **Transcription:** Converting audio to text using models like Whisper.
3. **Pre-processing:** Cleaning and summarizing the transcribed text for clarity.

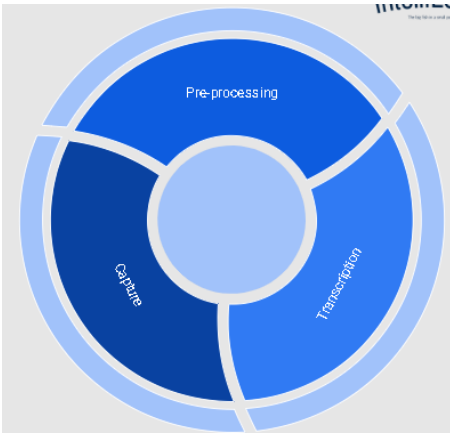


Figure 5 : Show preprocessing stages of speech to text

Text Similarity and Embedding

Text Similarity Measures: Text similarity measures are crucial for various NLP tasks such as information retrieval, document clustering, and text summarization. These measures can be categorized into string-based, corpus-based, and knowledge-based approaches.

- **String-Based Similarity:** Measures such as Levenshtein distance and Jaccard similarity evaluate the similarity based on the character or word sequences in the text.
- **Corpus-Based Similarity:** Methods like Latent Semantic Analysis (LSA) and Pointwise Mutual Information (PMI) use large text corpora to determine the semantic similarity between words.
- **Knowledge-Based Similarity:** Techniques leveraging structured semantic networks, such as WordNet, evaluate the similarity between words based on their meanings.

Embedding Techniques:

- **Word2Vec and GloVe:** Capture semantic relationships between words by representing them as vectors in a continuous vector space.
- **BERT:** Generates context-aware embeddings, improving tasks like named entity recognition, word sense disambiguation, and coreference resolution.

Automatic Speech Recognition (ASR) in Education

Enhancements in ASR Technology: Modern ASR systems, such as Whisper, offer high accuracy and multilingual capabilities, making them suitable for educational applications. These systems can transcribe lectures and convert spoken content into text, aiding accessibility and learning.

Applications in Education:

- **Accessibility:** Enhances educational experiences by enabling accurate transcription of lectures and notes, promoting accessibility for students with disabilities.
- **Focus on Understanding:** Allows students to focus on understanding rather than note-taking.

Educational Chatbots

Capabilities and Limitations: Educational chatbots provide automated responses to student queries, enhancing engagement and accessibility. However, traditional chatbots often rely on rule-based systems and limited datasets, restricting their effectiveness in handling complex queries.

Advancements with LLMs: Leveraging large language models (LLMs) like GPT can significantly enhance the capabilities of educational chatbots by providing more comprehensive and nuanced responses.

Background

Speech-to-Text (STT):

Also known as Automatic Speech Recognition (ASR), STT has seen significant advancements driven by research in deep learning and neural network architectures. Early approaches, rooted in Hidden Markov Models (HMMs), have transitioned to more sophisticated techniques such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and attention mechanisms. These advancements have enabled the development of end-to-end ASR models capable of directly mapping speech signals to text without relying on intermediate representations. Additionally, transfer learning strategies and multimodal approaches have been explored to improve performance, particularly in low-resource settings or noisy environments. Ongoing research efforts focus on enhancing the robustness, accuracy, and adaptability of ASR systems, addressing challenges such as variability in speech patterns, accents, and background noise.

Automatic Text Summarization:

This field seeks to condense large amounts of text into shorter, more coherent summaries while retaining vital information and meaning. Traditional approaches include extractive methods, which detect and extract relevant sentences or phrases from the source text based on characteristics such as sentence importance or centrality. More recently, abstractive summarization techniques have gained popularity, where the system generates summaries by paraphrasing and rephrasing information in a new form, frequently employing neural network topologies such as transformers. These models learn to understand the input text's context and semantics, allowing them to provide more fluent and human-like summaries. Text summarizing research also includes domain-specific summarization, multi-document summarization, and measures for objectively evaluating the quality of generated summaries. Future research areas will focus on enhancing the coherence, informativeness, and efficiency of summarizing systems, as well as tackling issues such as dealing with various forms of content, including multimedia sources, and ensuring that summaries are generated ethically and impartially.

Embedding Techniques in NLP:

These represent words as vectors in a continuous space, capturing semantic and syntactic similarities, aiding NLP models in understanding language. Traditional methods like Word2Vec and GloVe learn embeddings from large text corpora using shallow neural networks or matrix factorization. However, they lack context-awareness, limiting their ability to handle word ambiguity. Contextualized word embeddings, powered by transformer-based models like BERT, address this limitation by considering the surrounding context when generating word embeddings, resulting in more nuanced representations that capture fine-grained semantic relationships. This allows for better handling of ambiguity and improved performance in tasks like named entity recognition, word sense disambiguation, and co-reference resolution. These contextual embeddings are derived from pre-trained Language Models (LMs) trained on massive text datasets, learning general language features by predicting words based on their context. Specific architectures for generating contextual embeddings will be delved into in the next section.

Large Language Models (LLMs):

These models represent the apex of natural language processing research, employing transformer topologies to achieve unparalleled levels of comprehension and human-like text production. These models, pre-trained on massive volumes of text data, excel in various NLP tasks such as language translation, text summarization, text generation, code generation, sentiment analysis, and question answering. The transformer architecture's self-attention mechanism enables LLMs to detect nuanced language patterns and long-range connections inside text, allowing them to provide coherent and contextually relevant replies. Furthermore, LLMs have shown encouraging performance in few-shot and zero-shot learning circumstances, allowing them to complete tasks with little or no task-specific training data.

Text Generation:

This is a key problem in natural language processing (NLP) that involves producing coherent and contextually relevant text based on input prompts or conditioning information. This field has made great progress, particularly with the introduction of large-scale language models (LLMs) based on transformer architectures. These models, trained on huge volumes of text data, show extraordinary ability to generate human-like writing in a variety of fields, including language translation, conversation generation, summarization, and story generation. Recent research efforts have focused on improving the fluency, coherence, and diversity of generated text, addressing issues such as maintaining context over long sequences, decreasing repeating patterns, and managing the generated content's style or sentiment. Techniques such as fine-tuning, diversity-promoting objectives, and controlled generation have been investigated to improve the quality and adaptability of text generation systems.

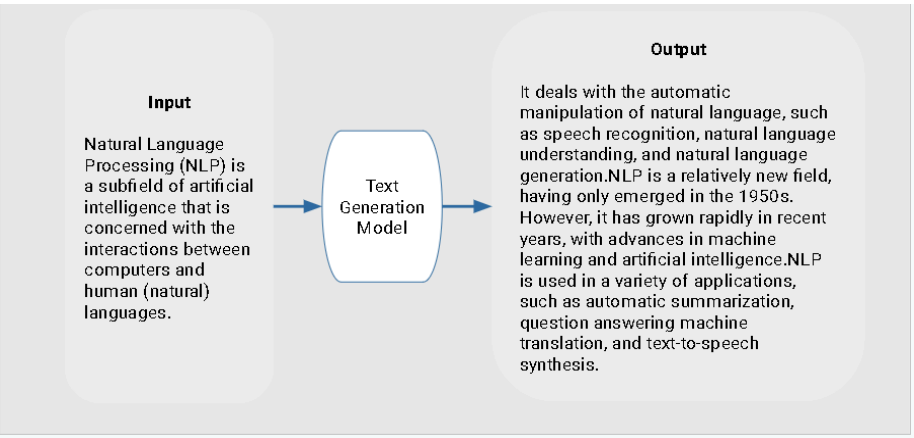


Figure 6: This figure showcase example of text generation

Code Generation with LLMs:

This is a cutting-edge approach in the field of automated software development. Using the advantages of LLMs, particularly transformer-based designs, researchers have explored innovative approaches for autonomously producing code from natural language specifications or high-level programming tasks. These models, pre-trained on large volumes of text data, show extraordinary proficiency in interpreting and generating human-like text, making them ideal for tasks such as code production. Researchers have obtained outstanding syntactic and semantic code creation results by fine-tuning LLMs on code generation tasks and providing suitable context or hints. Furthermore, approaches like prompt engineering, conditional generation, and multi-task learning have been used to increase the quality and efficiency of code generation with LLMs.

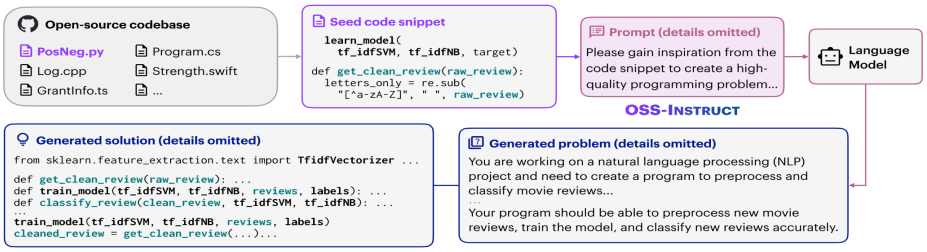


Figure 7 showcases a code generation sample, illustrating the effectiveness of these advanced techniques in producing accurate and relevant code snippets.

Related Work

Building on Existing Advancements: While Intellillearn leverages Large Language Models (LLMs) to create a transformative learning experience, it's valuable to acknowledge existing efforts in educational technology.

Chatbots and Educational Technology:

- **Educational Chatbots:** Chatbots have emerged as a tool for educational interaction. Investigating existing chatbots reveals their capabilities in addressing student needs, such as question answering or content delivery. However, these chatbots often rely on rule-based systems or limited datasets, leading to limitations in handling complex queries or adapting to individual learners.
- **Intellillearn's Advantage:** Intellillearn's LLM-powered chatbot interface surpasses these limitations. By leveraging the power of LLMs, Intellillearn can provide more comprehensive and nuanced responses, fostering a more natural and engaging learning experience.

Automatic Speech Recognition (ASR) for Education:

- **Leveraging Speech for Enhanced Learning:** Automatic Speech Recognition (ASR) technology presents exciting possibilities for improving educational experiences. Studies have explored its

application in online learning environments, allowing students to dictate notes or transcribe lectures. These functionalities can be particularly beneficial for:

- **Accessibility:** Students with disabilities can leverage ASR for dictation, promoting active participation and independent learning.
- **Note-Taking:** Transcribing lectures enables students to focus on understanding concepts rather than frantically writing everything down.
- **Review and Comprehension:** Transcripts allow students to revisit lectures at their own pace, clarifying missed points or reinforcing key concepts.

However, current ASR systems face challenges, particularly with:

- **Accuracy:** Accents, background noise, and complex terminology can lead to errors in transcriptions, potentially hindering comprehension.
- **Integration:** Seamless integration of ASR into learning platforms is crucial for user adoption and efficient workflow.

Intellillearn's Opportunity: Intellillearn can capitalize on the advantages of ASR while addressing its limitations.

- **Enhanced Accessibility:** Intellillearn can integrate ASR to offer voice-based interaction, further promoting inclusivity for students with disabilities.
- **Improved Transcription Accuracy:** By potentially leveraging domain-specific language models or integrating with advanced ASR systems, Intellillearn can strive for higher accuracy in transcribing educational content, including lectures and discussions.
- **Seamless Integration:** Intellillearn's design should ensure that ASR functionalities are seamlessly integrated, allowing for smooth switching between speech and text input and efficient transcript management.

Text Summarization for Learning:

- **Educational Summarization Tools:** Existing tools summarize educational content, focusing on specific domains or general knowledge. While these tools can be helpful, summaries may lack nuance or miss key details.

- **Intellillearn's Distinction:** Intellillearn's summarization capabilities utilize LLMs, enabling them to create more comprehensive and tailored summaries that consider context and individual learning needs. This approach can enhance student comprehension by providing focused information relevant to their progress.

Large Language Models (LLMs) in Education:

- **LLM Applications in Learning Platforms:** Other platforms are exploring LLMs in education, focusing on question answering or personalized feedback. While these projects share similar goals, they may employ different LLM architectures or lack the comprehensive approach of Intellillearn.
- **Intellillearn's Innovation:** Intellillearn differentiates itself by utilizing LLMs for a broader range of functionalities, including question answering, personalized feedback, content adaptation, and potentially voice interaction through ASR integration. This multifaceted approach creates a more holistic and adaptive learning environment.

Educational Software with Similar Goals:

- **Educational Software:** Products like Typeset.io or Study Fetch focus on creating or curating educational materials. While these tools provide valuable resources, they often lack the interactive and personalized learning experience offered by Intellillearn.
- **Intellillearn's Strength:** Intellillearn leverages LLMs to create a dynamic and personalized learning experience that goes beyond static content delivery. The LLM capabilities enable Intellillearn to adapt to individual needs, provide real-time feedback, and create a more engaging learning environment.

Existing Systems

- **Typeset.io:** Focuses on academic research and writing, offering tools for citation generation and paper summarization but lacks audio-to-text conversion capabilities.
- **Study Fetch:** Provides interactive study tools and AI-powered flashcards but does not specialize in lecture summarization or advanced text generation.
- **Copilot:** Excels in code generation and completion but does not address summarization or question answering for general text or lectures.

Innovative Approaches

- **Chat-PDF:** Integrates multiple functionalities, including text summarization and question answering, enhancing accessibility and interaction with diverse content types.
- **IntelliLearn:** Leverages LLMs to provide comprehensive educational support through features like text summarization, question generation, and multimedia support, demonstrating the potential of integrating advanced AI technologies in education.

Table 1

This Table demonstrates educational software comparisons that are similar to our idea but different in features that summarize that successful Intellillearn in many features and availability first.

	Text Summarization	Speech To Text	Text Generation	Code Generation
Typeset.io	Available	Not Available	Available	Not Available
Study Fetch	Available	Not Available	Available	Not Available
Copilot	Not Available	Not Available	Not Available	Available
Chat-PDF	Available	Not Available	Available	Not Available
IntelliLearn	Available	Available	Available	Available

Chapter three

3 PROPOSED TERMINOLOGY

This chapter defines the key terminology used throughout the IntelliLearn project. Understanding these terms is crucial for comprehending the system's functionality and the technological concepts it employs. The chapter covers essential concepts such as Speech-to-Text (STT), Text Summarization, Text Generation, Large Language Models (LLMs), and more, providing a foundation for the subsequent technical discussions .

1. **Speech-to-Text (STT)**: Technology that converts spoken language into written text. This includes various models like Whisper and Wav2vec2 that transcribe audio inputs into textual data, which can then be processed for further analysis and summarization.
2. **Text Summarization**: The process of distilling the most important information from a source text. This can be achieved through extractive methods (selecting key sentences) or abstractive methods (generating new sentences that convey the main points).
3. **Text Generation**: AI models generating human-like text based on given inputs. This includes generating responses in chatbots, completing sentences, or even writing essays.
4. **Large Language Models (LLMs)**: Advanced AI models like GPT-3 and BERT that can understand and generate text by analyzing vast amounts of data and learning complex language patterns.

5. **Text Embedding**: Techniques used to convert text into numerical vectors that capture semantic meaning and contextual relationships, enabling better text analysis and processing.
6. **Question Generation**: The process by which AI models create questions based on given text, aiding in self-assessment and study.
7. **Speaker Diarization**: The process of segmenting audio recordings by speaker labels, improving transcript readability and facilitating the summarization process.
8. **Natural Language Processing (NLP)**: A field of AI that focuses on the interaction between computers and humans through natural language, encompassing tasks like speech recognition, text generation, and sentiment analysis.
9. **TF-IDF (Term Frequency-Inverse Document Frequency)**: A statistical measure used to evaluate the importance of a word in a document relative to a corpus, commonly used in text mining and information retrieval.
10. **Word2Vec**: A group of related models used to produce word embeddings, capturing semantic relationships between words in a continuous vector space.
11. **BERT (Bidirectional Encoder Representations from Transformers)**: A pre-trained language model developed by Google that improves various NLP tasks by considering the context from both directions.
12. **Abstractive Summarization**: A type of text summarization that generates new sentences to capture the main ideas, rather than selecting sentences from the source text.
13. **Extractive Summarization**: A summarization technique that involves selecting and combining existing sentences from the original text to create a summary.
14. **Contextual Word Embeddings**: Representations of words that capture their meanings in different contexts, enabling better understanding and processing of language by AI models.
15. **Transformer Architecture**: A neural network design that uses self-attention mechanisms to improve the processing of sequential data, leading to advancements in NLP tasks such as text generation and translation.
16. **Natural Language Understanding (NLU)**: The ability of a machine to comprehend and interpret human language, focusing on the meaning behind the words.
17. **Multimodal Learning**: An approach that integrates multiple types of data (e.g., text, audio, images) to improve the performance and robustness of AI models.
18. **Named Entity Recognition (NER)**: A subtask of information extraction that identifies and classifies entities in text into predefined categories such as names of persons, organizations, locations, etc.
19. **Neural Machine Translation (NMT)**: The use of neural network models to translate text from one language to another, enhancing the accuracy and fluency of translations.
20. **Sequence-to-Sequence (Seq2Seq) Model**: A type of neural network architecture used for tasks like machine translation and text summarization, where the input and output are sequences of text.

21. **Attention Mechanism**: A component of neural networks that allows the model to focus on specific parts of the input sequence, improving the handling of long-range dependencies and context.
22. **Recurrent Neural Networks (RNNs)**: A type of neural network designed for processing sequential data, with connections that form directed cycles to create an internal state for dynamic temporal behavior.
23. **Long Short-Term Memory (LSTM)**: An extension of RNNs that addresses the vanishing gradient problem, allowing the network to remember important information over long periods.
24. **Convolutional Neural Networks (CNNs)**: A type of deep learning model primarily used for image processing but also applied to NLP tasks for capturing local dependencies in text.
25. **Transfer Learning**: A technique where a pre-trained model is fine-tuned on a different but related task, leveraging prior knowledge to improve performance on the new task.
26. **Few-Shot Learning**: The ability of an AI model to learn new tasks from a small amount of data, demonstrating flexibility and adaptability in handling diverse scenarios.
27. **Zero-Shot Learning**: The capability of an AI model to perform tasks it was not explicitly trained on, using general knowledge acquired during training on other tasks.
28. **Prompt Engineering**: The practice of designing and optimizing input prompts to elicit desired responses from large language models, enhancing their utility for specific applications.
29. **Conditional Text Generation**: A method where text generation is guided by specific conditions or constraints, producing output that adheres to predefined criteria.
30. **Ethical AI**: The study and application of ethical principles to the development and deployment of AI systems, ensuring fairness, transparency, and accountability.

Chapter Four

4 PROPOSED MODEL

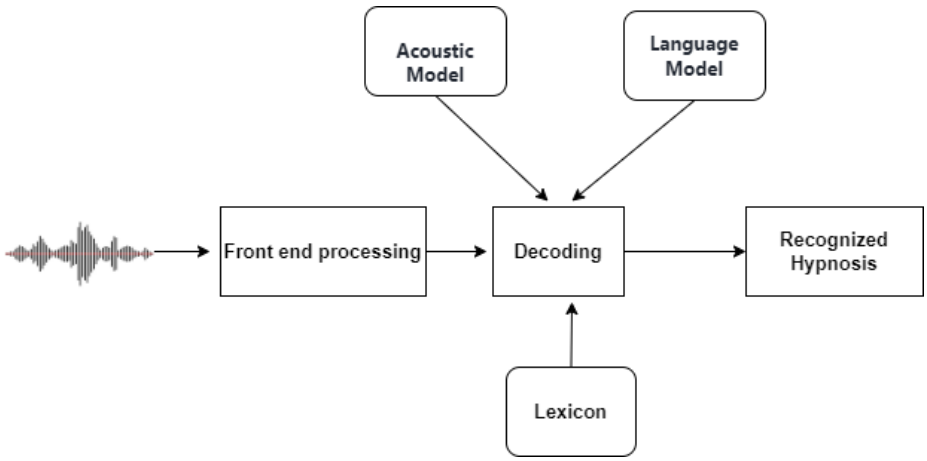
This chapter outlines the proposed model for IntelliLearn, detailing its components and how they work together to achieve the project's objectives. It describes the architecture of the system, including the integration of OpenAI's Whisper and xAI's Grok, and explains how each part contributes to text analysis, summarization, and generation. The chapter also discusses the design considerations and technical choices made during the development of the model .

Speech To Text (STT)

Speech-to-text (STT) technology is a software program that converts spoken words into written text. It is often referred to as speech recognition or computer speech recognition [11]. The importance of technology has grown across a number of uses, such as text transcription of lectures.

Component of ASR :

- Feature Extraction : It converts the speech signal into a sequence of acoustic feature vectors. These observations should be compact and carry sufficient information for recognition in the later stage.
- Acoustic Model : It contains a statistical representation of the distinct sounds that make up each word in the Language Model or Grammar. Each distinct sound corresponds to a phoneme.
- Language Model: it contains a very large list of words and their probability of occurrence in a given sequence.
- Lexicon : The lexicon is a large dictionary that contains all of the words that the speech recognition system is able to recognize.
- Decode r: It is a software program that takes the sounds spoken by a user and searches the acoustic model for the equivalent sounds. When a match is made, the decoder determines the phoneme corresponding to the sound. It keeps track of the matching phonemes until it reaches a pause in the users speech. It then searches the language model for the equivalent series of phonemes. If a match is made, it returns the text of the corresponding word or phrase to the calling program. Figure 8 :Architecture of an ASR System - As Shown



First model: Whisper Speech to Text (STT) Model

The Whisper Speech to Text (STT) model, developed by OpenAI, represents a groundbreaking advancement in the realm of automatic speech recognition (ASR). It is meticulously engineered to address the challenges posed by various acoustic environments, particularly excelling in low-volume settings where traditional STT models struggle to maintain accuracy.

Key Features:

- **Low-Volume Accuracy:** Whisper STT is optimized to accurately transcribe speech in environments with minimal background noise, ensuring clarity and precision even when spoken at lower volumes.
- **Robust Performance:** Leveraging a sophisticated encoder-decoder Transformer architecture, Whisper STT exhibits robust performance across diverse dialects, accents, and speech patterns. This versatility enhances transcription accuracy in multilingual and varied linguistic contexts.
- **Adaptive Learning:** The model incorporates deep learning principles and neural network architectures, enabling continuous improvement through iterative training on extensive and diverse speech datasets. This adaptive learning approach enhances its ability to handle complex speech variations effectively.

- **Real-time Processing:** Designed for real-time applications, Whisper STT minimizes processing latency without compromising on accuracy. This capability makes it suitable for applications requiring immediate transcription feedback, such as live broadcasts and interactive voice interfaces.
- **Privacy Considerations:** Whisper STT prioritizes user privacy by offering options for processing speech data locally or utilizing advanced encryption methods for cloud-based processing. This ensures compliance with stringent data protection regulations and addresses concerns regarding data security.

Technical Specifications:

- **Architecture:** Whisper STT employs a hybrid architecture combining recurrent neural networks (RNNs) with attention mechanisms within its Transformer framework. This architecture optimizes both contextual understanding and computational efficiency, supporting high-performance speech recognition across diverse scenarios.
- **Training Data:** Trained on a diverse corpus of annotated speech datasets, Whisper STT encompasses both proprietary and publicly available sources. This training strategy enhances its ability to recognize and transcribe speech accurately across varying acoustic conditions and linguistic contexts.
- **Deployment:** The model is designed for versatile deployment, capable of integration into cloud infrastructure, edge devices, and software applications via robust APIs. This flexibility ensures accessibility across different platforms and environments, facilitating seamless integration into existing technological ecosystems.

Applications:

- **Healthcare:** Whisper STT facilitates precise transcription of whispered medical consultations and diagnostics in quiet clinical settings, ensuring accurate documentation and enhancing healthcare delivery.
- **Security:** In security applications, Whisper STT enhances surveillance systems by accurately transcribing whispered conversations in sensitive environments with minimal audio disturbance, aiding in monitoring and analysis.
- **Broadcasting:** For broadcasters, Whisper STT enables clear and precise transcription of whispered commentary during live events or broadcasts, enhancing accessibility and comprehension for audiences.

Diarization and Its Role in STT Systems

What is Diarization?

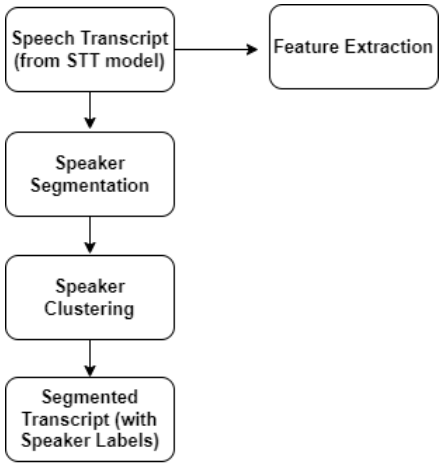
Speaker diarization complements STT systems like Whisper by segmenting audio streams into distinct speaker segments and attributing them to individual speakers. This process enhances the usability and comprehension of transcribed content by providing context about "who spoke when?"

Benefits and Applications of Diarization

- **Improved Readability:** By organizing audio segments according to speakers, diarization produces structured and readable transcripts, particularly beneficial in scenarios involving multiple speakers or overlapping dialogue.
- **Efficient Indexing:** Diarization enables efficient indexing and retrieval of specific audio segments by speaker, enhancing searchability and usability in archival and data mining applications.
- **Contextual Understanding:** It enriches the transcribed text with speaker-specific information, improving comprehension and facilitating deeper analysis in applications such as lectures, meetings, and customer interactions.

Speaker Segmentation and Clustering in Diarization

Incorporating Whisper STT with diarization tools like PyAnnote involves a detailed process:



1. **Speech Transcript Generation:** Whisper STT transcribes audio recordings into text, forming the initial transcript for further processing.
2. **Feature Extraction:** Tools like PyAnnote extract acoustic features directly from the original audio, capturing unique vocal characteristics such as pitch, timbre, and speaking style.
3. **Speaker Segmentation:** Techniques including silence detection, voice activity detection (VAD), and energy-based change point detection are applied to identify speaker change points within the audio stream.
4. **Speaker Clustering:** Machine learning algorithms within PyAnnote cluster audio segments based on extracted features, grouping segments with similar acoustic profiles into coherent speaker clusters.

5. **Segmented Transcript:** The final output is a segmented transcript where each segment is annotated with the corresponding speaker, providing a structured and informative representation of the original audio content.

Figure 9 Show about Speaker Segmentation

STS model

Semantic Text Similarity (STS) model that combines corpus-based and knowledge-based methods to measure the similarity between texts. It incorporates string similarity, semantic word similarity, and optional common-word order similarity functions to provide a comprehensive measure of text similarity. The STS model is evaluated in tasks like paraphrase identification, demonstrating competitive accuracy and precision while offering lower time complexity compared to existing approaches.

Text Similarity

Measuring the similarity between words, sentences, paragraphs and documents is an important component in various tasks such as information retrieval, document clustering, word-sense disambiguation, automatic essay scoring, short answer grading, machine translation and text summarization[50].

Main Categories of Text Similarity Approaches:

- **String-Based similarity** methods focus on analyzing string sequences and character composition to measure similarity. The review discusses fourteen algorithms within this category, delineating between character-based and term-based distance measures. This classification provides a foundational understanding of how similarity can be computed based on the structural composition of text[50].
- **Corpus-Based similarity** measures delve into semantic similarity, utilizing information derived from large corpora to determine the similarity between words. The document elaborates on nine algorithms falling under this category, including HAL, LSA, GLSA, ESA, CL-ESA, PMI-IR, SCO-PMI, NGD, and DISCO. Each algorithm offers a unique approach to quantifying semantic similarity, contributing to a nuanced understanding of text similarity assessment[50].
- **Knowledge-Based similarity** methods rely on semantic networks, such as WordNet, to identify similarities between words based on their conceptual relationships. The review introduces nine algorithms in this category, six focusing on semantic similarity (res, lin, jcn, lch, wup, and path), and three focusing on semantic relatedness (hso, lesk, and vector). This classification underscores the importance of leveraging structured knowledge representations to augment text similarity analysis[50].
- **Hybrid** methods use multiple similarity measures it leverage the strengths of different approaches to achieve more robust and accurate similarity scores.

Text Embedding

Why Text Embedding ?

Extracting meaning from human language has long been a challenge for computers. Word embeddings, a powerful technique in Natural Language Processing (NLP), used to map words from vocabulary to vectors of real numbers. This mapping causes the words that emerge from a similar context that can be correlated with each other.

The free text words of the vocabulary are converted into numeric values (vectors). This transformation is necessary as many Machine Learning models can understand only the vector representation.One of the elementary transformation approaches is one-hot encoding, where each word determines one dimension and the binary value (1 or 0) indicates the presence or absence of the word. One-hot encoding is computationally impossible when dealing with the entire dictionary, as the representation demands thousands of dimensions. Besides, this encoding has no hidden relationships among the words..

Currently, word embeddings are one of the successful applications of unsupervised learning as they do not need pricey annotation. Word embeddings are represented in a way that similar words have similar encoding. Word embeddings are useful in feature generation and Natural Language Processing tasks like text classification, document clustering, and sentiment classification. Word embedding technique can be categorised into Traditional or Frequency word embedding, Static word embedding, and Contextualized word embedding.

Focus now on the review of different word embedding techniques, how they are used to represent the text, capture meanings, similarity among the text, and importance of the text

Word Embedding Techniques

It is found from the literature survey that there exist three major word embedding techniques namely, Traditional word embedding, Static word embedding, and Contextualized word embedding. The following subsections present surveys on these techniques

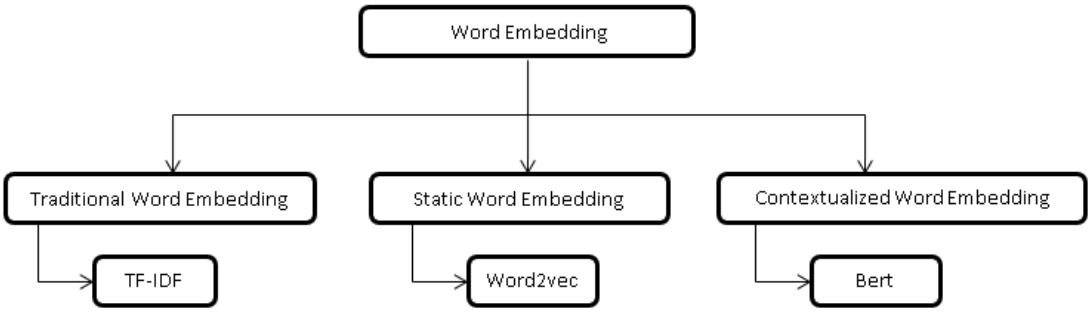


Figure 10 show word embedding technique

- Traditional Word Embedding:

Traditional word embedding is based on the frequency that considers the whole document and discovers the significance of rare words in the document

The first model we tested is the bag of words model with term frequency-inverse document frequency scores. Here the documents are also represented as vectors but instead of a vector of '0's and '1's, now the document contains scores for each of the words. These scores are calculated by multiplying TF and IDF for specific words. So, the score of any word in any document can be represented as per the following equation:

$$TF\text{-}IDF(word, doc) = TF(word, doc) * IDF(word) \quad (1)$$

Equation 1 show about Bag Of Words

Therefore in this method, two matrices have to be calculated, one containing the inverse document frequency of a word in the whole corpus of documents and another containing the term frequency of each word in each document. The formulae to calculate both of them are as follows:

$$TF(word, doc) = (Frequencyofword \in the\ doc\ ) / (No\ ofwords \in the\ doc) \quad (2)$$

$$IDF(word) = log_e [\ (1 + No.\ ofdocs) / (No.\ of\ docs\ with\ word)\ ] \quad (3)$$

Formula calculation of TF-IDF

The simple TF-IDF model works well and gives importance to the uncommon words rather than treating all the words as equal in case of a binary bag of words model. This model however fails to perform accurately when it encounters any sentence containing negations. This negation is a very common linguistic construction that affects word/sentence polarity, also TF-IDF Ignores Semantics: TF-IDF can't grasp the meaning of words. For instance, "computer mouse" and "large rodent" would have similar scores if they appeared frequently together in a specific context, even though they have different meanings, also Focuses on Words, Not Order: TF-IDF treats documents as a "bag of words," meaning the

order of the words doesn't matter. This can be a drawback when word order is crucial for meaning, like in sentences.

Despite these limitations, TF-IDF remains a valuable tool for various tasks. By being aware of its shortcomings, you can effectively use it alongside other techniques when needed.

•Static Word Embedding

Static word embedding is prediction-based that provides probabilities to the words and maps each word into a vector[52]. Static embeddings are learned by training the lookup tables which convert words into dense vectors. This embedding is static in the way that they do not alter the context once been learned and also the embedding tables do not change among different sentences.

Word2vec is a powerful technique for learning word representations, also known as word embeddings. It achieves this by leveraging a shallow neural network architecture trained on a massive corpus of text data. This section details the core components of Word2vec

- **Training on a Large Corpus:**

The foundation of Word2vec lies in its training data. By processing a vast amount of text, Word2vec can capture the inherent relationships and contextual usage of words within the corpus. This allows the model to develop an understanding of semantic similarities between words.

- **Neural Network Architecture:**

Word2vec employs a shallow neural network architecture, typically consisting of one or two hidden layers. This network operates on word sequences, aiming to predict surrounding words based on the current word (Skip-gram) or vice versa (CBOW).

- **Training Methods:**

Word2vec encompasses two primary training methods:

- **Continuous Bag-of-Words Model**

The first proposed architecture is similar to the feedforward NNLM, where the nonlinear hidden layer is removed and the projection layer is shared for all words (not just the projection matrix); thus, all words

get projected into the same position (their vectors are averaged). We call this architecture a bag-of-words model as the order of words in the history does not influence the projection. Furthermore, we also use words from the future; we have obtained the best performance on the task introduced in the next section by building a log-linear classifier with four future and four history

words at the input, where the training criterion is to correctly classify the current (middle) word. Training complexity is then

$$\mathbf{Q = N \times D + D \times \log2(V)(4)}$$

Equation 3 of Continuous Bag-of words

Denote this model further as CBOW, as unlike standard bag-of-words model, it uses continuous distributed representation of the context. The model architecture is shown at Figure 1. Note that the weight matrix between the input and the projection layer is shared for all word positions in the same way as in the NNLM.

- **Continuous Skip-gram Mode**

The second architecture is similar to CBOW, but instead of predicting the current word based on the context, it tries to maximize classification of a word based on another word in the same sentence. More precisely, we use each current word as an input to a log-linear classifier with continuous projection layer, and predict words within a certain range before and after the current word. We found that increasing the range improves the quality of the resulting word vectors, but it also increases the computational complexity. Since the more distant words are usually less related to the current word than those close to it, we give less weight to the distant words by sampling less from those words in our training examples. The training complexity of this architecture is proportional to

$$\mathbf{Q = C \times (D + D \times \log2(V))(5)}$$

Equation 4 Continuous Skip-gram Mode

where C is the maximum distance of the words. Thus, if we choose C = 5, for each training word we will select randomly a number R in range < 1; C >, and then use R words from history and R words from the future of the current word as correct labels. This will require us to do R × 2 word classifications, with the current word as input, and each of the R + R words as output. In the following experiments, we use C = 10

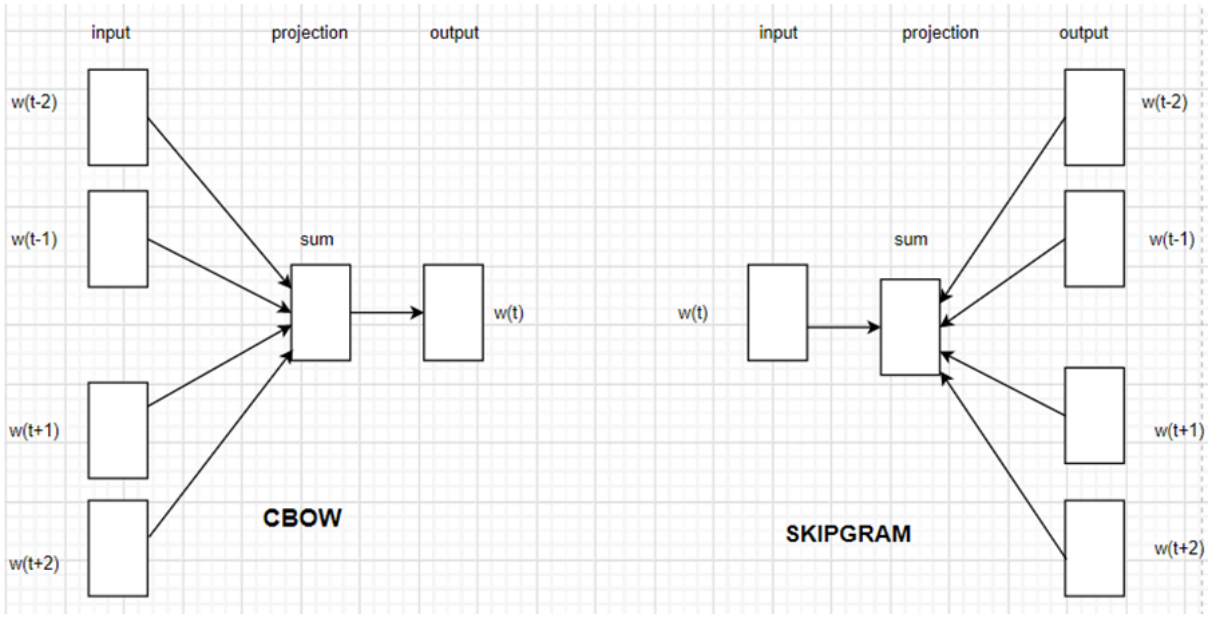


Figure 11 show main difference between CBOW & Skip Gram

- Contextualized Word Embedding

contextualized word embedding is based on the context of a particular word, in which, similar words will have contrast context representations. These representations dynamically vary based upon the context in which a word appears.
BERT (Bidirectional Encoder Representations from Transformers has revolutionized the field of Natural Language Processing (NLP) by offering a powerful pre-trained language model. This section delves into the core concepts behind BERT, including its architecture, pre-training process, fine-tuning capabilities, and its application in text embedding tasks.

Architecture

BERT leverages the Transformer architecture, a powerful neural network design well-suited for sequence-to-sequence tasks like machine translation and text summarization. The core component of the Transformer is the encoder, which consists of multiple layers with an attention mechanism. This mechanism allows the model to attend to different parts of the input sentence simultaneously, capturing long-range and short-range dependencies between words. Unlike the standard Transformer encoder used for tasks like machine translation, BERT modifies the architecture to achieve bidirectionality. This means BERT can consider both the preceding and following words in a sentence when analyzing a particular word, leading to a deeper grasp of context.

- **Pre-training**

BERT's pre-training stage plays a crucial role in its effectiveness[56]. It involves training the model on a massive dataset of text and text pairs using two primary tasks:

Masked Language Modeling (MLM): Random words within the input sentence are masked, and the model predicts the masked words based on the surrounding context. This process helps BERT grasp the meaning and relationships between words.

Next Sentence Prediction (NSP): The model receives two sentences and predicts if the second sentence logically follows the first sentence. This objective trains BERT to understand the relationships between sentences and how they flow together coherently.

Through these pre-training tasks, BERT acquires a vast knowledge base of language usage patterns and semantics. This pre-trained knowledge significantly reduces the amount of training data required for specific NLP tasks compared to training a model from scratch.

- **Fine-tuning**

One of BERT's key strengths is its ability to be fine-tuned for various NLP tasks[55]. While the pre-trained BERT model excels at understanding general language patterns, fine-tuning allows researchers and developers to adapt it to specific tasks. This fine-tuning process involves adding additional output layers on top of the pre-trained encoder. These output layers are trained on labelled data specific to the target NLP task. For instance, to fine-tune BERT for sentiment analysis, the additional layers would be trained on a dataset of text labelled with positive, negative, or neutral sentiment.

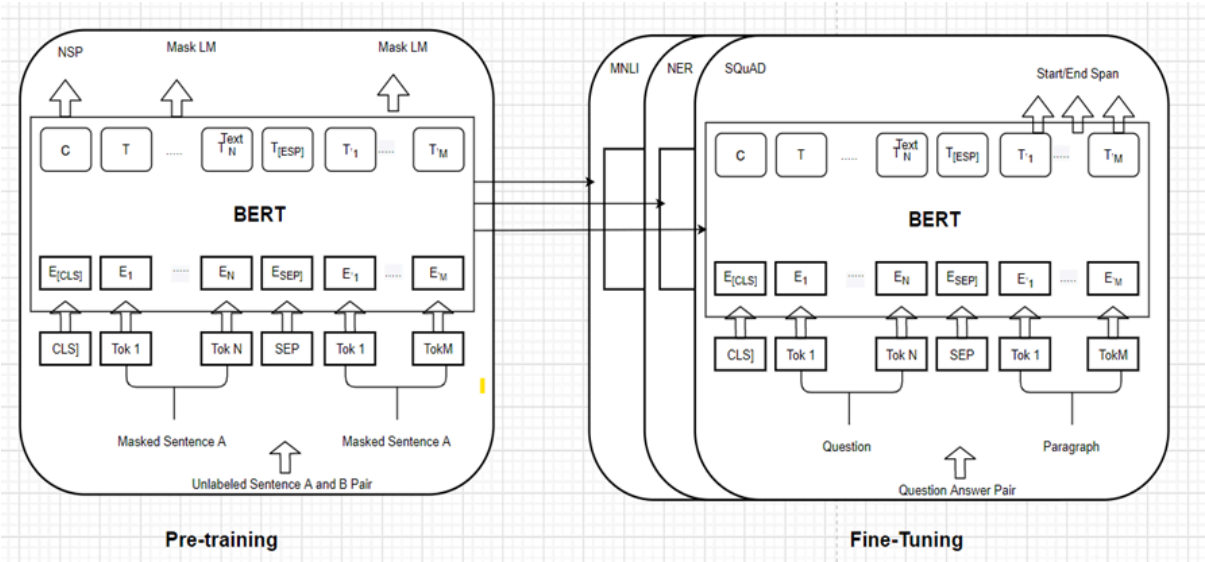


Figure 12 show main difference between pre-training and fine Tuning

BERT Approach

BERT's pre-trained knowledge and contextual understanding make it well-suited for text embedding tasks. Text embeddings are dense vector representations that capture the semantic meaning of a text segment. There are two common approaches to extract text embeddings from BERT:

Using the Last Hidden Layer: This common method extracts the output vector corresponding to each word from the final encoder layer in the pre-trained BERT model. This vector can be considered the word embedding, encapsulating the word's meaning based on the surrounding context in the sentence.

Pooling Strategies: Another approach involves applying pooling functions (e.g., averaging or max-pooling) across all hidden layers for a particular word. This pooling aims to capture a more holistic representation of the word by aggregating information from different layers within the BERT encoder.

Text embeddings generated from BERT offer several advantages compared to traditional methods like TF-IDF:

•Contextual Awareness: BERT considers both preceding and following words, resulting in embeddings that are sensitive to the context in which a word appears.

•Richer Representations: Pre-trained on vast amounts of text, BERT embeddings capture more nuanced semantic relationships between words.

•Task Agnostic: BERT embeddings are generally applicable to various NLP tasks without requiring domain-specific training data.

By leveraging BERT embeddings, NLP applications can achieve superior performance by incorporating rich contextual information and semantic understanding.

Text Generation

Text generation, which is often formally referred to as natural language generation (NLG), is one of the most important yet challenging tasks in natural language processing (NLP). NLG aims at producing understandable text in human language from linguistic or non-linguistic data in a variety of forms such as textual data, numerical data, image data, structured knowledge bases, and knowledge graphs. Among these, text-to-text generation is one of the most important applications and thus often shortly referred to as “text generation”.

Components of Text Generation Systems:

Text generation can be viewed through the lens of its core components:

- **Language Model:** This component plays a crucial role, similar to the STT language model. It statistically analyzes large text corpora to understand how words typically co-occur and sequence. This knowledge is then leveraged to predict the most likely next word in a sequence, enabling the generation of grammatically correct and coherent text. Different language model architectures exist, including n-gram models, recurrent neural networks (RNNs) like LSTMs, and transformers].
- **Knowledge Base:** While not always present, some text generation systems utilize a knowledge base. This component acts as a repository of factual information, similar to a lexicon in STT. The system can access and incorporate this knowledge into generated text, ensuring factual accuracy and potentially enriching the content..
- **Task-Specific Encoder:** In certain text generation tasks, a specific encoder component might be used. This encoder analyzes the input (e.g., a news headline, a product description prompt) and extracts relevant information to guide the generation process. This can help tailor the generated text to the specific task or domain.
- **Decoder:** The decoder component, analogous to the STT decoder, takes the output from the language model and knowledge base (if used) and translates it into human-readable text. This might involve generating word-by-word or using a more complex approach depending on the chosen architecture.

Text Summarization:

Automatic text summarization system generates summary, i.e., condensed form of the document that contains a few important sentences selected from the document. In late fifties , text summarization began and till now, there is great improvement in this field. A large number of techniques and approaches have been developed in this field of research . A summary generated by an automatic text summarizer should consist of the most relevant information in a document and at the same time, it should occupy less space than the original document. Nevertheless, automatic summary generation is a challenging task. There are many issues like redundancy, temporal dimension, co-reference, sentence ordering, etc that need particular attention when summarizing multiple numbers of documents, thereby, making this task more complex .

Generating Summary for Input Document :

The task of compressing a piece of text into a shorter version, minimising the size of the original text while keeping crucial informational aspects and content meaning, is known as summarization. A summary is a reductive transformation of a source text into a summary text by extraction or generation (4). According to another definition, “An automatic summary is a text generated by a software that is coherent and contains a significant amount of relevant information from the source text. Its compression rate τ is less than a third of the length of the original document (5). The ratio between the length of the summary and the length of the source document is calculated by the compression rate τ as shown below:

$$t=\text{Summary}/\text{Source}$$

Equation Number 5 Calculate compression rate

Where | • | indicates the length of the document in characters, words, or Sentences. τ can be expressed as a percentage. In fact, (C. Y. Lin, 1999) study shows that the best performances of automatic summarization systems are found with a compression rate of τ = 15 to 30% of the length of the source document.

Understanding the source text and creating a brief and abbreviated version of it are two processes in the human generation of summaries. The summarizer's linguistic and extra-linguistic abilities and knowledge are required for both understanding the material and producing summaries. Although people can write better summaries (in terms of readability, content, form, and conciseness). Automatic text summarizing is a useful supplement to manual summation rather than a substitute.

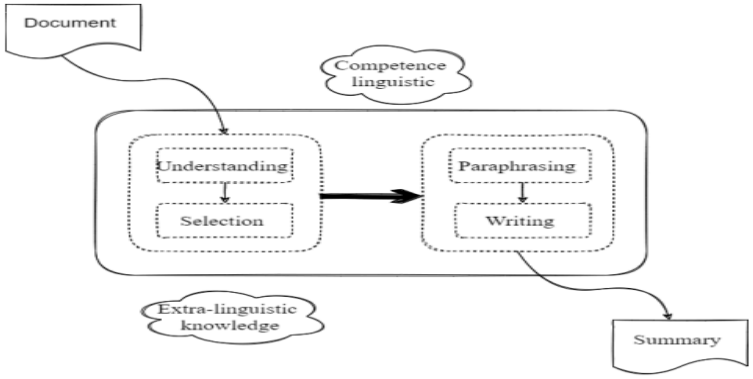


Fig. 13 shows how humans produce the summaries of an original text document.

Second model: Grok LLM model

The Grok LLM (Large Language Model) represents a notable advancement in natural language processing (NLP), particularly in the domain of text generation and understanding. the Grok LLM leverages state-of-the-art Transformer architecture, which has become a standard for large-scale language models due to its ability to handle complex linguistic tasks.

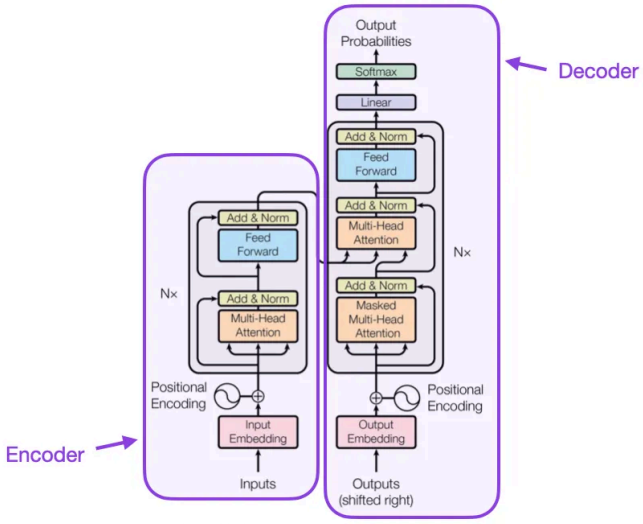


Figure 1: The Transformer - model architecture.

Key Features and Capabilities:

1. **Transformer Architecture:** Similar to models like GPT (Generative Pre-trained Transformer) series, Grok LLM employs a Transformer-based architecture. This architecture allows the model to process and generate text by capturing dependencies and relationships across long sequences of data.
2. **Scale and Parameters:** Grok LLM is characterized by its large scale, typically with millions to billions of parameters. This vast parameterization enables the model to encode and generate nuanced representations of language, facilitating tasks ranging from text completion to sentiment analysis.
3. **Training Data:** The model is trained on extensive and diverse datasets, including text from the internet, books, articles, and other sources. This broad training corpus helps Grok LLM understand and generate text that reflects the intricacies and nuances of natural language.
4. **Applications:** Grok LLM finds applications across various domains:
    - **Text Generation:** It excels in generating coherent and contextually appropriate text, making it useful for content creation, automated storytelling, and dialogue systems.
    - **Language Understanding:** The model can comprehend and respond to natural language queries, providing accurate answers and insights across different topics and domains.
    - **Translation and Summarization:** Grok LLM supports tasks such as language translation and document summarization by generating accurate and concise summaries or translated texts.
5. **Fine-tuning and Adaptability:** Beyond its pre-trained capabilities, Grok LLM can be fine-tuned on specific datasets or tasks, enhancing its performance and adaptability to particular use cases or industries.

Figure 14 show main architecture of Grok Model

Use Cases and Impact:

- **Customer Service:** Grok LLM can be integrated into customer service platforms to automate responses, understand customer queries, and provide personalized interactions.
- **Educational Tools:** It serves as a valuable tool for educational purposes, aiding in language learning, generating educational content, and providing explanations or tutoring in various subjects.
- **Research and Development:** Researchers utilize Grok LLM for exploring new avenues in NLP, developing new algorithms, and conducting experiments in language modeling and understanding.

---

5 PROJECT SIMULATION AND PERFORMANCE EVALUATION

This chapter describes the methodologies used for simulating and evaluating the performance of IntelliLearn. It includes the criteria for performance evaluation, the datasets used, and the metrics for assessing the system's effectiveness. The chapter presents the results of these evaluations, demonstrating how well the system meets its objectives and performs under various conditions .

5.1 Evaluation Metrics:

This section provides an analysis of performance metrics used to evaluate various components within the speech-to-text workflow, text embedding techniques, text summarization models, and text generation capabilities. The evaluations include comparisons against established benchmarks and highlight the strengths and weaknesses of each approach.

5.1 Speech-to-Text Workflow for Lecture Summarization

Results Analysis:

Based on the evaluation of speech-to-text models using performance metrics such as time consumption, memory usage, and word error rates (WER) using SequenceMatcher and Levenshtein methods, the following models were assessed:

Model Name	Time Performance (seconds)	Memory Usage (MB)	Word Error Rate 1 (SequenceMatcher)	Word Error Rate 2 (Levenshtein)

Vosk	56.95	5153.98	2.86	13.71
Whisper	101.74	8216.17	61.14	95.43
Wav2vec2	13.60	3395.39	15.43	76.00
SpeechBrain	151.51	1652.44	17.14	84.00

Table 3 show model name with running performance

A Simulation was conducted on a machine with the following specs:

CPU: intel® i7-10510 @ 2.30 GHz

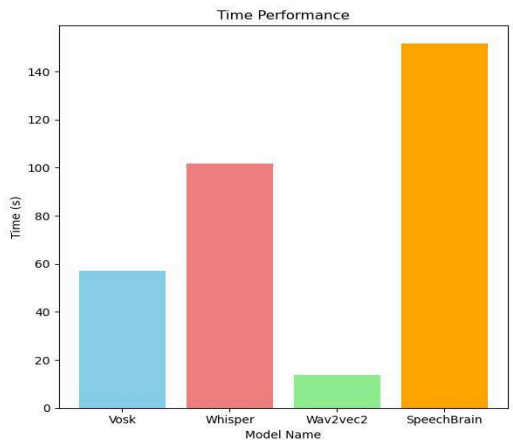
Number of Cores: 12

RAM: 16GB

GPU: Nvidia GTX 1660TI

Evaluation Rationale:

- **Selection of Whisper**: Despite higher time and memory usage, Whisper was chosen for its superior performance metrics and multilingual capabilities.
- **Comparison Summary**:
    - **Time Performance**: Vosk (56.95s), Whisper (101.74s), Wav2vec2 (13.60s), SpeechBrain (151.51s)
    - **Memory Usage**: Vosk (5153.98MB), Whisper (8216.17MB), Wav2vec2 (3395.39MB), SpeechBrain (1652.44MB)
- **Word Error Rates**:
    - **SequenceMatcher**: Vosk (2.86%), Whisper (61.14%), Wav2vec2 (15.43%), SpeechBrain (17.14%)
    - **Levenshtein**: Vosk (13.71%), Whisper (95.43%), Wav2vec2 (76.00%), SpeechBrain (84.00%)



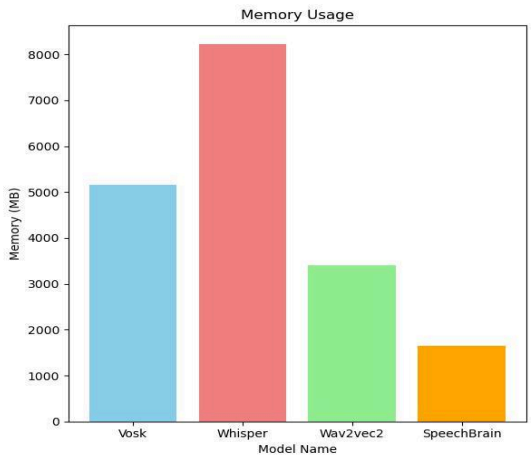
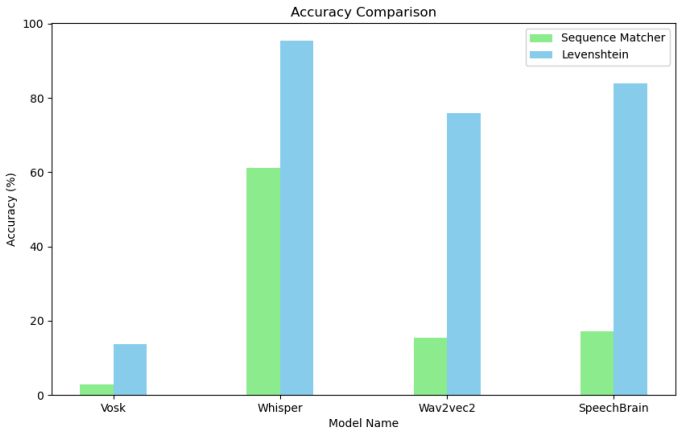


Figure 15 A & Figure 15 B

The bar charts above show the memory usage (in MB) and the time performance (in seconds) of the speech-to-text models: Vosk, Whisper, Wav2vec2, and SpeechBrain.

Figure 16

This figure shows a comparison of word error rates using the SequenceMatcher and Levenshtein methods among speech-to-text models.



Despite its slower processing time and higher memory consumption, Whisper's multilingual capabilities and high accuracy rates make it a suitable choice for lecture transcription and summarization .

5.2 Text Similarity for Paraphrase Identification, Semantic Similarity (corpus-based)

According to (W. Gomaa & A. Fahmy, 2013), measuring the similarity between words, sentences, and documents is an important component in various tasks such as information retrieval, document clustering, short answer grading, and text summarization.

The research on text similarity is categorized into four main types:

- String-Based Similarities
- Corpus-Based Similarities
- Knowledge-Based Similarities
- Hybrid Similarity Measures

Metric	Accuracy	Precision	Recall	Measure
PMI-IR	**69.9**	**70.2**	**95.2**	**81.0**
LSA	**68.4**	**69.7**	**95.2**	**80.5**

STS	**72.6**	**74.7**	**89.1**	**81.3**

Table 4 : Performance Metrics for Text Similarity Measures

This table presents accuracy, precision, recall, and F1 measure scores for text similarity measures PMI-IR, LSA, and STS, providing insights into their comparative performance in semantic text analysis tasks.

The STS model is chosen for its robust integration of corpus-based and knowledge-based techniques, combining string similarity, semantic word analysis, and order similarity functions. It excels at accurately measuring text similarity across various tasks, like paraphrase identification, offering competitive precision and efficiency compared to other methods.

5.3 Text-Embedding Techniques

In the context of text embedding:

Different text embedding techniques were evaluated based on their performance metrics:

Algorithm	Time	Memory usage
TF-IDF	**0.0124 sec**	**164.9921875 MiB**
One hot encoding	**0.0028 sec**	**180.88671875 MiB**
Bert	**0.6252 sec**	**1523.4765625 MiB**
Word2Vec	**0.0003 sec**	**8313.515625 MiB**

Table 5

This table evaluates different text embedding algorithms—TF-IDF, One Hot Encoding, BERT, and Word2Vec—based on their time efficiency and memory usage, crucial for understanding their practical applicability in various natural language processing tasks.

5.4 Text Summarization

In this section, four text summarization models were evaluated: T5, BART, BigBird-Pegasus, and LLaMA3. Each model generated summaries based on a given input text, and their outputs were assessed for semantic similarity using BERT and Word2Vec embeddings.

Semantic-Textual Similarity Evaluation

To gauge the effectiveness of the summarization models, semantic textual similarity (STS) scores were computed between the predicted summaries and the actual output using BERT and Word2Vec embeddings. The STS scores for each model are visualized below:

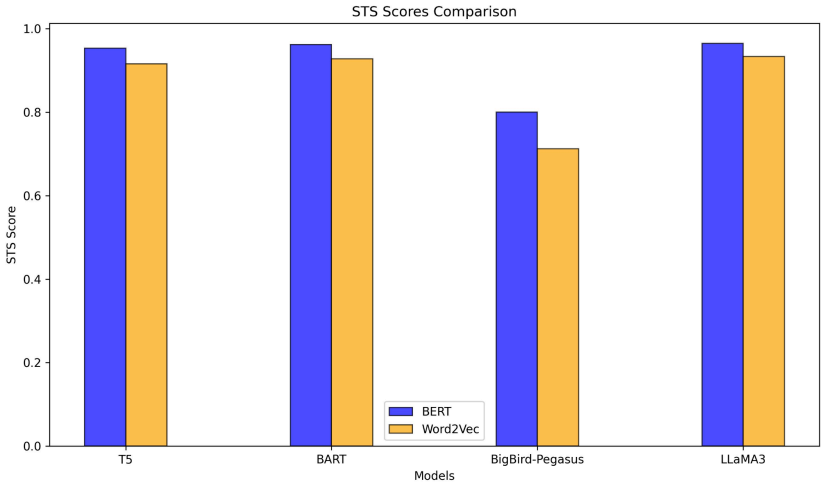


Figure 17

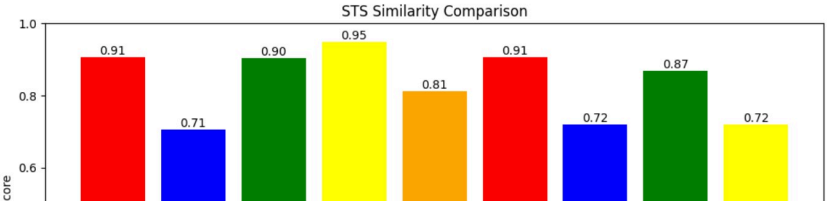
The blue bars represent STS scores obtained using BERT embeddings, while the orange bars depict scores from Word2Vec embeddings. Higher bars indicate greater semantic similarity between the model-generated summaries and the ground truth.

This evaluation provides insights into the performance of each model in capturing semantic relationships within the context of text summarization tasks. Based on the STS scores, LLaMA3 emerges as the best-performing model, demonstrating its superior ability to generate summaries that closely align with the original content compared to other evaluated models.

5.4 Text Generation

Text Generation STS Similarity Scores Comparison

In evaluating text generation models, Semantic Textual Similarity (STS) scores were computed to measure the similarity between generated texts and their corresponding ground truth. The following



models were compared based on their STS scores:

- **Falcon-40b**
- **Falcon-18b**
- **ruGPT-3**
- **Llama-3**
- **Flan-t5-xxl**
- **Mamba-130M**
- **GPT2**
- **Gpt3-small**
- **xlnet-LC**

The STS scores obtained using BERT embeddings are illustrated in the bar chart below:

Figure 18

This comparison highlights LLaMA3's effectiveness in text generation tasks, underscoring its capability to produce outputs with high semantic fidelity relative to the input texts.

5.5 Code Generation

The performance of Llama 3 70B against two comparable models on 5 LLM benchmarks.

Here, LLaMA 3 70B demonstrates competitive performance across various benchmarks:

- It achieves comparable results with Gemini Pro 1.5 on three benchmarks.
- LLaMA 3 80B outperforms on the HumanEval benchmark for code generation.
- It secures the second position, following Gemini Pro 1.5, on the MATH benchmark.

Figure 19. The performance of Llama 3 70b against two comparable models on 5 LLM benchmarks.

Results about Grok Model

Task	Dataset	Model	Metric Name	Metric Value	Global Rank	Result	Benchmark
Arithmetic Reasoning	GSM8K	GaC(Qwen2-72B-Instruct + Llama-3-70B-Instruct)	Accuracy	90.91	# 21		Compare
Multi-task Language Understanding	MMLU	GaC(Qwen2-72B-Instruct + Llama-3-70B-Instruct)	Average (%)	83.54	# 8		Compare
Question Answering	TriviaQA	GaC(Qwen2-72B-Instruct + Llama-3-70B-Instruct)	EM	79.29	# 9		Compare

Table 6 show about each task in grok model with dataset and benchmark value

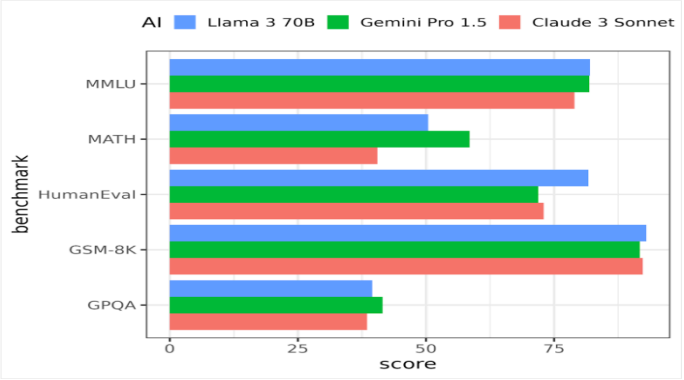
Evaluation Using Benchmark: Massive Multitask Language Understanding (MMLU)

Total Subjects Evaluated: 57

Number of Questions per Subject: 25 multiple-choice questions

Model Performance Summary:

- The model was evaluated on a total of 1,425 questions (57 subjects x 25 questions each). Estimated accuracy remains around 82%.



Chapter 6

Data type and Preprocessing flow

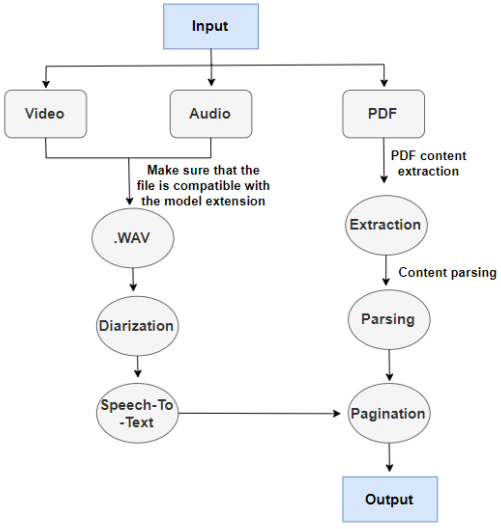
This chapter outlines the types of data that our website handles and details the preprocessing flow for each type. The preprocessing steps are crucial for converting raw data into a format that can be efficiently processed by the system, ensuring accurate and relevant outputs. By detailing the types of data and their corresponding preprocessing flows, we aim to provide a clear understanding of how our system processes and transforms user-uploaded files into meaningful and actionable content.

6.1 Data Types

Our website allows users to upload three types of files:

- PDF
- Video
- Audio

Each type of file undergoes specific preprocessing steps to extract and format the content for further analysis and summarization.



6.2 Processing Flow

- **Read and preprocess the PDF**:The PDF file is read and processed to extract text from each slide.
- **Text Pagination:**The parsed content is segmented into chunks, each containing up to 30,000 tokens.

- **Content Parsing:**Extracted text is parsed and formatted into a readable structure.
- **Summarization:**Each chunk of text undergoes summarization using an LLM (Large Language Model) with a specified prompt.
- **Combining Summaries**:Summaries from each chunk are merged into a cohesive output.
- **Final Output:**If the combined summary exceeds a set limit (e.g., 1000 words), it undergoes further summarization.
- **Output Delivery:**The final summarized output is formatted into lines of a specified length (e.g., 50 words per line) and presented

Figure 20 show preprocessing flow

Chapter seven

7. Website and Endpoints

This chapter provides an in-depth overview of the website and its endpoints, detailing their purposes, usage, and responses. The aim is to give a comprehensive understanding of how the web application operates and interacts with users.

7.1 Overview

The web application is designed to allow users to upload and process various types of files, including PDFs, audio files, and videos. Built using FastAPI, a modern web framework, the application provides a responsive interface for uploading files, processing content, and generating responses based on user prompts.

7.2 Website Functionality

The website serves as the primary interface for users to interact with the application. It features a simple and user-friendly design, making it easy for users to upload files and receive processed content. The website offers the following key functionalities:

- **File Upload**: Users can upload files through an HTML form.
- **File Processing**: Uploaded files are processed to extract and transcribe content.
- **Prompt Response**: Users can submit prompts to generate responses based on the processed file content.

7.3 Endpoint Overview

The web application defines several endpoints that handle specific tasks. Each endpoint is responsible for different aspects of file processing and user interaction. Below is a summary of the endpoints, their purposes, usage, and responses:

Endpoint	Purpose	Usage	Response
GET /	Serve an HTML upload form.	Provides a user interface for file uploads.	HTML page with an upload form.
POST /upload	Handle file uploads and process the files.	- Save the uploaded file. Validate the file type. Process PDFs, transcribe audio, or convert and process videos.	Type of file and processed content.
POST /process_prompt	Process a user prompt with the uploaded file content.	Generate a response based on the provided prompt and file content.	Generated completion or response.

Table 7 show description for each endpoint

7.4 Root Endpoint

The root endpoint ("/") is designed to serve an HTML form that allows users to upload files. This form acts as the primary user interface for the application.

- **Path**: /
- **Purpose**: To provide a simple and accessible HTML form for file uploads.
- **Usage**: Users navigate to the root URL to see the upload form.
- **Response**: The server responds with an HTML page containing the upload form.

This endpoint ensures that users have a straightforward way to begin interacting with the application by uploading their files.

7.5 File Upload Endpoint

The file upload endpoint ("/upload") is crucial for handling the actual file upload and processing. It manages several important tasks:

- **File Upload Initiation**: When a user uploads a file, the server logs the start of the process.

- **File Type Validation**: Ensures that the uploaded file is of a supported type.
- **Temporary Storage**: Saves the uploaded file temporarily on the server for processing.
- **File Processing**: Depending on the file type, the server processes PDFs, transcribes audio files, or converts and processes videos.
- **Error Handling**: Manages errors and exceptions that might occur during file processing.
- **Cleanup**: Ensures temporary files are deleted after processing to free up resources.
- **Path**: /upload
- **Purpose**: To handle file uploads and process the uploaded files.
- **Usage**: Users upload a file through the provided form or API endpoint.
- **Response**: The server responds with the type of file and its processed content.

This endpoint is essential for transforming user-uploaded files into usable content for further interactions.

7.6 PDF Processing

For PDF files, the application extracts text content, preprocesses it, and creates a summarized version. This involves reading the PDF, parsing the content, and dividing it into manageable chunks. The processed text is then summarized and returned to the user.

7.7 Audio Processing

For audio files, the application transcribes the audio content into text. If the audio file is not in WAV format, it is first converted. The transcription process makes the audio content accessible and easy to interact with.

7.8 Video Processing

For video files, the application extracts the audio track and converts it to WAV format if necessary. The audio content is then transcribed, allowing users to interact with the video content through text.

7.9 Prompt Processing Endpoint

The prompt processing endpoint ("/process_prompt") allows users to send prompts along with the processed file content to generate a response. This endpoint is designed to interact with the content generated from uploaded files.

- **Path**: /process_prompt
- **Purpose**: To process a user prompt with the uploaded file content.
- **Usage**: Users send a prompt and file content to this endpoint.
- **Response**: The server responds with the generated completion or response.

This endpoint enhances the application by allowing users to generate meaningful interactions based on their uploaded content and custom prompts.

7.10 Running the Application

The application is designed to run using a web server that serves the FastAPI application on a specified host and port. This setup ensures that the web application is accessible to users for file uploads and interactions.

By following these detailed explanations, users and developers can understand the structure and functionality of the website and its endpoints, enabling effective use and further development of the application.

Chapter 8

8. System Requirements and Diagrams

This chapter outlines the system requirements and provides diagrams to illustrate the interactions and workflows within the IntelliLearn system. The system requirements are divided into functional and non-functional categories to ensure a comprehensive understanding of the system's capabilities and performance standards.

8.1 Functional Requirements

1. **User Interaction**:
    - The system should allow users to initiate interaction through a user-friendly interface.
    - Users should be able to upload files (PDF, audio, or video) for processing.
2. **File Upload**:
    - The system must support uploading of PDF files and audio/video files.
    - Upon upload, the system should differentiate between audio/video and PDF files.
3. **Diarization**:
    - For audio files, the system should perform speaker diarization to segment the audio by speaker.
4. **Speech-to-Text (STT)**:
    - Convert audio speech into text accurately using advanced STT models.
5. **Summarization**:
    - The system should summarize the converted text from audio files.
    - For PDF files, perform text summarization directly.
6. **Add Prompt**:
    - Allow users to add prompts or queries after summarization.
    - The system should handle queries and provide relevant responses.
7. **Text Generation**:
    - Generate human-like text based on the summarized content or user prompts.
8. **Code Generation**:
    - Generate code snippets if the user’s query or the content requires it.
9. **Output Generation**:
    - The system should return generated text and code outputs to the user.

8.2 Non-Functional Requirements

1. **Performance**:
    - The system must process and return outputs (text, summaries, code) within an acceptable time frame to ensure a smooth user experience.
    - Ensure low latency for real-time speech-to-text conversion and summarization.
2. **Scalability**:
    - The system should handle multiple simultaneous users and large volumes of data efficiently.
    - It must be capable of scaling up with increased user load without compromising performance.
3. **Accuracy**:
    - Maintain high accuracy in speech-to-text conversion, text summarization, and code generation.
    - Regular updates and training of models to handle diverse datasets and improve accuracy.
4. **Usability**:
    - Provide a simple and intuitive user interface.
    - Ensure that the system is accessible to users with varying levels of technical proficiency.
5. **Security**:
    - Implement robust security measures to protect user data during upload, processing, and storage.
    - Ensure compliance with data protection regulations (e.g., GDPR, CCPA).
6. **Reliability**:
    - Ensure the system is reliable and can recover quickly from failures.
    - Regular backups and redundancy to prevent data loss.
7. **Maintainability**:
    - Design the system to be easily maintainable and upgradable.
    - Clear documentation for each module to facilitate troubleshooting and updates.
8. **Compatibility**:
    - Ensure the system is compatible with various platforms and devices.
    - Seamless integration with existing educational tools and platforms.

8.3 Diagrams That Related to the system :

Activity Diagram :

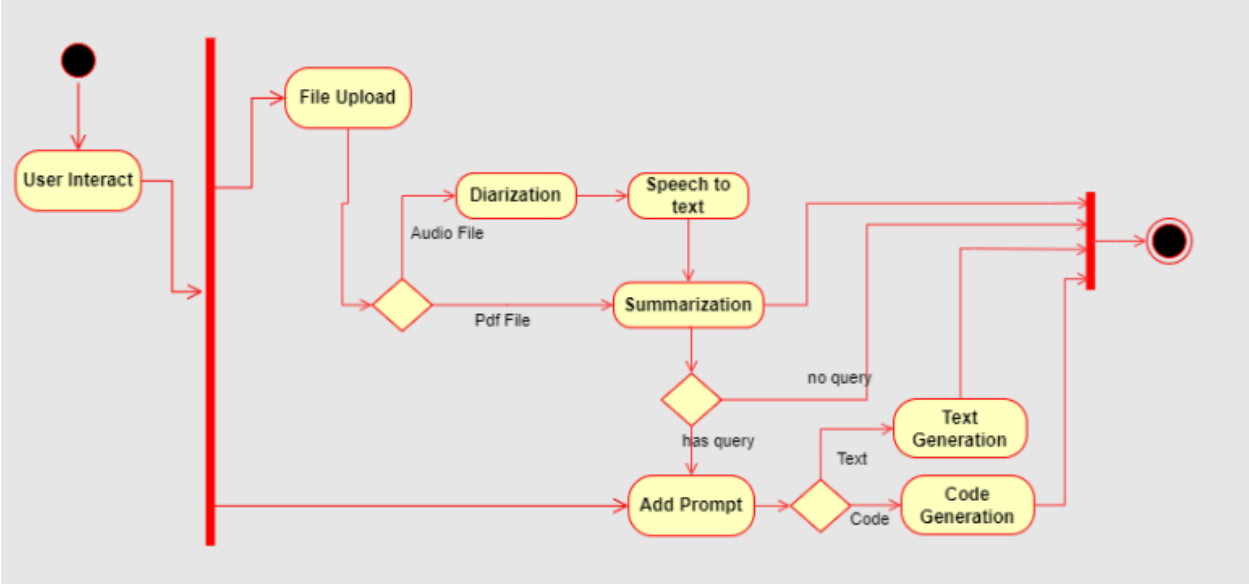


Figure 21 Activity Diagram

The activity diagram illustrates the flow of user interaction with the system, from uploading files to generating and retrieving text or code outputs.

1. **User Interact**: Initiates the process.
2. **File Upload**: Uploads either an audio/video file or a PDF file.
3. **Diarization**: Processes audio files for speaker segmentation.
4. **Speech-to-Text**: Converts audio speech to text.
5. **Summarization**: Summarizes the converted or uploaded text.
6. **Add Prompt**: Allows user to add queries.
7. **Text Generation**: Generates text based on summarized content or prompts.
8. **Code Generation**: Generates code if needed.
9. **Output**: Returns the generated outputs to the user.

Sequence Diagram :

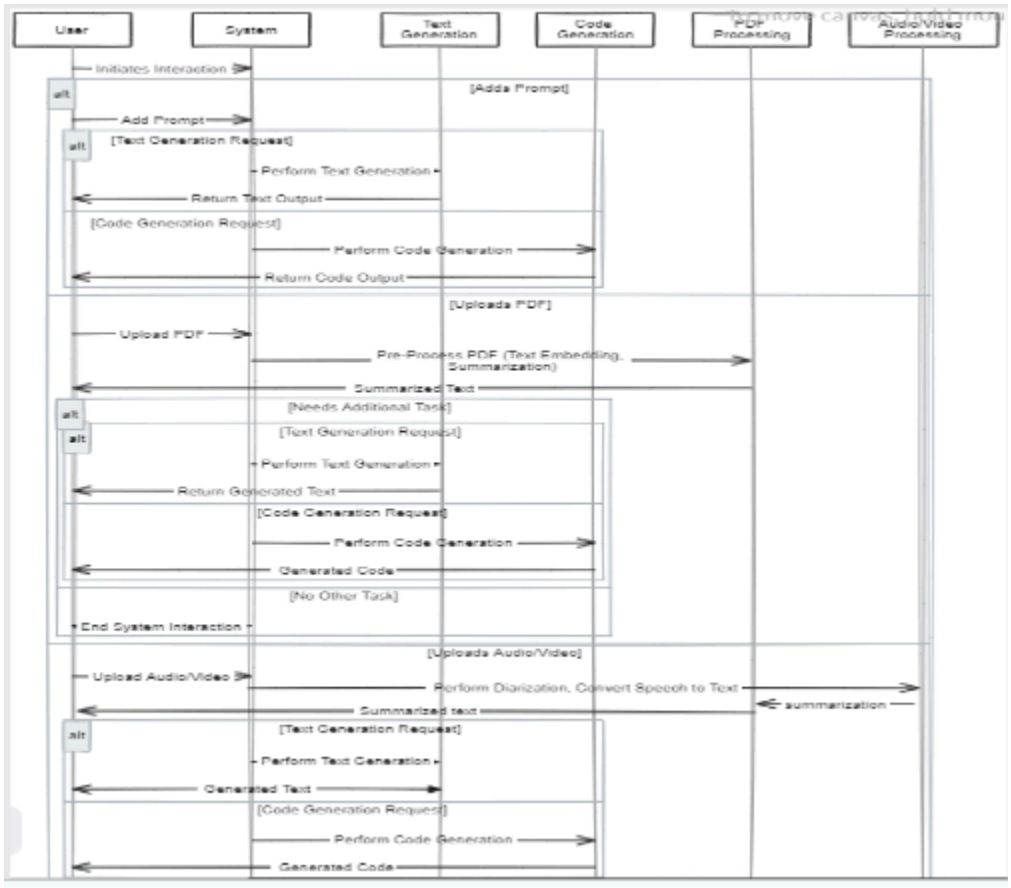


Figure 22 sequence diagram

The sequence diagram provides a detailed view of the interactions between the user, system components, and processes during the file processing and text/code generation.

1. **User**: Initiates interaction and uploads files.
2. **System**: Handles file uploads, processing, and returning outputs.
3. **Text Generation**: Processes text generation requests.
4. **Code Generation**: Handles code generation tasks.
5. **Processing**: Manages the workflow for text and audio processing

Process Flow Description for IntelliLearn :

The provided flowchart illustrates the process flow for the IntelliLearn system, detailing how different types of user-uploaded content are processed and how various tasks are performed. The flow is divided into two main branches: one for handling text uploads (such as PDFs) and the other for handling audio/video uploads.

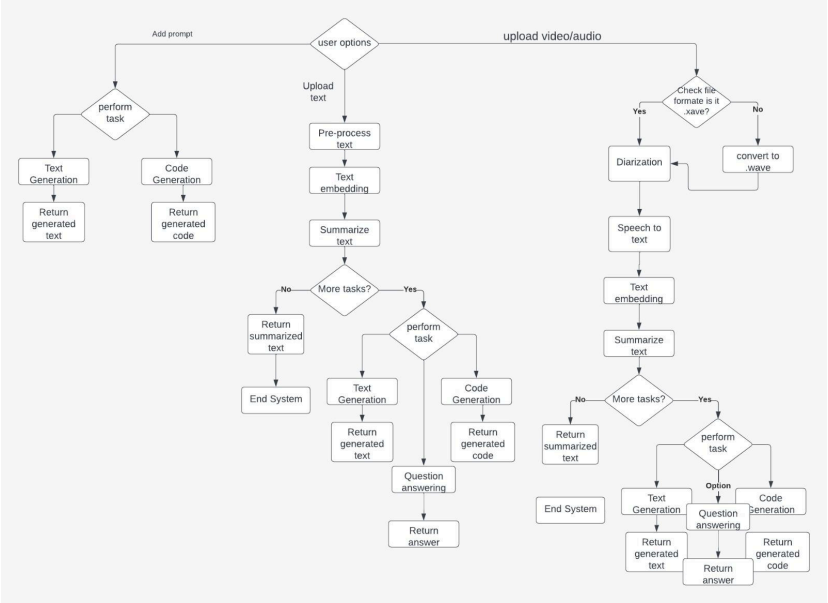


Fig 23 Show Process Flow

1. User Options

The process begins with user options, where the user can upload either text (PDF) or audio/video files.

2. Text Upload Processing

Upload Text:

- The user uploads a text file (PDF).

Pre-process Text:

- The system reads and processes the PDF to extract the text content.

Text Embedding:

- The extracted text is converted into an embedding format that the AI models can process.

Summarize Text:

- The embedded text undergoes summarization using an LLM (Large Language Model).

More Tasks?:

- The system checks if there are additional tasks to be performed based on user input or predefined workflows.
    - **No**: If no additional tasks are required, the system returns the summarized text and ends the process.
    - **Yes**: If additional tasks are required, the system moves to the next step.

Perform Task:

- Depending on the user prompt or the system's requirements, the following tasks can be performed:
    - **Text Generation**: Generates human-like text based on the summarized content.
        - **Return Generated Text**: The generated text is returned to the user.
    - **Code Generation**: Generates code snippets if relevant.
        - **Return Generated Code**: The generated code is returned to the user.
    - **Question Answering**: The system answers specific queries based on the summarized text.
        - **Return Answer**: The answer is returned to the user.

3. Audio/Video Upload Processing

Upload Audio/Video:

- The user uploads an audio or video file.

Check File Format:

- The system verifies if the file format is supported (e.g., .wav).
    - **No**: If not in the correct format, the system converts it to the required format (.wav).

Diarization:

- For audio files, the system performs speaker diarization to segment the audio by speaker.

Speech-to-Text (STT):

- The segmented audio is converted into text using advanced STT models.

Text Embedding:

- The converted text is embedded for further processing.

Summarize Text:

- The embedded text is summarized using an LLM.

More Tasks?:

- Similar to the text upload processing, the system checks for additional tasks.
    - **No**: If no additional tasks are required, the system returns the summarized text and ends the process.
    - **Yes**: If additional tasks are required, the system proceeds to perform them.

Perform Task:

- Depending on the user prompt or the system's requirements, the following tasks can be performed:
    - **Text Generation**: Generates human-like text based on the summarized content.
        - **Return Generated Text**: The generated text is returned to the user.
    - **Code Generation**: Generates code snippets if relevant.
        - **Return Generated Code**: The generated code is returned to the user.
    - **Question Answering**: The system answers specific queries based on the summarized text.
        - **Return Answer**: The answer is returned to the user.

Chapter Nine

9 BUSINESS MODEL

This chapter outlines the business model for IntelliLearn, detailing the key components that drive the value proposition, customer relationships, and revenue streams. By understanding these elements, we can ensure the sustainability and growth of IntelliLearn in the educational technology market.

9.1 Key Partnerships

Educational Institutions:

- Collaborate with schools and universities for integration into their systems.
- Provide tailored solutions to meet specific educational needs and curricula.

Technology Providers:

- Partner with leading AI and cloud service providers for robust, scalable solutions.
- Ensure the system's infrastructure can handle large-scale operations efficiently.

9.2 Key Activities

AI Model Development:

- Continuously develop and improve AI and NLP models to enhance system capabilities.
- Keep models updated with the latest research and technological advancements.

Content Processing:

- Efficiently process and summarize diverse content types, including PDFs, images, audio files, and videos.
- Ensure high accuracy and relevance in content summarization and text generation.

Customer Support:

- Provide ongoing support to users to resolve issues and improve user experience.
- Offer training and resources to help users maximize the benefits of the system.

Marketing:

- Promote the product to reach target audiences through various channels.
- Highlight the unique value propositions and benefits of IntelliLearn to potential users.

9.3 Key Resources

AI and NLP Technology:

- Core technology for content processing, including advanced algorithms and models.
- Ensure the technology is continuously improved and adapted to new challenges.

Cloud Services:

- Infrastructure to handle large-scale operations and ensure reliable performance.
- Utilize scalable cloud solutions to accommodate growing user base and data volume.

Development Team:

- Skilled professionals in AI, NLP, and software development.
- Ensure the system is developed, maintained, and updated efficiently.

Customer Support Team:

- Dedicated team to assist users with any issues or queries.
- Provide high-quality support to enhance user satisfaction and retention.

9.4 Value Propositions

Efficient Content Processing:

- Use advanced AI to summarize diverse content types quickly and accurately.
- Provide users with concise and relevant information, saving time and effort.

User-Friendly Interaction:

- Offer a natural chatbot interface for seamless information retrieval.
- Ensure the system is intuitive and easy to use for users of all technical levels.

Educational Support:

- Automated question generation and answering to aid learning.
- Enhance students' understanding and retention of educational material.

9.5 Customer Relationships

Subscription Plans:

- Offer various plans to meet different customer needs, from individual users to large institutions.
- Provide flexibility in pricing and features to accommodate diverse user requirements.

Freemium Model:

- Basic features available for free, with premium features offered for a fee.

- Encourage users to try the system and upgrade to access advanced functionalities.

Customer Support:

- Continuous support to resolve issues and improve user experience.
- Offer resources and training to help users make the most of the system.

9.6 Channels

Online Platform:

- Dedicated website and mobile application for easy access to the system.
- Ensure the platform is accessible and user-friendly across devices.

Institutional Integration:

- Embed the tool within educational institutions' learning management systems.
- Provide seamless integration to enhance the learning experience within existing platforms.

9.7 Customer Segments

Students and Educators:

- Primary users needing enhanced learning tools and resources.
- Focus on improving accessibility, comprehension, and engagement in education.

Researchers and Professionals:

- Secondary users requiring efficient summarization and information retrieval.
- Provide tools to support research, professional development, and continuous learning.

9.8 Cost Structure

Development Costs:

- AI model development, API licensing fees, and cloud services.
- Investment in technology and infrastructure to ensure high performance and reliability.

Operational Costs:

- Server maintenance, customer support, and marketing expenses.
- Ongoing costs to keep the system running smoothly and support user growth.

Research and Development:

- Continuous improvement of AI models and exploration of new functionalities.
- Invest in innovation to maintain competitive advantage and meet evolving user needs.

9.9 Revenue Streams

Subscription-Based:

- Different tiers for individual users, educational institutions, and enterprise clients.
- Offer various pricing plans to cater to different segments and needs.

Freemium Model:

- Basic features available for free, with advanced features offered for a fee (e.g., detailed summarizations, personalized learning modules).
- Encourage users to upgrade to access premium functionalities and enhance their experience.

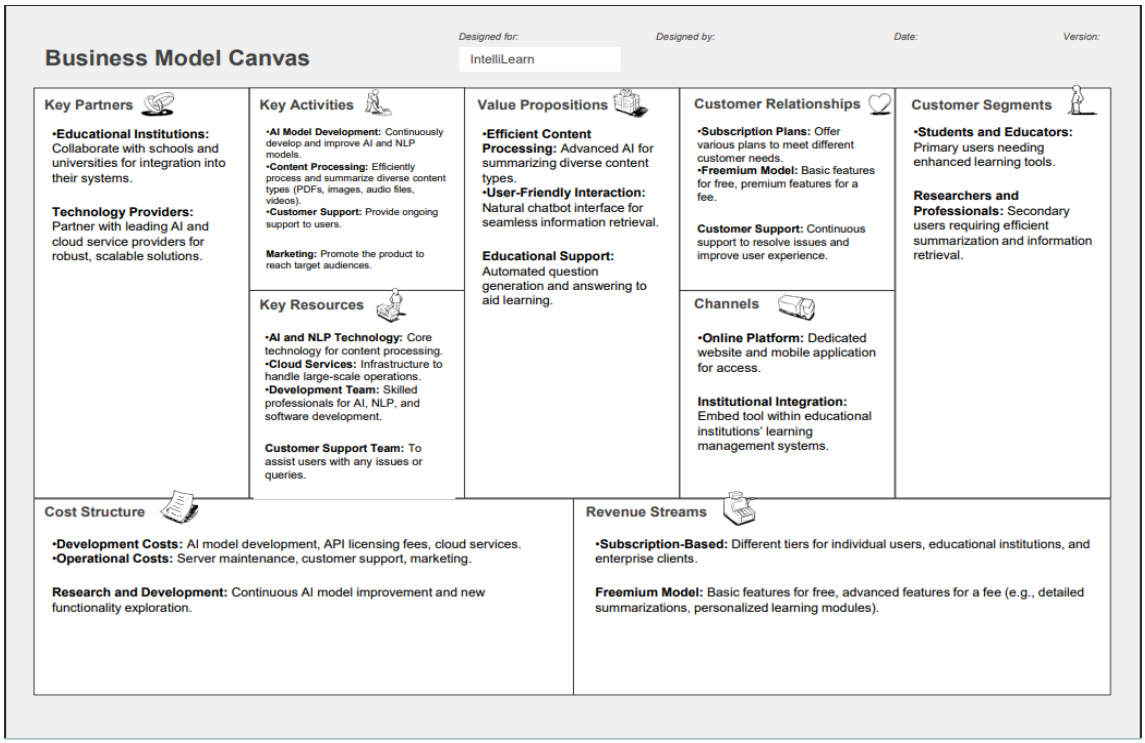


Figure 24 Business model canvas

Chapter TEN

10 CONCLUSION AND FUTURE WORK

This chapter summarizes the achievements and impact of the IntelliLearn project, highlighting how it enhances content accessibility and comprehension through advanced AI technologies. It also outlines potential future developments and enhancements to further improve the system's capabilities and user experience. By reflecting on the project's accomplishments and setting a path for future work, this chapter underscores the ongoing commitment to innovation and excellence in educational technology.

Conclusion

In this project, we developed **"IntelliLearn,"** an advanced AI system designed to enhance content accessibility and comprehension through sophisticated text analysis, summarization, and generation. By integrating cutting-edge technologies such as OpenAI's Whisper and xAI's Grok, IntelliLearn processes a wide variety of content types, including PDFs, images, audio files, and videos. Leveraging state-of-the-art NLP and machine learning techniques, IntelliLearn facilitates better understanding and retention of complex information. The system effectively bridges the gap between vast information sources and users, providing a user-friendly interface and robust educational support. IntelliLearn not only enhances the accessibility of educational content but also streamlines the process of information extraction and utilization, thereby significantly contributing to the field of educational technology.

Future Work

1. **Integration of Additional Language Models**:
    - Incorporate more advanced language models and domain-specific knowledge bases to enhance performance and context awareness.
    - Explore the integration of multilingual support to cater to a broader audience.
2. **Reinforcement Learning for Optimization**:
    - Investigate the application of reinforcement learning techniques to optimize text summarization and generation processes.
    - Utilize reinforcement learning to adapt and improve model responses based on user interactions and feedback.
3. **Enhanced Accessibility Features**:
    - Develop and implement additional accessibility features to support users with disabilities, ensuring inclusivity.

- Gather comprehensive user feedback to identify areas for improvement and iteratively enhance the system's usability.

4. **Advanced Video Summarization**:
    - Expand the system's capabilities to include video summarization, allowing users to extract key information from educational videos efficiently.
    - Implement lecture summary functionalities to assist students in reviewing and understanding lecture content.

5. **Visual Aids Generation**:
    - Develop tools to automatically generate visual aids such as charts, graphs, and infographics from text content, aiding visual learners.
    - Enhance the system's ability to create interactive and engaging educational materials.
6. **User-Centric Enhancements**:
    - Continuously refine the user interface based on user experience research to ensure it remains intuitive and user-friendly.
    - Introduce personalized learning paths and recommendations based on user performance and preferences.
7. **Scalability and Performance Improvements**:
    - Invest in scalable cloud infrastructure to handle increasing data volumes and user load without compromising performance.
    - Optimize algorithms for faster processing times and lower latency in real-time applications.
8. **Compliance and Security Enhancements**:
    - Ensure ongoing compliance with data protection regulations (e.g., GDPR, CCPA) to maintain user trust and data security.
    - Implement advanced security measures to protect user data during transmission, processing, and storage.
9. **Educational Partnerships**:
    - Collaborate with more educational institutions to integrate IntelliLearn into their systems, tailoring solutions to meet specific needs.
    - Establish partnerships with educational content creators to expand the range of supported materials.
10. **Comprehensive Evaluation and Benchmarking**:
    - Conduct extensive evaluations and benchmarking against existing educational tools to continuously measure and improve IntelliLearn's effectiveness.
    - Publish findings in academic and industry forums to contribute to the broader field of educational technology.

11 REFERENCES

1. OpenAI. "Whisper: A Robust Speech Recognition System." OpenAI, 2023.
2. Baevski, Alexei, et al. "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations." NeurIPS, 2020.
3. Raffel, Colin, et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." arXiv, 2020.
4. See, Abigail, et al. "Get To The Point: Summarization with Pointer-Generator Networks." ACL, 2017. Link
5. Brown, Tom, et al. "Language Models are Few-Shot Learners." NeurIPS, 2020.
6. Radford, Alec, et al. "GPT-3: Language Models are Few-Shot Learners." OpenAI, 2020.
7. Devlin, Jacob, et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL, 2019.
8. Mikolov, Tomas, et al. "Efficient Estimation of Word Representations in Vector Space." arXiv, 2013.
9. Du, Xinya, et al. "Learning to Ask: Neural Question Generation for Reading Comprehension." ACL, 2017.
10. Wang, Di, et al. "PathQG: Neural Question Generation from Facts." ACL, 2020.
11. Anguera, Xavier, et al. "Speaker Diarization: A Review of Recent Research." IEEE Transactions on Audio, Speech, and Language Processing, 2012.
12. Zhang, Aonan, et al. "Fully Supervised Speaker Diarization." ICASSP, 2019.
13. Jurafsky, Dan, and Martin, James H. "Speech and Language Processing." Pearson, 2019.
14. Goldberg, Yoav. "A Primer on Neural Network Models for Natural Language Processing." Journal of Artificial Intelligence Research, 2016.
15. Salton, Gerard, and Buckley, Christopher. "Term-weighting approaches in automatic text retrieval." Information Processing & Management, 1988.
16. Ramos, Juan. "Using TF-IDF to Determine Word Relevance in Document Queries." Technical Report, 2003.
17. Nallapati, Ramesh, et al. "Abstractive Text Summarization using Sequence-to-Sequence RNNs and Beyond." CoNLL, 2016.
18. Nenkova, Ani, and McKeown, Kathleen. "A Survey of Text Summarization Techniques." Mining Text Data, 2012.

19. Gillick, Dan, et al. "Global Inference for Sentence Compression: An Integer Linear Programming Approach." Journal of Artificial Intelligence Research, 2009. Link
20. BAİDOO-ANU, D., & OWUSU ANSAH, L. (2023). Education in the Era of Generative Artificial Intelligence (AI): Understanding the Potential Benefits of ChatGPT in Promoting Teaching and Learning. Journal of AI, 7(1), 52-62. https://doi.org/10.61969/jai.1337500
21. Bayoudh, K., Knani, R., Hamdaoui, F. et al. A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. Vis Comput 38, 2939–2970 (2022). https://doi.org/10.1007/s00371-021-02166-7
22. Yu, H., Miao, C., Leung, C., & White, T. J. (2017). Towards AI-powered personalization in MOOC learning. npj Science of Learning, 2(1), 15. https://www.nature.com/articles/s41539-017-0016-3
23. Xiao, C., Xu, S. X., Zhang, K., Wang, Y., & Xia, L. (2023, July). Evaluating reading comprehension exercises generated by LLMs: A showcase of ChatGPT in education applications. In Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023) (pp. 610-625). https://aclanthology.org/2023.bea-1.52/
24. Chen, L., Chen, P., & Lin, Z. (2020). Artificial intelligence in education: A review. Ieee Access, 8, 75264-75278. https://ieeexplore.ieee.org/abstract/document/9069875
25. García-Peñalvo, F. J. (2023). The perception of Artificial Intelligence in educational contexts after the launch of ChatGPT: Disruption or Panic?. https://repositorio.grial.eu/handle/grial/2838
26. Sreelakshmi, A. S., Abhinaya, S. B., Nair, A., & Nirmala, S. J. (2019, November). A question answering and quiz generation chatbot for education. In 2019 Grace Hopper Celebration India (GHCI) (pp. 1-6). IEEE. https://ieeexplore.ieee.org/abstract/document/9071832
27. Alasadi, E. A., & Baiz, C. R. (2023). Generative AI in education and research: Opportunities, concerns, and solutions. Journal of Chemical Education, 100(8), 2965-2971. https://pubs.acs.org/doi/abs/10.1021/acs.jchemed.3c00323
28. Chan, C. K. Y. (2023). A comprehensive AI policy education framework for university teaching and learning. International journal of educational technology in higher education, 20(1), 38. https://link.springer.com/article/10.1186/s41239-023-00408-3
29. Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., ... & Hu, X. (2023). Harnessing the power of llms in practice: A survey on chatgpt and beyond. ACM

Transactions on Knowledge Discovery from Data. https://dl.acm.org/doi/abs/10.1145/3649506
30. Rudnicky, A. I., Hauptmann, A. G., & Lee, K. F. (1994). Survey of current speech technology. Communications of the ACM, 37(3), 52-57. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=762790cfcd9074e663feaf4112b08055dd21903b
31. Roger, V., Farinas, J., & Pinquier, J. (2022). Deep neural networks for automatic speech processing: a survey from large corpora to limited data. EURASIP Journal on Audio, Speech, and Music Processing, 2022(1), 19. https://link.springer.com/article/10.1186/s13636-022-00251-w
32. Tokuda, K., Yoshimura, T., Masuko, T., Kobayashi, T., & Kitamura, T. (2000, June). Speech parameter generation algorithms for HMM-based speech synthesis. In 2000 IEEE international conference on acoustics, speech, and signal processing. Proceedings (Cat. No. 00CH37100) (Vol. 3, pp. 1315-1318). IEEE. https://ieeexplore.ieee.org/abstract/document/861820/
33. Graves, A., & Schmidhuber, J. (2005, July). Framewise phoneme classification with bidirectional LSTM networks. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005. (Vol. 4, pp. 2047-2052). IEEE. https://ieeexplore.ieee.org/abstract/document/1556215/
34. Tachibana, H., Uenoyama, K., & Aihara, S. (2018, April). Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention. In 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 4784-4788). IEEE. https://ieeexplore.ieee.org/abstract/document/8461829
35. Nakamura, K., Hashimoto, K., Oura, K., Nankaku, Y., & Tokuda, K. (2019). Singing voice synthesis based on convolutional neural networks. arXiv preprint arXiv:1904.06868. https://arxiv.org/abs/1904.06868
36. Ping, W., Peng, K., Gibiansky, A., Arik, S. O., Kannan, A., Narang, S., ... & Miller, J. (2017). Deep voice 3: Scaling text-to-speech with convolutional sequence learning. arXiv preprint arXiv:1710.07654. https://arxiv.org/abs/1710.07654
37. Chorowski, J. K., Bahdanau, D., Serdyuk, D., Cho, K., & Bengio, Y. (2015). Attention-based models for speech recognition. Advances in neural information processing systems, 28. https://proceedings.neurips.cc/paper/2015/hash/1068c6e4c8051cfd4e9ea8072e3189e2-Abstract.html

38. Synnaeve, G., Xu, Q., Kahn, J., Likhomanenko, T., Grave, E., Pratap, V., ... & Collobert, R. (2019). End-to-end asr: from supervised to semi-supervised learning with modern architectures. arXiv preprint arXiv:1911.08460. https://arxiv.org/abs/1911.08460
39. Li, B., Chang, S. Y., Sainath, T. N., Pang, R., He, Y., Strohman, T., & Wu, Y. (2020, May). Towards fast and accurate streaming end-to-end ASR. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 6069-6073). IEEE. https://ieeexplore.ieee.org/abstract/document/9054715
40. Tu, T., Chen, Y. J., Yeh, C. C., & Lee, H. Y. (2019). End-to-end text-to-speech for low-resource languages by cross-lingual transfer learning. arXiv preprint arXiv:1904.06508. https://arxiv.org/abs/1904.06508
41. Singh, P., Srivastava, R., Rana, K. P. S., & Kumar, V. (2021). A multimodal hierarchical approach to speech emotion recognition from audio and text. Knowledge-Based Systems, 229, 107316. https://www.sciencedirect.com/science/article/abs/pii/S0950705121005785
42. Deng, L. (2016). Deep learning: from speech recognition to language and multimodal processing. APSIPA Transactions on Signal and Information Processing, 5, e1. https://www.cambridge.org/core/journals/apsipa-transactions-on-signal-and-information-processing/article/deep-learning-from-speech-recognition-to-language-and-multimodal-processing/F1F1FDB71C597878FE31E5964CF5DFAE
43. Benzeghiba, M., De Mori, R., Deroo, O., Dupont, S., Erbes, T., Jouvet, D., ... & Wellekens, C. (2007). Automatic speech recognition and speech variability: A review. Speech communication, 49(10-11), 763-786. https://www.sciencedirect.com/science/article/abs/pii/S0167639307000404
44. El-Kassas, W. S., Salama, C. R., Rafea, A. A., & Mohamed, H. K. (2021). Automatic text summarization: A comprehensive survey. Expert systems with applications, 165, 113679. https://www.sciencedirect.com/science/article/abs/pii/S0957417420305030
45. Widyassari, A. P., Rustad, S., Shidik, G. F., Noersasongko, E., Syukur, A., & Affandy, A. (2022). Review of automatic text summarization techniques & methods. Journal of King Saud University-Computer and Information Sciences, 34(4), 1029-1046. https://www.sciencedirect.com/science/article/pii/S1319157820303712
46. Moratanch, N., & Chitrakala, S. (2017, January). A survey on extractive text summarization. In 2017 international conference on computer, communication and

signal processing (ICCCSP) (pp. 1-6). IEEE. https://ieeexplore.ieee.org/abstract/document/7944061
47. Gupta, S., & Gupta, S. K. (2019). Abstractive summarization: An overview of the state of the art. Expert Systems with Applications, 121, 49-65. https://www.sciencedirect.com/science/article/abs/pii/S0957417418307735
48. Carenini, G., Cheung, J. C. K., & Pauls, A. (2013). Multi‐document summarization of evaluative text. Computational Intelligence, 29(4), 545-576. https://onlinelibrary.wiley.com/doi/full/10.1111/j.1467-8640.2012.00417.x
49. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. https://arxiv.org/abs/1301.3781
50. Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543). https://aclanthology.org/D14-1162.pdf
51. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://arxiv.org/abs/1810.04805
52. Värtinen, S., Hämäläinen, P., & Guckelsberger, C. (2022). Generating role-playing game quests with GPT language models. IEEE transactions on games. https://ieeexplore.ieee.org/abstract/document/9980408
53. Data generation as sequential decision making Adv. Neural Inf. Process. Syst. (2015), pp. 3249-3257 https://proceedings.neurips.cc/paper/2015/hash/09b15d48a1514d8209b192a8b8f34e48-Abstract.html
54. Z. Yang, Z. Hu, R. Salakhutdinov, T. Berg-Kirkpatrick Improved variational autoencoders for text modeling using dilated convolutions Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org (2017), pp. 3881-3890 https://proceedings.mlr.press/v70/yang17d.html?ref=https://githubhelp.com
55. Guo Q, Cao J, Xie X, Liu S, Li X, Chen B and Peng X. Exploring the Potential of ChatGPT in Automated Code Refinement: An Empirical Study. Proceedings of the 46th IEEE/ACM International Conference on Software Engineering. (1-13). https://dl.acm.org/doi/abs/10.1145/3597503.3623306
56. Zhou X, Sun Z and Li G. (2024). DB-GPT: Large Language Model Meets Database. Data Science and Engineering. 10.1007/s41019-023-00235-6. https://link.springer.com/article/10.1007/s41019-023-00235-6

57. G. Murtarelli, A. Gregory, S. Romenti A conversation-based perspective for shaping ethical human–machine interactions: The challenge of chatbots Journal of Business Research, 129 (2021), pp. 927-935 https://www.sciencedirect.com/science/article/pii/S0148296320305944
58. B.R. Ranoliya, N. Raghuwanshi, S. Singh Chatbot for university related faqs 2017 international conference on advances in computing, communications, and informatics (ICACCI), pages 1525–1530. IEEE (2017) https://ieeexplore.ieee.org/abstract/document/8126057
59. Hwang, G., & Chen, N. S. (2014). The effects of speech recognition technology on learning performance in a synchronous cyber classroom environment. Computers & Education, 71, 246-254. https://www.researchgate.net/publication/267263862_Effects_of_Speech-to-Text_Recognition_Application_on_Learning_Performance_in_Synchronous_Cyber_Classrooms/references
60. Liu, X., & Li, Q. (2015). A review of applying speech recognition technology to enhance learning. Educational Technology & Society, 18(3), 101-112.https://www.researchgate.net/publication/267811277_Review_of_Speech-to-Text_Recognition_Technology_for_Enhancing_Learning
61. Neto, J. L., Freitas, A. A., & Kaestner, C. A. (2002). Automatic text summarization using a machine learning approach. In Advances in Artificial Intelligence: 16th Brazilian Symposium on Artificial Intelligence, SBIA 2002 Porto de Galinhas/Recife, Brazil, November 11–14, 2002 Proceedings 16 (pp. 205-215). Springer Berlin Heidelberg.https://link.springer.com/chapter/10.1007/3-540-36127-8_20
62. Ahmed Abdelali, Hamdy Mubarak, Shammur Absar Chowdhury, Maram Hasanain, Basel Mousi, Sabri Boughorbel, Yassine El Kheir, Daniel Izham, Fahim Dalvi, Majd Hawasly, et al. 2023. Benchmarking Arabic AI with Large Language Models. arXiv preprint arXiv:2305.14982(2023) https://arxiv.org/abs/2305.14982
63. Kabir Ahuja, Rishav Hada, Millicent Ochieng, Prachi Jain, Harshita Diddee, Samuel Maina, Tanuja Ganu, Sameer Segal, Maxamed Axmed, Kalika Bali, et al. 2023. Mega: Multilingual evaluation of generative ai. arXiv preprint arXiv:2303.12528(2023). .https://arxiv.org/abs/2303.12528
64. Prasad, V. (2015). Voice recognition system: speech-to-text. Journal of Applied and Fundamental Sciences, 1(2), 191. https://journals.dbuniversity.ac.in/ojs/index.php/JFAS/article/view/103

65. Wang, S., Yang, C. H. H., Wu, J., & Zhang, C. (2023). Can whisper perform speech-based in-context learning. arXiv preprint arXiv:2309.07081. https://arxiv.org/abs/2309.07081
66. Anguera, X., Bozonnet, S., Evans, N., Fredouille, C., Friedland, G., & Vinyals, O. (2012). Speaker diarization: A review of recent research. IEEE Transactions on audio, speech, and language processing, 20(2), 356-370. https://ieeexplore.ieee.org/abstract/document/6135543
67. Tranter, S. E., & Reynolds, D. A. (2006). An overview of automatic speaker diarization systems. IEEE Transactions on audio, speech, and language processing, 14(5), 1557-1565. https://ieeexplore.ieee.org/abstract/document/1677976
68. H.Gomaa, W., & A. Fahmy, A. (2013, April 18). A Survey of Text Similarity Approaches. International Journal of Computer Applications, 68(13), 13–18. https://doi.org/10.5120/11638-7118
69. Islam, A., & Inkpen, D. (2008). Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data (TKDD), 2(2), 1-25.https://dl.acm.org/doi/abs/10.1145/1376815.1376819
70. Selva Birunda, S., & Kanniga Devi, R. (2021). A review on word embedding techniques for text classification. Innovative Data Communication Technologies and Application: Proceedings of ICIDCA 2020, 267-281. https://link.springer.com/chapter/10.1007/978-981-15-9651-3_23
71. Das, B., & Chakraborty, S. (2018). An improved text sentiment classification model using TF-IDF and next word negation. arXiv preprint arXiv:1806.06407. https://arxiv.org/abs/1806.06407
72. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. https://arxiv.org/abs/1301.3781
73. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://arxiv.org/abs/1810.04805
74. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30. https://proceedings.neurips.cc/paper/7181-attention-is-all
75. Yu, W., Zhu, C., Li, Z., Hu, Z., Wang, Q., Ji, H., & Jiang, M. (2022). A survey of knowledge-enhanced text generation. ACM Computing Surveys, 54(11s), 1-38.https://dl.acm.org/doi/full/10.1145/3512467

76. Sellam, T., Das, D., & Parikh, A. P. (2020). BLEURT: Learning robust metrics for text generation. arXiv preprint arXiv:2004.04696.https://arxiv.org/pdf/2004.04696.pdf
77. Luhn H (1958) The automatic creation of literature abstracts. IBM J Res Dev 2:159–165
78. Jones KS (2007) Automatic summarising: the state of the art. Inf Process Manag 43:1449–1481. doi:10.1016/ j.ipm.2007.03.009
79. Goldstein J, Mittal V, Carbonelll J, Kantrowitz M (2000) Multi-document summarization by sentence extraction. In: NAACL-ANLP 2000 workshop on automatic summarization. pp 40–48
80. Radev, D. R., Jing, H., Styś, M., & Tam, D. (2004). Centroid-based summarization of multiple documents. Information Processing and Management, 40(6), 919– 938. https://doi.org/10.1016/j.ipm.2003.10.006
81. Hovy, E., & Lin, C.-Y. (1996). Automated text summarization and the SUMMARIST system. 197. https://doi.org/10.3115/1119089.1119121