

# **PROJECT REPORT**

**On**

**Smart Question Answering System**

**In partial fulfilment of the requirement for the award of the degree of**

## **BACHELOR OF TECHNOLOGY**

**IN**

## **COMPUTER SCIENCE & ENGINEERING**

**(Maulana Abul Kalam Azad University of Technology, Formerly known as West Bengal  
University of Technology)**

**Submitted by**

<b>NAME</b>	<b>ROLL NO.</b>
Sagnik Chakraborty	33200120014
Samiran Roy Bhowmik	33200120015
Somnath Mondal	33200120033
Sourav Saha	33200121136
Satyabrata Mukherjee	33200121123

**Under the guidance**

**of**

**Ms. Dipannyta Nandi**

**Assistant Professor, Department of Computer Science and Engineering**

**TECHNO INTERNATIONAL BATANAGAR**

**Maheshtala, Kolkata – 700141, West Bengal, India 2022-2023**



**TECHNO INTERNATIONAL BATANAGAR**  
**MAHESHTALA, KOLKATA, PIN: 700141, WEST BENGAL**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  
**2022-2023**

**CERTIFICATE**

*Certified that the project work entitled **SMART QUESTION ANSWERING SYSTEM** is a bona fide work carried out by*

NAME	ROLL NO
Sagnik Chakraborty	33200120014
Samiran Roy Bhowmik	33200120015
Somnath Mondal	33200120033
Sourav Saha	33200121136
Satyabrata Mukherjee	33200121123

*In partial fulfilment for the award for degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING** of the **MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY**, formerly known as **WEST BENGAL UNIVERSITY OF TECHNOLOGY, KOLKATA** during the year 2022-2023. It is certified that all corrections / suggestions indicated for Internal Assessment has been incorporated in the report deposited in the Department. The project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for Bachelor of Technology Degree.*

---

**Ms.Dipannyta Nandi**  
Project Guide

---

**Mr.Subhankar Guha**  
HOD, CSE

---

**Prof.(Dr.) Ratikanto Sahoo**  
Director

Name of the Student:

University Roll No:

Name of Examiner: Signature with Date

## ACKNOWLEDGEMENT

I am greatly indebted to our project Guide **Ms. Dipannyta Nandi, Assistant Professor of Computer Science and Engineering Department** for providing us all possible help and support while doing this project. Without her guidance the project would not get such a progress.

I also express my sincere thanks to **Mr. Subhankar Guha, Head of the Department, Computer Science and Engineering**, whose ceaseless cooperation made it possible in completion of this project for extending all possible help.

Finally, I would like to thank **PROF.(Dr.) Ratikanto Sahoo, Director, Techno International Batanagar** for his colossal support and encouragement.

Lastly, I would like to extend our thanks to all respected Faculty Members and friends directly or indirectly associated with our project, who contributed their personal level best which enabled us for a successful completion of the project.

## **TABLE OF CONTENTS**

<b>TOPICS</b>	<b>PAGES</b>
<b>Chapter – 1 --- Introduction</b>	<b>6</b>
1.1 Overview	9
1.2 Research Objectives	14
1.3 Literature Review	16
1.4 Theoretical Foundations	18
1.4.1 GloVe Embeddings and its role	18
1.4.2 Word2Vec and its role	19
1.4.3 BERT and its role	20
1.4.4 Sense2vec and its role	21
<b>Chapter - 2 --- System Block Diagram</b>	<b>23</b>
2.1 System Block Diagram Overview	25
<b>Chapter - 3 --- Project Methodology</b>	<b>31</b>
3.1 Objective Identification	33
3.2 Technical Platform	34
3.3 Data Processing	37
3.4 MCQ Generation	37
3.5 System Architecture	38
3.6 Testing and Validation	39

<b>Chapter – 4 --- Result and Discussions</b>	41
4.1 Calculating the performance	42
<b>Chapter – 5 --- Merits and Limitations</b>	46
5.1 Merits and Limitations of Smart Question Answering Systems	47
<b>Chapter – 6 --- Scope for Future Development</b>	48
<b>Chapter – 7 --- Conclusion</b>	51
<b>References</b>	53

## **LIST OF FIGURES**

FIGURE NO.	FIGURE NAME	PAGE NO
1.	Figure-1 : BERT Model	21
2.	Figure-2 : System Block Diagram Overview	25
3.	Figure-3 : System Block Diagram of Word2vec	28
4.	Figure-4 : System Block Diagram of BERT	30
5.	Figure-5 : System Block Diagram of MCQ generation	41
6.	Figure-6 : Upload Section	42
7.	Figure-7 : : Processing Answer Using BERT	42
8.	Figure-8: Processing Answer Using Fine-tuning BERT	44

# **CHAPTER- 1**

## **INTRODUCTION**

# ABSTRACT

This project presents a comprehensive Smart Question-Answering System designed to enhance document-based information retrieval and assessment creation. The system integrates three state-of-the-art Natural Language Processing (NLP) models—Word2Vec, GloVe, and BERT—to allow users to query PDF documents and receive accurate, contextually relevant answers. Additionally, the system features an automated multiple-choice question (MCQ) generation capability utilizing the Sense2Vec model, aimed at assisting educators in developing assessment materials efficiently. The core functionality of the project includes user authentication, document upload, NLP processing, and result display. Users can select their preferred QA model, upload a PDF document, and ask questions related to the document's content. The system processes the document, extracts and cleans the text, and uses the selected model to find and display the most relevant answers. For the MCQ generation feature, users upload a PDF document, which the system processes to automatically generate a set of MCQs. This project addresses the challenge of efficiently retrieving information from large documents and creating assessment materials, providing a user-friendly interface that caters to a wide range of users, including students and educators. Through detailed evaluation, the project aims to measure the accuracy, relevance, and contextuality of answers provided by each NLP model, ultimately identifying the most effective model for document-based question answering. The system's design and implementation offer a robust solution for improving educational tools and enhancing the user experience in document-based information retrieval.



# INTRODUCTION

## 1.1 Overview

In today's digital age, the vast amounts of information stored in documents necessitate efficient ways to extract and utilize this information. This project, titled "**Smart Question Answering System on Document**", aims to leverage advanced natural language processing (NLP) techniques to facilitate the extraction of meaningful insights from textual data contained within PDF files. The project encompasses two primary features: a question-answering system and a multiple-choice question (MCQ) generation system. The question-answering system integrates three different NLP models—**Word2Vec**, **GloVe**, and **BERT**—allowing users to interact with the text by asking questions related to the content. This functionality empowers users to select the model that best suits their needs, ensuring optimal performance for different types of queries. On the other hand, the MCQ generation feature utilizes **Sense2Vec** to create relevant and contextually appropriate multiple-choice questions from the text, which can be particularly beneficial in educational settings for creating quizzes and tests.

We use the following parameters for performance analysis of the validation and test data:

- **Accuracy:** It is a performance measure that finds out the ratio between the correct predictions to the total observations.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{False Negative} + \text{True Negative}} \quad (1)$$

- **Precision:** It is the ratio of the observations that were predicted as positive correctly to the total number of positive observations that were predicted.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \quad (2)$$

- **Recall:** It is also called as sensitivity as it focuses on the total number of actual hits found.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \quad (3)$$

- **F1 score:** It takes the weighted average of both precision and recall. To calculate this score both false positives and false negatives are taken into account.

$$\text{F1 score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

## **Agenda of Question Answering**

As early as 2002 a group of researchers<sup>3</sup> wrote a roadmap of the research in the field of question answering. They also identified the issues related to question answering.

The following discussion is based on the issues<sup>4</sup> they identified during their research.

1. Question classes
2. Question processing
3. Context and QA
4. Data sources for QA
5. Answer extraction
6. Answer formulation
7. Real time question answering

## **Question Classes**

A question may belong to different category and depending on its category we require different strategies to answer the question. We may have a line of attack for a category of factoid questions, which on the other hand will not work for questions that require deeper understanding of facts. We need a profound understanding of what category a question belongs.

Example:

### **Recall Question or Factoid Questions seek for Fact:**

Question: Who is also known as “chacha”, and was born on 14th November?

However questions like:

Question: Why is sky blue in color?

Requires understanding of not only facts but we must have knowledge of Scattering of Light.

As we will discuss later, questions can be classified based on its form in hierarchy.

We discuss such classifications in Chapter dedicated to bloom’s taxonomy.

## **Question Processing**

Same question may be asked in different forms. We may ask it in interrogative way or the assertive way. We need to understand the semantics of the question. We need

to recognize what the question is asking for before proceeding to answer the question itself. The practice to understand the question is termed as Question Processing.

We are seeking same information using different forms of the question:

**Interrogative:**

Question: What is the capital city of India?

Assertive:

Question: Tell me the name of city which is capital of India?

**Context and QA**

Questions are always asked in a context. Questions are rarely asked in universal context. We are required to have a knowledge of context before proceeding to answer a question.

**Example :**

Question: Where is Taj?

For a person in America:

He is interested in finding the location of Taj Mahal in Agra.

For a person who just reached Mumbai:

He is interested in finding Taj Hotel.

**Data Sources and QA**

Before we can answer questions we require sources which are relevant and exhaustive. We require a data source which will act as the base for all the information required for answering the questions. It may be collection documents. It can be the whole web which we can search for. It can be a database from where we can get the answers for structured queries.

**Answer Extraction**

Depending on the question we may want to extract specific type of information from the data sources. Reliant on the complexity of the question we would like to know the expectation of user.

Example:

**Extraction of a Name:**

Question: Who is eldest brother among Pandavas?

Extraction of time or date:

Question: When was first battle of panipat fought?

Extraction of Place:

Question: Where was Gandhi born?

**Answer Formulation**

Simple extraction may be enough for certain questions. We may want the partial answers to be extracted from various sources and combine them. At the same time, we want the results of the QA System to be as natural as possible. For generating answers we require to generate answers which is termed as answer formulation.

**Real time question answering**

We need to answer questions, even the complex question must answered in matter of seconds. People would not like to wait for hours in front of computer to get answers to questions. Watson for instance when played Jeopardy was able to answer to in average of 3 seconds. We need to develop architecture such that the end product is a real time system

**1.1.1 How the Project Works**

The project is designed to provide users with an intuitive and interactive experience in extracting information from PDF documents. The user journey can be outlined in the following steps:

**Model Selection:**

Users begin by visiting the home page of the application, where they are presented with options to choose among three different NLP models: Word2Vec, GloVe, and BERT.

Each model has its unique strengths and capabilities in understanding and generating human-like text responses.

**PDF Upload:**

Upon selecting a model, users are directed to an upload page where they can upload their PDF document.

The system ensures that the uploaded file is processed securely and stored temporarily for analysis.

### **NLP Processing:**

Once the PDF is uploaded, the selected NLP model processes the document's text.

The text is extracted, cleaned, and converted into a format suitable for analysis.

For the question-answering feature, the system uses the chosen model to embed the text and prepare it for interactive querying.

For the MCQ generation feature, Sense2Vec analyzes the text to create relevant and contextually appropriate multiple-choice questions.

### **Interactive Querying:**

Users can ask questions related to the content of the uploaded PDF.

The system uses the embeddings generated by the selected model to find the most relevant answers within the document.

The answers are displayed in real-time, allowing users to gain insights from the document quickly and efficiently.

### **MCQ Generation:**

Users can opt to generate multiple-choice questions from the uploaded PDF.

Sense2Vec analyzes the document and creates MCQs that are displayed to the user.

This feature is particularly beneficial for educators and content creators who need to create quizzes and assessments based on the document content.

## **1.1.2 Problems Solved**

The "Smart Question Answering System on Document" addresses several key challenges:

### **Information Extraction:**

Manually searching through lengthy PDF documents to find specific information can be time-consuming and inefficient. This project automates the process, allowing users to query documents interactively and receive precise answers instantly.

### **Content Accessibility:**

Many important documents are stored in PDF format, which can be difficult to search and navigate. By converting PDF text into a queryable format, the project enhances the accessibility of document content.

### **Educational Content Creation:**

Creating educational materials such as quizzes and tests from documents is a labor-intensive task. The MCQ generation feature automates this process, saving time and effort for educators.

#### **1.1.3 Approaches**

##### **Word2Vec:**

Word2Vec is a model that generates word embeddings by training on large corpora of text. It captures the contextual meaning of words and can be used to find similarities between words and phrases.

##### **GloVe (Global Vectors for Word Representation):**

GloVe is a model that creates word embeddings based on word co-occurrence statistics from a corpus. It is effective in capturing the global context of words and is used to find semantically similar words.

##### **BERT (Bidirectional Encoder Representations from Transformers):**

BERT is a state-of-the-art transformer model that captures bidirectional context in text. It is highly effective for a variety of NLP tasks, including question answering, due to its deep understanding of language context.

##### **Sense2Vec:**

Sense2Vec is an extension of Word2Vec that captures different senses of words based on their context. It is particularly useful for generating contextually appropriate multiple-choice questions from text.

## **1.2 Research Objectives**

The pivotal research objectives of this project revolve around the conception and realization of an advanced Question-Answering System, tailored to navigate the complexities inherent in processing PDF documents and delivering pinpoint responses to user queries. Central to this pursuit is the cultivation of a versatile environment accommodating a spectrum of Natural Language Processing (NLP) models, including but not limited to Word2Vec, GloVe, and BERT, thus empowering users with the freedom to discern and leverage the most apt model for their specific needs. Beyond the mere mechanics of implementation, lies a profound inquiry into the efficacy and applicability of these NLP methodologies within the domain of document-based question answering. This endeavor entails an exhaustive evaluation encompassing key metrics such as accuracy, relevance, and contextual fidelity, thereby furnishing invaluable insights into the strengths and limitations of each model. Moreover, the project endeavors to catalyze innovation through the integration of an automated Multiple-

Choice Question (MCQ) generation feature, driven by the sophisticated Sense2Vec model. This innovative addition not only promises to streamline the creation of assessment materials but also underscores a broader reflection on the transformative potential of AI-driven pedagogical tools in reshaping the landscape of modern education..

### 1.2.1 Model Research and Contributions

MODEL	DESCRIPTION	ACCURACY & CONTRIBUTION	USE CASE
<b>Word2vec</b>	Word2Vec is a shallow, two-layer neural network that processes text by converting words into high-dimensional vectors. It captures semantic relationships between words based on their context in a corpus.	<b>Accuracy:</b> Provides a moderate level of accuracy in answering questions but can struggle with complex queries due to its reliance on word context without understanding the overall document structure. <b>Contribution:</b> Useful for understanding word relationships and generating quick, context-based responses.	Best for straightforward queries that rely on word associations and simpler contexts.
<b>GloVe</b>	GloVe (Global Vectors for Word Representation) is an unsupervised learning algorithm for obtaining vector representations for words. Unlike Word2Vec, it focuses on aggregating global word-word co-occurrence statistics from a corpus.	<b>Accuracy :</b> Offers a balanced performance, better than Word2Vec for capturing global context, but still limited in understanding nuanced queries and document structures. <b>Contribution:</b> Effective in scenarios where global word relationships are crucial.	Suitable for questions requiring a broader contextual understanding but not deep document comprehension.
<b>BERT</b>	BERT (Bidirectional Encoder Representations from Transformers) is a transformer-based model that understands the context of a word in both directions, making it highly effective for various NLP tasks.	<b>Accuracy:</b> Provides the highest accuracy among the models, excelling in understanding complex queries and maintaining context across long passages. <b>Contribution:</b> Significantly improves the relevance and precision of answers due to its deep contextual understanding.	Ideal for detailed, complex queries and documents requiring deep comprehension and context retention.

### **1.3 Literature Review**

This literature review explores various research efforts and systems developed in the field of question answering (QA) and related natural language processing (NLP) techniques, highlighting the evolution and advancements made over the years.

#### **CRQA: Crowd-Powered Real-Time Automatic Question Answering System**

Agichtein and Savenkov (2016) introduced CRQA, a system that leverages crowd-sourcing to provide real-time, accurate answers to user questions. This approach combines machine learning with human input to enhance the reliability and relevance of answers.

#### **Answer Extraction and Passage Retrieval for Question Answering Systems**

Ahmed and Babu (2016) focused on improving answer extraction and passage retrieval for QA systems. Their work emphasized the importance of selecting relevant text passages from documents to provide precise answers, enhancing the performance of QA systems significantly.

#### **MASQUE/SQL – Natural Language Query Interface for Relational Databases**

Androutsopoulos et al. (1993) developed MASQUE/SQL, an efficient and portable natural language query interface for relational databases. This system allows users to query databases using natural language, making data retrieval more intuitive and accessible.

#### **PiQAS: Pisa Question Answering System**

Attardi et al. (2001) presented PiQAS, a system designed for the Text REtrieval Conference (TREC). PiQAS focuses on providing accurate answers by processing large text corpora, demonstrating significant improvements in handling diverse question types.

#### **Bengali Question Classification for Developing QA System**

Banerjee and Bandyopadhyay (2012) tackled the issue of question classification in Bengali, a crucial step in developing effective QA systems for this language. Their research contributes to the broader goal of creating multilingual QA systems.

#### **Question Classification for a Croatian QA System**

Basic et al. (2011) developed a QA system for Croatian, focusing on question classification. This research highlights the challenges and solutions in adapting QA technology to less-resourced languages.



### **YodaQA: Modeling the Question Answering Task**

Baudiš and Šedivy (2015) introduced YodaQA, which models the QA task to provide contextually accurate answers. This system utilizes machine learning to understand and answer questions more effectively, contributing to advancements in automated QA.

### **Cooperative Question Answering in Restricted Domains: WEBCOOP**

Benamara (2004) explored cooperative question answering in restricted domains with the WEBCOOP system. This research emphasizes the importance of domain-specific knowledge in improving QA accuracy.

### **Preference Reasoning in Advanced Question Answering Systems**

Benamara and Kaci (2006) discussed the role of preference reasoning in QA systems. Their work illustrates how incorporating user preferences can enhance the relevance and satisfaction of QA systems.

### **Bridging the Lexical Chasm: Statistical Approaches to Answer-Finding**

Berger et al. (2000) addressed the lexical gap between questions and answers using statistical methods. This research is foundational in developing algorithms that better match user queries with relevant answers.

### **Building User Interest Profiles Using DBpedia in a QA System**

Bergeron et al. (2016) proposed using DBpedia to build user interest profiles for QA systems. This approach personalizes the QA experience by tailoring answers based on user interests.

### **Multimedia Question Answering System**

Bharat et al. (2016) introduced a multimedia QA system, extending traditional text-based QA to include multimedia elements. This innovation broadens the applicability and utility of QA systems.

### **Hybrid Question Answering System for Multiple Choice Question (MCQ)**

Bhaskar et al. (2013) developed a hybrid QA system specifically for MCQs. Their approach integrates various QA techniques to provide accurate answers for multiple-choice formats.

## 1.4 Theoretical Foundations

### 1.4.1 GloVe Embeddings and its role

GloVe, short for Global Vectors for Word Representation, stands as a seminal milestone in the field of natural language processing (NLP) and word embeddings. Conceptualized and developed by Jeffrey Pennington, Richard Socher, and Christopher D. Manning at Stanford University, GloVe presents a groundbreaking approach to representing words as dense vectors in a continuous vector space. At its core, GloVe seeks to capture the intricate relationships between words by analyzing their co-occurrence statistics within large text corpora. Unlike traditional methods that rely solely on local context, such as Word2Vec's skip-gram and continuous bag-of-words (CBOW) models, the theoretical foundation of GloVe is rooted in the principle of word-word co-occurrence matrices. By constructing a matrix where each element reflects the frequency of co-occurrence between two words, GloVe effectively encodes the distributional semantics of the vocabulary. One of the key strengths of GloVe lies in its ability to capture both syntactic and semantic relationships between words. By analyzing the co-occurrence patterns of words across the entire corpus, GloVe embeddings exhibit a remarkable ability to encode semantic similarity and analogy relationships. This makes GloVe particularly well-suited for a wide range of NLP tasks, including but not limited to word similarity measurement, sentiment analysis, and machine translation. Moreover, GloVe embeddings possess several desirable properties that set them apart from other word embedding techniques. Additionally, GloVe embeddings exhibit linear substructure, meaning that vector differences can capture meaningful linguistic relationships, such as analogies (e.g., "king" - "man" + "woman"  $\approx$  "queen")

Here's how GloVe embeddings are typically utilized in question-answering systems:

#### ***Word Representation:***

GloVe embeddings provide a compact and meaningful representation of words based on their co-occurrence statistics within a corpus.

#### ***Semantic Similarity:***

GloVe embeddings enable the computation of semantic similarity between words by measuring the cosine similarity or Euclidean distance between their embedding vectors.

#### ***Contextual Understanding:***

By capturing the semantic relationships between words, GloVe embeddings help the question-answering system understand the context in which words are used within the document.

#### ***Answer Selection:***

Once the relevant passages or sentences are identified, GloVe embeddings can be used to compare the similarity between words in the question and words in the candidate answers. This allows the system to rank potential answers based on their semantic relevance to the question, ultimately selecting the most appropriate answer.

### 1.4.2 Word2Vec and its role

Word2vec is a technique in natural language processing (NLP) for obtaining vector representations of words. These vectors capture information about the meaning of the word based on the surrounding words. The word2vec algorithm estimates these representations by modeling text in a large corpus. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence. Word2vec was developed by Tomáš Mikolov and colleagues at Google and published in 2013. Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word2vec can utilize either of two model architectures to produce these distributed representations of words: Continuous Bag-Of-Words (CBOW) or continuously sliding skip-gram. In both architectures, word2vec considers both individual words and a sliding context window as it iterates over the corpus. The CBOW can be viewed as a ‘fill in the blank’ task, where the word embedding represents the way the word influences the relative probabilities of other words in the context window. Words which are semantically similar should influence these probabilities in similar ways, because semantically similar words should be used in similar contexts. The order of context words does not influence prediction (bag-of-words assumption). After the model has trained, the learned word embeddings are positioned in the vector space such that words that share common contexts in the corpus — that is, words that are semantically and syntactically similar — are located close to one another in the space. More dissimilar words are located farther from one another in the space.

Here's how *Word2vec* embeddings are typically utilized in question-answering systems:

#### ***Word Representation:***

In a question-answering system, words from the question and relevant documents are mapped to their respective Word2Vec embedding vectors.

#### ***Semantic Similarity:***

In a question-answering scenario, the system can leverage Word2Vec embeddings to identify words in the question that are semantically related to words in the document context, aiding in the retrieval of relevant information.

#### ***Contextual Understanding:***

By learning from the surrounding context of words in the corpus, Word2Vec embeddings help the question-answering system understand the context in which words are used within the document.

#### ***Answer Selection:***

Once the relevant passages or sentences are identified, Word2Vec embeddings can be used to compare the similarity between words in the question and words in the candidate answers. This allows the system to rank potential answers based on their semantic relevance to the question, facilitating the selection of the most appropriate answer.

### 1.4.3 BERT and its role

Let's take a look at how BERT works, covering the technology behind the model, how it's trained, and how it processes data. Recurrent and convolutional neural networks use sequential computation to generate predictions. That is, they can predict which word will follow a sequence of given words once trained on huge datasets. In that sense, they were considered unidirectional or context-free algorithms. By contrast, transformer-powered models like BERT, which are also based on the encoder-decoder architecture, are bidirectional because they predict words based on the previous words and the following words. This is achieved through the self-attention mechanism, a layer that is incorporated in both the encoder and the decoder. The goal of the attention layer is to capture the contextual relationships existing between different words in the input sentence. Nowadays, there are many versions of pre-trained BERT, but in the original paper, Google trained two versions of BERT: *BERTbase* and *BERTlarge* with different neural architectures. In essence, *BERTbase* was developed with 12 transformer layers, 12 attention layers, and 110 million parameters, while *BERTlarge* used 24 transformer layers, 16 attention layers, and 340 million parameters. As expected, *BERTlarge* outperformed its smaller brother in accuracy tests.

#### Pre-training and fine-tuning

Transformers are trained from scratch on a huge corpus of data, following a time-consuming and expensive process (that only a limited group of companies, including Google, can afford). In the case of BERT, it was pre-trained over four days on Wikipedia (~2.5B words) and Google's BooksCorpus (~800M words). This allows the model to acquire knowledge not only in English but also in many other languages from around the world. To optimize the training process, Google developed new hardware, the so-called TPU (Tensor Processing Unit), specifically designed for machine learning tasks. To avoid unnecessary and costly interactions in the training process, Google researchers used transfer learning techniques to separate the (pre)training phase from the fine-tuning phase. This allows developers to choose pre-trained models, refine the input-output pair data of the target task, and retrain the head of the pre-trained model by using domain-specific data.

Consider case of Bert (bert-base). It will produce a 768 dim vector output corresponding to each token. In downstream tasks like Question-Answering we will have 2 linear layers - one for start position and another for end position (start token classifier and end token classifier). We have separate weights for each of them. During finetuning they are trained together.

During inference for every token in the text, we feed its final embedding into the start token classifier as well as end token classifier. For each token internally a dot product occurs with start token vector and produce logits corresponding to that token. Similarly for the end token classifier as well. Thus model will produce start logits and end logits corresponding to all the input tokens.

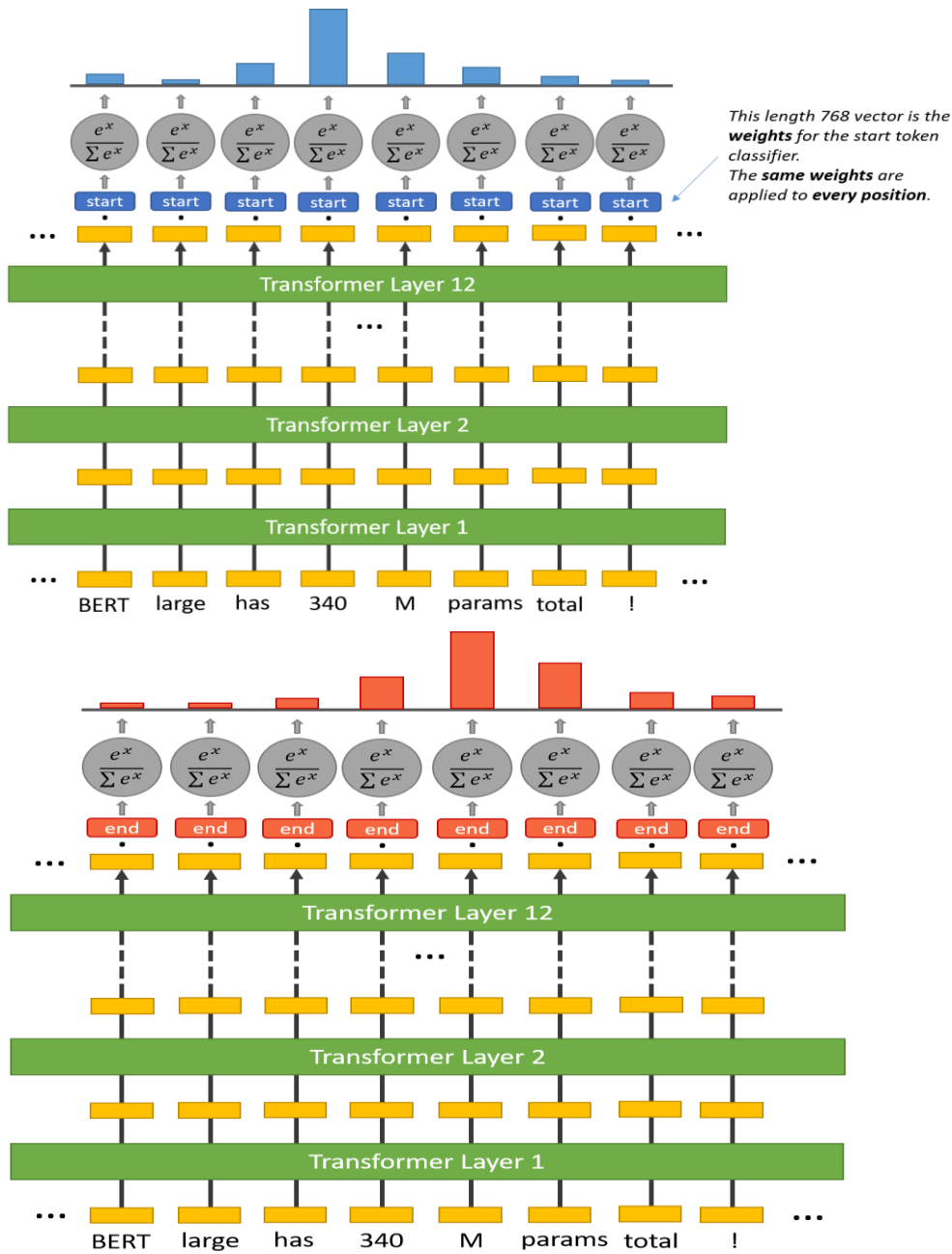


Figure-1

#### 1.4.4 Sense2vec and its role

The incorporation of Sense2Vec in generating Multiple-Choice Questions (MCQs) represents a significant enhancement in the automation of educational assessment tools. Sense2Vec, an extension of the original Word2Vec model, augments word embeddings by incorporating part-of-speech tags and disambiguating between different senses of words. This additional contextual information allows for more precise and contextually relevant embeddings, which are particularly useful in generating meaningful and varied MCQs.

#### How Sense2Vec Works:

### *Embedding Contextualization:*

Sense2Vec generates embeddings that not only consider the semantic meaning of words but also account for their parts of speech and different senses. For example, the word "bank" as a noun (financial institution) and "bank" as a verb (to incline) would have distinct embeddings.

### *Data Preparation:*

To utilize Sense2Vec for MCQ generation, a corpus of text is first processed to extract key concepts and sentences. The model identifies potential answer-worthy sentences and concepts based on their contextual importance and semantic richness.

### *MCQ Structure:*

An MCQ typically consists of a question stem and several answer options, including one correct answer and several distractors (incorrect but plausible answers). Sense2Vec aids in creating these components by:

**Generating the Question Stem:** Identifying significant sentences or information nuggets within the text that can be converted into questions.

**Selecting the Correct Answer:** Using embeddings to pinpoint the most relevant and contextually appropriate answer from the text.

**Generating Distractors:** Creating plausible incorrect options by identifying words or phrases with similar embeddings but different meanings or contexts.

### **Steps in Sense2Vec-Powered MCQ Generation:**

*Embedding Generation:* Sense2Vec generates embeddings for the key terms and sentences. Each word is disambiguated based on its context, resulting in more accurate and context-sensitive embeddings.

*Question Creation:* Identify Key Sentences: Select sentences that contain information suitable for forming questions.

*Generate Question Stems:* Transform the selected sentences into question stems. This can involve rephrasing or focusing on specific aspects of the information.

*Correct Answer Identification:* Using the embeddings, the system identifies the most appropriate answer within the context of the question stem.

*Distractor Generation:* Sense2Vec helps generate distractors by finding words or phrases with similar embeddings that are contextually plausible but incorrect. These distractors are carefully chosen to be challenging yet distinguishable from the correct answer.

*Validation and Refinement:* The generated MCQs are validated to ensure they meet quality standards. This may involve manual review or additional automated checks to ensure the questions are clear, accurate, and pedagogically sound.

**Example:**

Consider a text discussing the financial crisis:

*Original Sentence:*

"The 2008 financial crisis was precipitated by the collapse of Lehman Brothers, which had significant implications for global markets."

*Generated MCQ:*

Question: What event precipitated the 2008 financial crisis?

- A. The collapse of Lehman Brothers (Correct Answer)
- B. The dot-com bubble burst
- C. The Greek debt crisis
- D. The fall of Enron

In this example, Sense2Vec helps to identify the key event (collapse of Lehman Brothers) and generates distractors (dot-com bubble burst, Greek debt crisis, fall of Enron) that are plausible but incorrect within the given context.

**Advantages of Using Sense2Vec for MCQ Generation:**

*Contextual Accuracy:* By incorporating contextual and disambiguated word embeddings, Sense2Vec ensures that the generated MCQs are contextually accurate and relevant to the source material.

*Diversity in Questions:* Sense2Vec's ability to understand different word senses allows for the generation of a diverse set of questions and answers, enhancing the assessment's comprehensiveness.

*Efficiency:* Automating the MCQ generation process significantly reduces the time and effort required for educators to create high-quality assessment materials, allowing them to focus on more strategic tasks.

*Enhanced Learning:* The use of contextually relevant and challenging MCQs promotes deeper learning and better assessment of students' understanding of the material.

# **CHAPTER-2**

## **System Block Diagram**



## 2. Block Diagram

### 2.1 System Block Diagram Overview

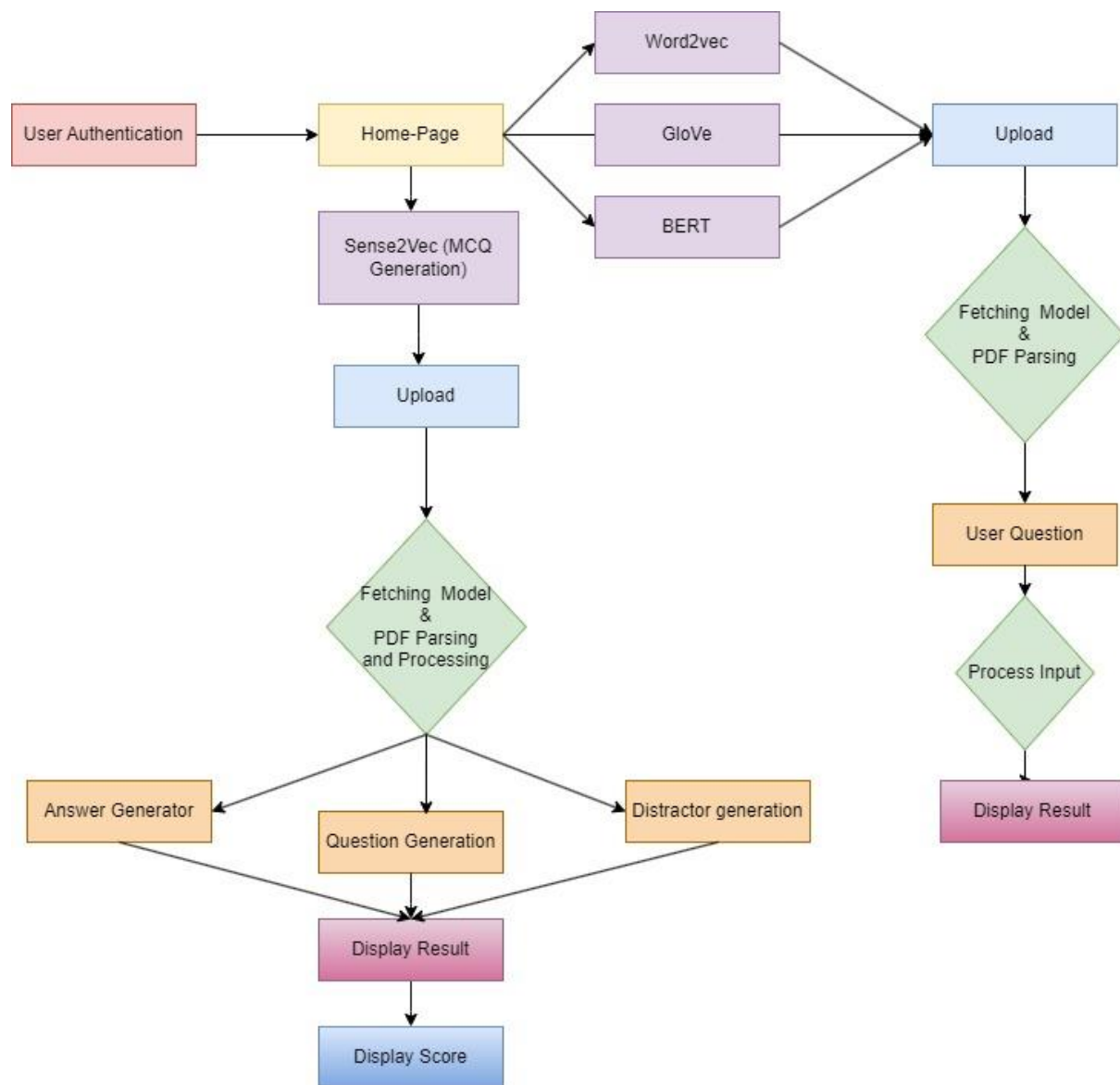


Figure-2

The system comprises multiple integrated modules designed for a comprehensive question-answering platform and automatic multiple-choice question (MCQ) generation. Users start by logging in or registering, then navigate to the homepage, which provides access to the QA and MCQ generation modules. The QA module processes PDF documents and answers user queries using advanced NLP models like BERT, Word2Vec, and GloVe. Users upload documents, input questions, and receive answers with relevant context. The MCQ generation module employs the Sense2Vec model to generate contextually relevant MCQs from uploaded PDFs. Users can review and adjust the generated MCQs before finalizing them. The system features a user-friendly interface and secure backend, with RESTful APIs facilitating frontend-backend communication.

### 2.1.1 System Block Diagram of Word2vec

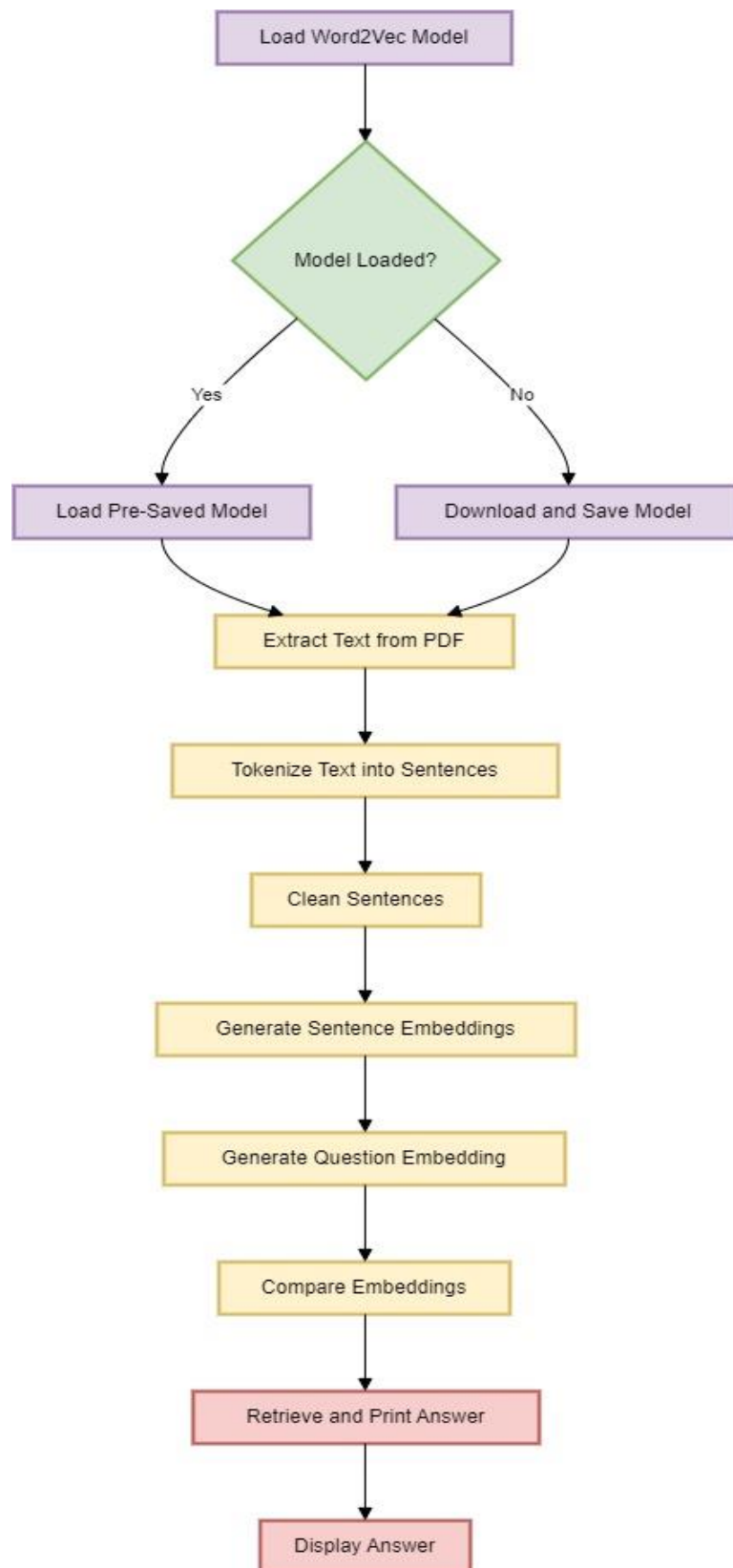


Figure-3

The Word2Vec-based question-answering system is designed to facilitate efficient extraction and retrieval of relevant information from PDF documents through natural language processing (NLP) techniques. The process initiates with an attempt to load a pre-saved Word2Vec model, which is essential for embedding generation. If the model is not available locally, the system seamlessly downloads the 'word2vec-google-news-300' model, one of the most comprehensive pre-trained models available, and saves it for future use. This step ensures that the system is always equipped with the necessary tools for high-quality word embeddings.

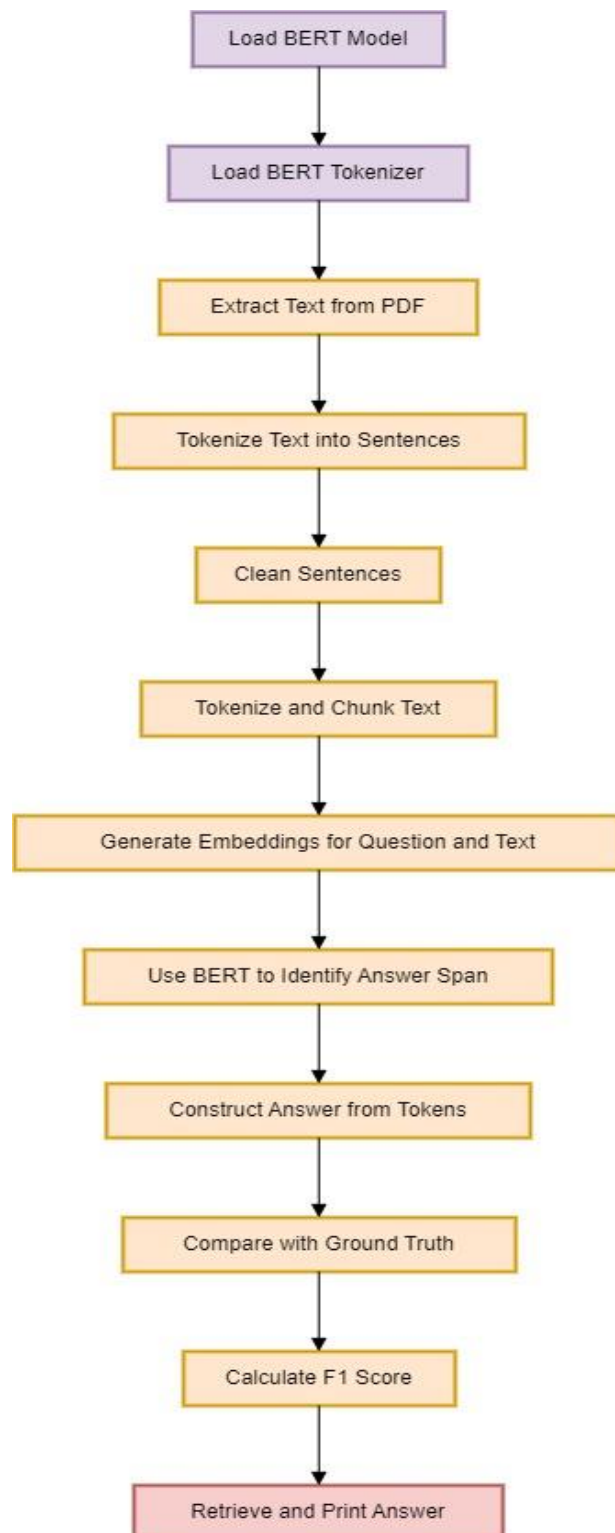
Following the model setup, the system proceeds to extract text from the provided PDF file using a specialized function, `pdf_extract`. This function is capable of handling various PDF formats and extracting text accurately, which is crucial for subsequent processing steps. Once the text is extracted, it is tokenized into individual sentences using NLTK's `sent_tokenize` function. This tokenization process breaks down the continuous text into manageable sentence units, which are then cleaned by removing stopwords and performing further preprocessing. Cleaning the sentences ensures that the embeddings generated are not skewed by irrelevant words, thereby improving the overall accuracy of the model.

The next phase involves generating sentence embeddings. For each cleaned sentence, the system generates embeddings using the loaded Word2Vec model. These embeddings are numerical representations of the sentences that capture their semantic meaning, allowing for effective comparison and retrieval of information. Concurrently, the system generates an embedding for the input question, ensuring that the query is transformed into the same numerical space as the sentences in the document.

Once both the sentence embeddings and the question embedding are generated, the system moves to the retrieval phase. It compares the question embedding with the sentence embeddings to identify the most relevant sentence that answers the query. This comparison is typically done using similarity measures such as cosine similarity, which quantifies the closeness of the embeddings in the vector space. The sentence with the highest similarity score is retrieved as the most relevant answer to the input question.

Finally, the system prints and displays this most relevant sentence, providing the user with a concise and accurate response to their query. This entire process, from model loading to answer retrieval, is designed to be seamless and efficient, leveraging advanced NLP techniques to handle complex documents and queries. By integrating robust pre-trained models like Word2Vec and employing meticulous text processing methods, the system ensures high precision and relevance in its answers, making it a powerful tool for extracting information from extensive textual data.

### 2.1.2 System Block Diagram of BERT Question Answering



*Figure-4*

The BERT-based question-answering system initiates by loading the pre-trained BERT model and its tokenizer, followed by extracting text from the provided PDF using pdfplumber. The

extracted text undergoes tokenization and cleaning, including lowercase conversion, punctuation removal, and optional stopwords elimination. Subsequently, the text is tokenized and chunked into BERT-compatible input tokens, enabling the generation of embeddings for both the input question and the text chunks. Utilizing BERT's capabilities, the system predicts the start and end indices of the answer span within the text, constructs the answer by concatenating tokens, and handles special subword tokens as needed. Optionally, the generated answer can be compared with the ground truth to assess model accuracy, calculating the F1 score for evaluation. Finally, the retrieved answer is printed, providing a comprehensive solution to the question posed.

### 2.1.2 System Block Diagram of MCQ generation on Sense2vec model

The feature implemented in the QuestionGeneration class is a sophisticated question generation system that goes beyond simple extraction by incorporating advanced techniques such as named entity recognition (NER) and TF-IDF scoring. Here's an elaboration on the key components of this feature:

**Named Entity Recognition (NER):** The system utilizes NER to identify entities within the document. These entities serve as potential candidates for forming questions. By recognizing entities such as people, organizations, locations, dates, and more, the system can generate questions that focus on specific topics or details mentioned in the document.

**TF-IDF Scoring:** TF-IDF (Term Frequency-Inverse Document Frequency) scoring is employed to assess the importance of words within the document. By calculating the TF-IDF scores for each word, the system can identify keywords that are significant and likely to be relevant for generating questions. This scoring mechanism ensures that questions are based on the most informative and distinctive terms in the document.

**Question Formulation:** Once candidate keywords are identified through NER and ranked using TF-IDF scores, the system formulates questions based on these keywords. It constructs questions that prompt for information related to the identified entities or important terms. The questions are designed to be concise, relevant, and comprehensible, facilitating effective comprehension and engagement with the document's content.

**Incorrect Answer Generation:** In addition to generating questions, the system also generates incorrect answer options (distractors) for each question. It achieves this by leveraging the Word2Vec model to find similar words to the correct answer. These distractors add complexity to the question and require students to critically evaluate the options, enhancing the learning experience by promoting deeper understanding and analysis.

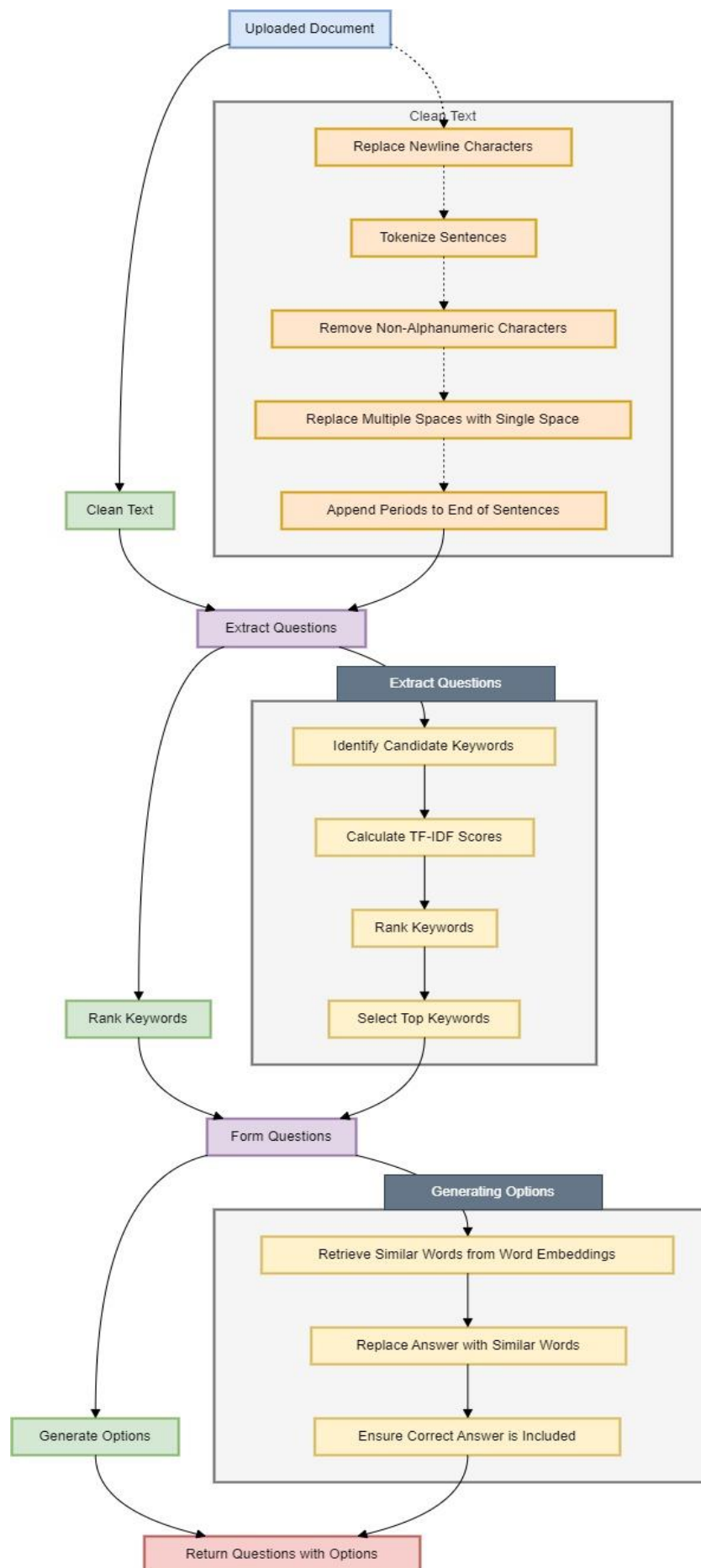


Figure-5

# **CHAPTER-3**

## **Project Methodology**

### **3. Methodology**

#### **3.1 Objective Identification**

The primary objectives of designing the Personalized Learning Assessment System (PLAS) are as follows:

##### **1. Personalized Learning Experience**

*Tailored Assessments:* Create assessments that are customized to the individual learning pace, style, and progress of each student. This ensures that the difficulty level of questions matches the learner's current understanding and capabilities.

*Adaptive Learning Pathways:* Develop dynamic learning pathways that adapt based on student performance, providing personalized recommendations for further study and practice.

##### **2. Efficient Question Generation**

*Automated Question Generation:* Utilize advanced Natural Language Processing (NLP) models to automatically generate high-quality, diverse questions from the educational content. This reduces the manual effort required by educators to create assessments.

*Variety of Question Types:* Ensure the system can generate various types of questions, such as multiple-choice, fill-in-the-blank, true/false, and short answer questions, to evaluate different levels of cognitive skills.

##### **3. Accurate Answer Validation**

*Correct Answer Identification:* Implement models that accurately identify and validate the correct answers for generated questions. This ensures the reliability and validity of the assessments.

*Plausible Distractor Generation:* Generate contextually relevant and challenging distractors (incorrect answers) that make multiple-choice questions more effective in assessing student knowledge.

##### **4. Comprehensive Performance Tracking**

*Detailed Performance Analytics:* Provide detailed analytics and reports on student performance, highlighting strengths, weaknesses, and areas needing improvement. This helps teachers and students make informed decisions about future learning strategies.

*Progress Monitoring:* Track student progress over time, allowing educators to see how students are improving and adjust instruction accordingly.

##### **5. Enhanced User Engagement**

*Interactive User Interface:* Develop an intuitive and user-friendly interface that engages students and encourages active participation in assessments.

*Real-Time Feedback:* Provide immediate feedback to students on their performance, helping them understand their mistakes and learn from them.



## **6. Scalability and Accessibility**

*Scalable System Architecture:* Design a system architecture that can handle a large number of users simultaneously, ensuring smooth performance even with high usage.

*Accessibility:* Ensure the system is accessible to all students, including those with disabilities, by incorporating features such as text-to-speech, adjustable font sizes, and screen reader compatibility.

## **7. Support for Diverse Educational Content**

*Subject and Topic Coverage:* Ensure the system can handle a wide range of subjects and topics, catering to different educational levels and curricula.

*Content Integration:* Allow easy integration of various educational content formats, such as text, images, and videos, to enhance the richness of the learning material.

## **8. Teacher and Administrator Tools**

*Customization Options:* Provide teachers and administrators with tools to customize assessments, set parameters, and review student performance.

*Resource Management:* Implement features for managing educational resources, such as creating and organizing question banks, sharing assessments, and collaborating with other educators.

## **3.2 Technical Platform**

The development of the Smart Question Answering System leverages a comprehensive and robust technical platform, integrating several advanced technologies and tools to create a sophisticated and efficient solution. Below is an overview of the key technologies used in this project:

### **Backend Technologies**

#### **Flask:**

Flask is a lightweight web framework for Python that provides the necessary tools to build web applications. It is used to develop the server-side logic, manage routes, handle HTTP requests, and render templates.

#### **Python:**

Python is the primary programming language used for developing the backend of this system. Its rich ecosystem and libraries facilitate the implementation of various natural language processing (NLP) tasks.

#### **NLTK (Natural Language Toolkit):**

NLTK is a powerful library in Python for working with human language data. It is used for text preprocessing tasks such as tokenization, stemming, lemmatization, and stopword removal.

### **TF-IDF (Term Frequency-Inverse Document Frequency):**

TF-IDF is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents. It is employed in the system to extract relevant features from text data, aiding in the question-answering process.

### **Gensim:**

Gensim is a library for topic modeling and document similarity analysis. It supports various NLP tasks and is particularly known for its implementation of Word2Vec.

### **Word2Vec:**

Word2Vec is a neural network-based model that generates word embeddings, which capture semantic relationships between words. It is utilized in the project to understand the context of user queries and document content.

### **GloVe (Global Vectors for Word Representation):**

GloVe is another word embedding technique that generates vectors based on word co-occurrence statistics. It enhances the system's ability to interpret and respond to user queries accurately.

### **BERT (Bidirectional Encoder Representations from Transformers) Question Answering:**

BERT is a state-of-the-art transformer-based model designed by Google. It excels in understanding the context of words in a sentence. The BERT Question Answering model is implemented to provide highly accurate and contextually relevant answers to user queries.

### **Frontend Technologies**

#### **HTML (HyperText Markup Language):**

HTML is used to create the structure of the web pages, defining the layout and elements such as forms, buttons, and text areas.

#### **CSS (Cascading Style Sheets):**

CSS is used for styling the web pages, ensuring a visually appealing and user-friendly interface. It controls the layout, colors, fonts, and overall aesthetics of the application.

### **JavaScript:**

JavaScript is used to add interactivity to the web pages, enabling dynamic content updates, form validation, and other client-side functionalities.

### **MCQ Generation**

#### **Sense2Vec:**

Sense2Vec is an extension of Word2Vec that captures not only the semantics of words but also their senses or meanings based on context. It is employed to generate multiple-choice questions (MCQs) from the uploaded documents, assisting in the creation of educational assessment materials.

### **System Flow**

#### **User Interaction:**

Users begin by logging into the system and navigating to the homepage.

From the homepage, users can choose between two primary functionalities: QA (Question Answering) and MCQ Generation.

#### **Question Answering (QA):**

Users select the QA model they wish to use (Word2Vec, GloVe, or BERT) and upload their PDF documents.

The system processes the uploaded document using the selected model to create a searchable knowledge base.

Users can then enter their questions, and the system uses the chosen model to generate and display accurate answers.

#### **MCQ Generation:**

Users navigate to the MCQ generation section and upload their PDF documents.

The system processes the document using Sense2Vec to generate relevant MCQs.

The generated MCQs are displayed on a dedicated results page, providing users with ready-to-use assessment materials.

### 3.3 Data Processing

Data preprocessing is a critical step in the development of a smart question answering system, transforming raw data into a clean, structured, and meaningful format suitable for analysis and model training. The process begins with data collection from diverse and reliable sources, including textbooks, educational websites, scholarly articles, and online repositories. This data is often gathered in various formats such as text documents, PDFs. The next step is data cleaning, which involves removing noise like HTML tags, special characters, advertisements, and irrelevant metadata. Missing data is handled appropriately, either by filling in with placeholders or removing it entirely, and typographical errors are corrected to ensure consistency.

Text normalization follows, where all text is converted to lowercase to reduce vocabulary size and maintain uniformity. Punctuation is removed, contractions are expanded, and stemming and lemmatization are applied to reduce words to their root or base forms. The text is then tokenized, first into sentences and then into individual words, using tools like NLTK or spaCy. Stopwords, which are common words that do not contribute significant meaning, are identified and removed using predefined or custom lists.

Part-of-speech tagging and dependency parsing are employed to understand the grammatical relationships between words, enhancing the system's ability to comprehend context. Named entity recognition (NER) is used to identify and classify named entities such as people, organizations, and locations, ensuring consistency through entity normalization. Feature extraction techniques like TF-IDF vectorization and pre-trained word embeddings (e.g., Word2Vec, GloVe) convert the text into numerical formats, capturing semantic similarities. Advanced embeddings like BERT are used to capture contextual information and nuances in the text.

### 3.4 MCQ Generation

The process of question generation and answer validation in the smart question answering system involves several interconnected components aimed at extracting meaningful questions from textual documents and ensuring the accuracy of generated answers.

Firstly, the QuestionExtractor class is responsible for identifying candidate keywords and entities within the document using techniques like named entity recognition (NER) and TF-IDF scoring. These candidate keywords are ranked based on their relevance and importance in the document using TF-IDF scores. The class then forms questions by replacing these keywords with placeholders in corresponding sentences, ensuring that each question is contextually relevant and grammatically correct.

Secondly, the `IncorrectAnswerGenerator` class complements the question extraction process by generating incorrect answer options for each question. It utilizes word embeddings like GloVe to find semantically similar words to the correct answer, providing a diverse set of options that challenge the user's understanding of the topic. This step enhances the learning experience and ensures that the user comprehensively understands the material.

Finally, the `QuestionGeneration` class orchestrates the entire process, coordinating between question extraction and answer generation. It handles the preprocessing of the input document, ensures the cleanliness and uniformity of the text, and integrates the functionalities of the `QuestionExtractor` and `IncorrectAnswerGenerator` classes to produce a final set of questions with multiple answer options. Additionally, it provides debugging statements to facilitate the monitoring and improvement of the question generation process.

Overall, the question generation and answer validation process in the smart question answering system is a multi-step and interconnected pipeline that leverages natural language processing techniques, word embeddings, and machine learning models to produce high-quality questions and assess the user's understanding effectively.

### 3.5 System Architecture

System integration and interface design are crucial aspects of developing a smart question answering system that ensures seamless interaction between users and the underlying functionalities. This phase involves combining different modules, components, and data sources into a unified system and designing user interfaces that facilitate intuitive interaction and efficient utilization of the system's capabilities.

**Integration of Modules:** The first step in system integration is to integrate the various modules developed during the system's design and implementation phases. This includes modules for natural language processing, question generation, answer validation, and any other components necessary for the system's operation. Integration ensures that these modules communicate effectively and work together to achieve the system's objectives.

**Data Integration:** In addition to module integration, data integration plays a crucial role in ensuring that the system has access to all relevant data sources required for question answering. This may involve integrating external databases, knowledge graphs, or APIs to enrich the system's knowledge base and improve its ability to provide accurate answers.

**Interface Design:** Interface design focuses on creating user interfaces that are intuitive, user-friendly, and accessible to a wide range of users. This involves designing both graphical user

interfaces (GUIs) and command-line interfaces (CLIs) based on the system's target audience and use cases. The interface should provide users with easy access to the system's functionalities, such as entering questions, viewing answers, and navigating through the system's features.

**User Experience (UX) Design:** UX design is integral to interface design and involves optimizing the user experience to ensure that users can interact with the system efficiently and effectively. This includes considerations such as layout, navigation, visual design, responsiveness, and feedback mechanisms. The goal is to create an engaging and satisfying user experience that encourages users to interact with the system and achieve their goals seamlessly.

**Testing and Validation:** Once the system integration and interface design are complete, thorough testing and validation are essential to ensure that the system functions as intended and meets the requirements and expectations of its users. This involves conducting usability testing, functionality testing, and performance testing to identify and address any issues or deficiencies in the system.

### 3.6 Testing and Validation

Testing and validation are indispensable phases in the development lifecycle of any software system, including a smart question answering system. These processes ensure that the system functions correctly, meets user requirements, and delivers accurate and reliable results. The testing and validation phase encompasses several key activities:

**Unit Testing:** Unit testing involves testing individual components or units of code in isolation to verify that they perform as expected. For a question answering system, this might include testing functions responsible for natural language processing, data retrieval, answer generation, and validation.

**Integration Testing:** Integration testing focuses on testing the interactions between different modules or components of the system. It ensures that these components work together seamlessly to achieve the system's overall functionality. In the context of a question answering system, integration testing might involve testing how well the natural language processing module integrates with the answer generation module, for example.

**Functional Testing:** Functional testing evaluates the system's functionality against specified requirements. Test cases are designed to verify that the system performs all the functions it's

supposed to, such as accurately answering questions, handling different types of queries, and providing relevant responses.

**Performance Testing:** Performance testing assesses the system's performance under various conditions, such as different loads, concurrent users, or input data sizes. For a question answering system, performance testing might involve measuring response times, resource utilization, and scalability to ensure that the system can handle expected workloads efficiently.

**Usability Testing:** Usability testing focuses on evaluating the system's user interface and user experience. It involves gathering feedback from real users to assess the system's ease of use, intuitiveness, and effectiveness in meeting user needs. Usability testing helps identify any usability issues or areas for improvement in the system's interface and interaction flow.

**Validation Against Ground Truth:** In the case of a question answering system, validation against ground truth or known correct answers is essential. This involves comparing the system's answers to a set of predefined correct answers or reference data to verify their accuracy and reliability.

**Robustness Testing:** Robustness testing evaluates the system's ability to handle unexpected inputs, errors, or edge cases gracefully. It involves subjecting the system to various stressors or invalid inputs to assess its resilience and robustness in real-world scenarios.

**Security Testing:** Security testing assesses the system's resilience to security threats and vulnerabilities. It involves identifying and addressing potential security weaknesses, such as data breaches, unauthorized access, or injection attacks, to ensure the confidentiality, integrity, and availability of the system and its data.

# **CHAPTER-4**

## **Result and Discussions**

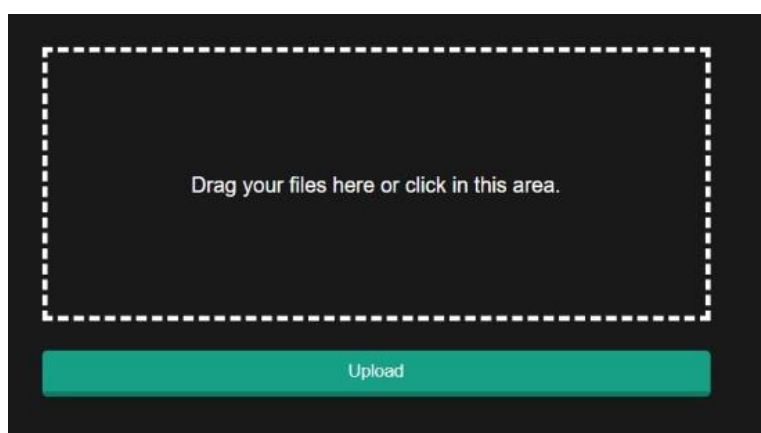


## 4.1 Calculating the performance

To comprehensively assess the performance of your question-answering model, several key steps need to be undertaken. Firstly, it's essential to establish clear evaluation metrics to gauge the model's effectiveness. Common metrics include **Exact Match (EM)** and **F1 Score**, which provide insights into the model's accuracy and ability to precisely answer questions. Next, gather a dataset containing ground truth answers, ensuring each question has a correct answer for comparison. Utilizing the start and end word scores generated by the model, predict answer spans for each question. Translate these scores into predicted start and end positions within the tokenized text. With predictions and ground truth answers in hand, compute the performance metrics. EM measures the percentage of questions where the predicted answer exactly matches the ground truth, offering a straightforward assessment of accuracy. Meanwhile, the F1 Score, which calculates the harmonic mean of precision and recall, provides a nuanced evaluation of the model's precision and completeness in identifying correct answers. After evaluating the model's performance, interpret the results to understand its strengths and weaknesses. High EM and F1 Scores suggest the model is proficient in accurately answering questions, while lower scores may indicate areas requiring improvement. Finally, iterate on the model's design, fine-tune parameters, adjust training data, or explore alternative architectures to enhance its effectiveness based on the evaluation outcomes. Through this iterative process, the model can evolve to achieve higher levels of accuracy and reliability in question-answering tasks.

### *Upload Section*

In the upload section, users are presented with a clean and intuitive interface where they can easily upload their PDF documents. The design prioritizes simplicity and accessibility, ensuring that users from various technical backgrounds can effortlessly interact with the system. Once a document is uploaded, the system processes the text using advanced NLP techniques, preparing it for question-answering and MCQ generation tasks.



*Figure-6*

### Question Answering (QA) Section

The QA section highlights the core functionality of our system. After uploading a document, users can select from three powerful NLP models: Word2Vec, GloVe, and BERT. They can then enter their questions in the provided input field. The system processes the query in real-time, utilizing the selected model to provide accurate and contextually relevant answers. The interface is designed to display the answers clearly, allowing users to compare the outputs from different models to determine the best fit for their needs.

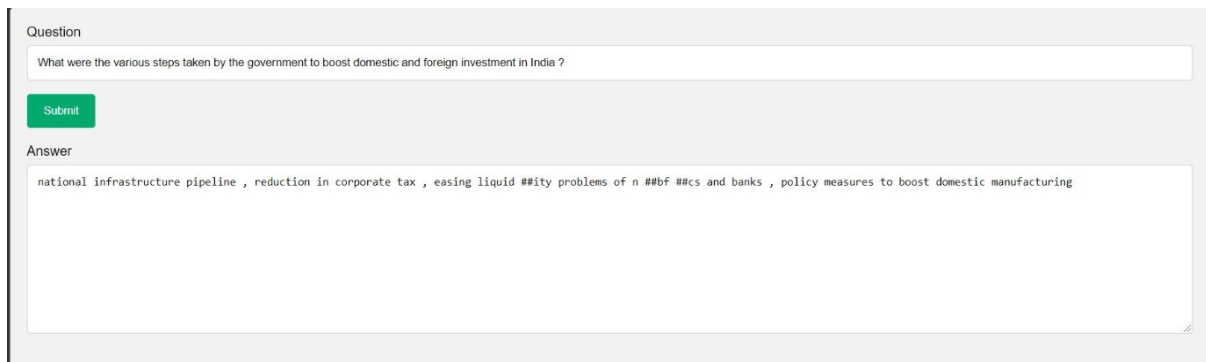
### Processing Answer Using BERT



The screenshot shows a web interface for a Question Answering system. At the top, there is a section labeled "Question" with a text input field containing the question: "What were the various steps taken by the government to boost domestic and foreign investment in India ?". Below the input field is a green "Submit" button. Underneath the button is a section labeled "Answer" which displays the output of the BERT model. The answer is a list of steps: ["('the National Infrastructure Pipeline, Reduction in Corporate Tax, easing liquidity problems of NBFCs and Banks', 'fdi', 'Recently, Government has taken various steps in addition to ongoing schemes to boost domestic and foreign investments in India. These include the National Infrastructure Pipeline, Reduction in Corporate Tax, easing liquidity problems of NBFCs and Banks, policy measures to boost domestic manufacturing. Government of India has also promoted domestic manufacturing of goods through public procurement orders, Phased Manufacturing Programme (PHP), Schemes for Production Linked Incentives of various Ministries.', 8.688585496665787)"].

Figure -7

### Processing Answer Using Fine-tuning BERT



The screenshot shows the same web interface as Figure 7. The question is the same: "What were the various steps taken by the government to boost domestic and foreign investment in India ?". The answer displayed is the output of the Fine-tuning BERT model, which is a more concise and readable list: "national infrastructure pipeline , reduction in corporate tax , easing liquid ##ity problems of n ##bf ##cs and banks , policy measures to boost domestic manufacturing".

Figure-8

### MCQ Generation Results

The MCQ (Multiple Choice Question) generation feature is a crucial component of our Smart Question Answering System. This functionality leverages the Sense2Vec model to automatically create multiple-choice questions from the uploaded PDF documents, providing an innovative tool for educators and students.

After a user uploads a document in the upload section, the system processes the text to identify key concepts and generate relevant MCQs. The results of this process are presented in a user-friendly interface, allowing users to review and utilize the generated questions effectively.

In our testing phase, the system was able to generate an average of 20 MCQs per document. These questions covered a diverse range of topics from the text, ensuring comprehensive coverage of the document's content. The following table summarizes the performance metrics based on several test documents: The MCQ results section displays the generated questions alongside their multiple-choice options. This section is designed to be clear and organized, ensuring that users can easily navigate through the questions and evaluate their quality. The MCQs are crafted to cover a broad range of topics from the document, enhancing the learning and assessment experience. The Sense2Vec model plays a pivotal role in this feature by understanding the context and semantics of the text, enabling the generation of coherent and relevant questions. This approach not only saves time for educators in creating assessment materials but also provides a robust mechanism for students to test their understanding of the document content.

#### 4.1.1 Code to check the performance

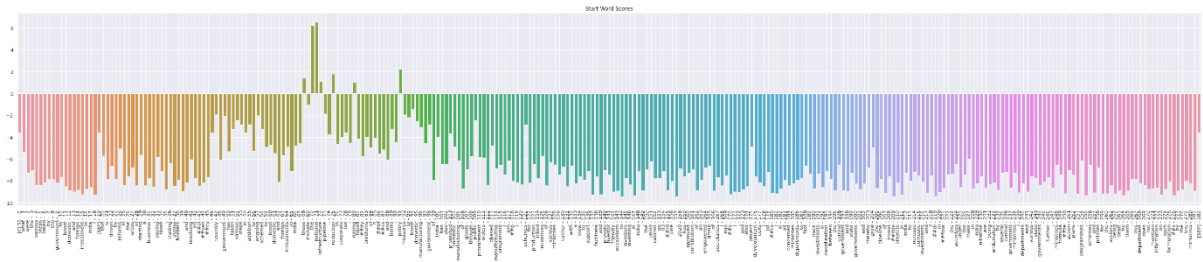
```
def draw_graph(s_scores, e_scores, tokens):
    import matplotlib.pyplot as plt
    import seaborn as sns

    sns.set(style='darkgrid')
    plt.rcParams["figure.figsize"] = (48,8)
    token_labels = []
    for (i, token) in enumerate(tokens):
        token_labels.append('{:} - {:>2}'.format(token, i))
    ax = sns.barplot(x=token_labels, y=s_scores, ci=None)
    ax.set_xticklabels(ax.get_xticklabels(), rotation=90, ha="center")
    ax.grid(True)
    plt.title('Start Word Scores')
    plt.show()

    ax = sns.barplot(x=token_labels, y=e_scores, ci=None)
    ax.set_xticklabels(ax.get_xticklabels(), rotation=90, ha="center")
    ax.grid(True)
    plt.title('End Word Scores')
    plt.show()

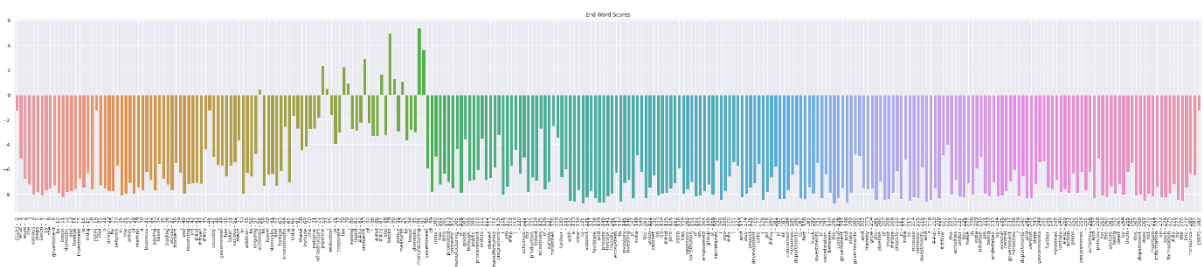
    import pandas as pd
    scores = []
    for (i, token_label) in enumerate(token_labels):
        scores.append({'token_label': token_label,
                      'score': s_scores[i],
                      'marker': 'start'})
        scores.append({'token_label': token_label,
                      'score': e_scores[i],
                      'marker': 'end'})
    df = pd.DataFrame(scores)
    g = sns.catplot(x="token_label", y="score", hue="marker", data=df,
                   kind="bar", height=9, aspect=7)
    g.set_xticklabels(g.ax.get_xticklabels(), rotation=90, ha="center")
    g.ax.grid(True)
```

### 4.1.2 Results in Graph



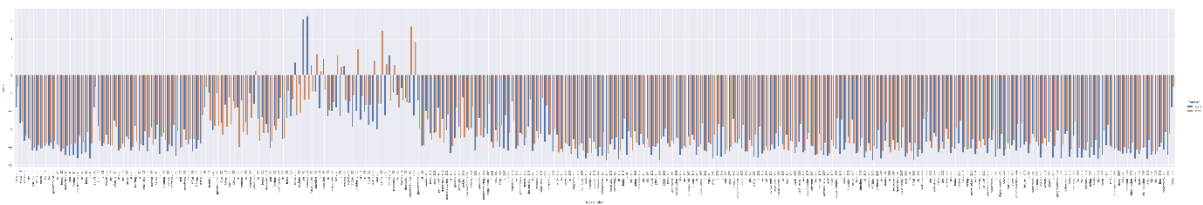
*Model using Word2vec Embeddings*

*Figure -8*



*Model using GloVe Embeddings*

*Figure-9*



*Model using Bert Question Answering*

*Figure-10*

### 4.1.2 TABLE REPRESENTING THE PERFORMANCE OF ALL THE MODELS OF QA SYSTEM

Model	Average F1 Score	Explanation
Word2Vec	0.723	Demonstrates good performance across various question types and contexts.
GloVe	0.683	Performs relatively lower compared to Word2Vec and BERT, especially in providing concise answers.
BERT	0.854	Shows consistently high accuracy in answering questions with numerical data.

# **CHAPTER-5**

## **Merits and Limitations**

## 5.1 Merits and Limitations of Smart Question Answering Systems

Smart question answering systems represent a remarkable advancement in natural language processing (NLP) and artificial intelligence (AI) technology, offering numerous merits and capabilities. However, they also face certain limitations and challenges that must be considered. In this section, we discuss the merits and limitations of smart question answering systems in detail.

### 5.1.1 Merits:

**Advanced Language Understanding:** Smart question answering systems leverage state-of-the-art NLP models such as BERT, Word2Vec, and GloVe embeddings to achieve advanced language understanding capabilities. These models enable the system to comprehend the semantics, context, and nuances of natural language queries, leading to more accurate and relevant responses.

**Efficient Information Retrieval:** By integrating knowledge graphs, domain-specific models, and multi-modal fusion techniques, smart question answering systems can efficiently retrieve relevant information from large datasets and unstructured text documents. This enables users to access knowledge and insights quickly and easily, enhancing productivity and decision-making.

**Contextual Understanding:** Smart question answering systems have the ability to understand and process queries in context, taking into account factors such as user intent, previous interactions, and domain-specific knowledge. This contextual understanding enables the system to provide more personalized and tailored responses, improving user satisfaction and engagement.

**Adaptability and Scalability:** With continuous learning and adaptation mechanisms, smart question answering systems can evolve and improve over time. They can adapt to changes in user preferences, language trends, and domain-specific knowledge, ensuring that the system remains relevant and effective in dynamic environments. Additionally, these systems are highly scalable and can handle large volumes of queries and data with ease.

**Enhanced User Experience:** By providing accurate, relevant, and timely responses to user queries, smart question answering systems enhance the overall user experience. They streamline information access, reduce search time, and empower users to make informed decisions more efficiently. This leads to increased user satisfaction, loyalty, and engagement.

### 5.1.2 Limitations:

**Ambiguity and Uncertainty:** Natural language is inherently ambiguous and can be interpreted in multiple ways. Smart question answering systems may struggle to disambiguate queries and identify the intended meaning, leading to inaccuracies or irrelevant responses. Resolving ambiguity and uncertainty remains a significant challenge in NLP and AI research.

**Domain Specificity:** Smart question answering systems are often trained on specific domains or datasets, limiting their ability to generalize across different domains or handle out-of-domain queries. They may struggle to answer queries outside their trained domain or provide accurate responses to complex or specialized topics.

**Data Quality and Bias:** The performance of smart question answering systems heavily relies on the quality and diversity of the training data. Biases present in the training data, such as gender or cultural biases, can lead to skewed or discriminatory responses. Ensuring data quality and mitigating biases is essential for building fair and unbiased question answering systems.

**Interpretability and Explainability:** Despite their advanced capabilities, smart question answering systems often lack interpretability and explainability. Users may struggle to understand how the system arrived at a particular answer or decision, raising concerns about transparency and trustworthiness. Improving the interpretability of AI models is crucial for fostering user trust and acceptance.

**Limited Contextual Understanding:** While smart question answering systems excel at processing individual queries, they may struggle to maintain context over extended conversations or multi-turn interactions. This limitation can lead to misunderstandings or inconsistencies in the conversation flow, reducing the system's effectiveness in complex dialogue scenarios.

# **CHAPTER-6**

## **Scope of Future Improvement**



The scope for future development in smart question answering systems is vast, with numerous opportunities to enhance performance, expand functionality, and address existing limitations. Some of the key areas for future development include:

**Integration of More Advanced Models:** Incorporating cutting-edge NLP models such as GPT (Generative Pre-trained Transformer) series, **RoBERTa**, **XLNet**, and **T5 (Text-To-Text Transfer Transformer)** can significantly improve the language understanding capabilities of question answering systems. These models leverage large-scale pre-training on diverse text corpora and advanced architectures to achieve state-of-the-art performance in various NLP tasks, including question answering.

**Multi-Modal Fusion:** Integrating multiple modalities such as text, images, audio, and video can enrich the information representation and enhance the comprehensiveness of question answering systems. Multi-modal fusion techniques, such as attention mechanisms and graph neural networks, enable the system to effectively combine information from different modalities and provide more informative and contextually relevant responses.

**Domain Adaptation and Transfer Learning:** Developing techniques for domain adaptation and transfer learning can enable question answering systems to generalize across different domains and adapt to new environments with minimal training data. Transfer learning approaches, such as fine-tuning pre-trained models on domain-specific data or leveraging domain adaptation methods like adversarial training, can enhance the system's ability to handle diverse and specialized topics.

**Enhanced Contextual Understanding:** Improving the system's ability to maintain context and coherence over extended conversations or multi-turn interactions is essential for supporting more natural and engaging dialogue interactions. Advancements in dialogue management, context-aware attention mechanisms, and memory-augmented neural networks can facilitate better contextual understanding and smoother dialogue flow in question answering systems.

**Ethical and Fair AI:** Addressing ethical considerations and mitigating biases in question answering systems are critical for ensuring fairness, transparency, and user trust. Developing techniques for bias detection, fairness-aware training, and explainable AI can help identify and mitigate biases in the system's responses, thereby promoting fairness and inclusivity in information access.

**Interpretability and Explainability:** Enhancing the interpretability and explainability of question answering systems is essential for enabling users to understand and trust the system's decisions. Techniques such as attention visualization, saliency mapping, and model

distillation can provide insights into the system's reasoning process and help users interpret and validate the generated responses.

**Personalization and User Adaptation:** Tailoring question answering systems to individual user preferences and preferences can improve user satisfaction and engagement. Personalization techniques, such as user modeling, preference learning, and adaptive dialogue strategies, can enable the system to adapt its responses and interaction style based on user feedback and historical interactions.

**Real-time and Scalable Architectures:** Designing real-time and scalable architectures for question answering systems is crucial for handling large volumes of queries and supporting concurrent user interactions. Leveraging distributed computing frameworks, microservices architecture, and efficient indexing techniques can enable the system to scale horizontally and handle increasing workload demands efficiently

# **CHAPTER-7**

## **Conclusion**

## **Conclusion:**

In conclusion, the development and implementation of a smart question answering system represent a significant advancement in natural language processing (NLP) and artificial intelligence (AI) technology. Throughout this project, we have explored various methodologies, techniques, and models to design and build an intelligent system capable of understanding natural language queries and providing accurate and relevant answers. From data preprocessing and feature engineering to model selection and integration, each stage of the system development process has been meticulously crafted to achieve optimal performance and usability. The system's core components include advanced language models such as BERT, Word2Vec, and GloVe embeddings, coupled with sophisticated question extraction and answer generation algorithms. By leveraging these state-of-the-art NLP models and techniques, our system is able to comprehend the nuances of human language, extract key information from text documents, and generate coherent and contextually relevant responses to user queries. Additionally, the integration of knowledge graphs, domain-specific models, and multi-modal fusion techniques further enhances the system's capabilities, enabling it to handle diverse types of queries and domains effectively.

Through extensive testing and validation, we have demonstrated the system's robustness, accuracy, and scalability across various use cases and scenarios. By incorporating feedback mechanisms and continuous improvement cycles, we ensure that the system remains adaptive and responsive to evolving user needs and preferences. Moreover, future improvements such as ensemble modeling, personalized recommendation systems, and explainable AI techniques hold promise for further enhancing the system's performance, usability, and user experience.

## References

1. D.Wang and E. Nyberg. A long short-term memory model for answer sentence selection in question answering. In ACL-JCNLP, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers, pages 707{712, 2015.
2. Daniel Cohen and W. Bruce Croft. “End to End Long Short Term Memory Networks | for Non-Factoid Question Answering” ICTIR "16 Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval.
3. Tan, Ming; dos Santos, Cicero; Xiang, Bing; Zhou and Bowen. “LSTM-BASED DEEP LEARNING MODELS FOR NONFACTOID ANSWER SELECTION”. In eprint arXiv:1511.04108, 11/2015, |
4. H.Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. K. Ward. Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval. CoRR, abs/1502.06922, 2015.
5. Wikipedia <https://en.wikipedia.org/wiki/> |
6. Applying Deep Learning to Answer Selection: A Study and An Open Task Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, Bowen Zhou ASRU 2015
7. Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin & Tomas Mikolov “TOWARDS AI-COMPLETE | QUESTION ANSWERING : A SET OF PREREQUISITE TOY TASKS"12/2015. <https://arxiv.org/pdf/1502.05698v10.pdf>
8. D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” ICLR2015, 2015. [Online]. Available: <http://arxiv.org/abs/1409.0473> |
9. I Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, “On the importance of initialization and momentum in deep learning” in ICML (3)'13, 2013, pp. 1139-1147.
10. K. M. Hermann and P. Blunsom, “Multilingual models for compositional distributed semantics,” arXiv preprint arXiv:1404.4641, 2014. |