# Sentiment Analysis and Question Answering using BERT

## 1. Introduction

Natural Language Processing (NLP) has significantly advanced in recent years, owing much of its progress to the development of transformer-based models like BERT (Bidirectional Encoder Representations from Transformers). BERT has set new benchmarks in various NLP tasks, such as sentiment analysis, question answering, and text classification. In this project, we leverage BERT's pre-trained models to perform two key NLP tasks: **Sentiment Analysis** and **Question Answering**. By using these models, we aim to analyze the sentiment of a given text and provide accurate answers to questions based on a provided context.

This report outlines the approach used to solve these problems, the software and hardware requirements, and provides example code to demonstrate the implementation.

## 2. Problem Statement

**Sentiment Analysis**: The task is to determine the sentiment of a given text, classifying it into one of the following categories:

- Very Negative
- Negative
- Neutral
- Positive
- Very Positive

The goal is to use BERT's multilingual pre-trained model for sentiment analysis to classify the sentiment of text with accuracy.

**Question Answering**: Given a context (a passage of text), the objective is to answer specific questions that are asked about the content of the context. For example, given a paragraph about BERT, a question like "What is BERT?" should return a relevant, accurate answer based on the context.

These tasks are critical for various applications in NLP, including customer feedback analysis, chatbot development, and information retrieval systems.

## 3. Software and Hardware Requirements

**Software Requirements**:

- **Python**: The primary programming language used for implementing this solution.
- **PyTorch**: A deep learning framework used to load and run BERT models.
- **Transformers Library**: Developed by Hugging Face, this library provides easy access to pre-trained models such as BERT and other transformer-based models for various NLP tasks.
- **Torch**: For handling tensor operations necessary for NLP tasks and deep learning models.
- **pip** (or **conda**): A package manager used to install the required libraries.

**Required Libraries**:

- torch (for tensor operations)
- transformers (for accessing pre-trained models like BERT)

To install these, you can run:

```
pip install torch transformers
```

**Hardware Requirements**:

- **CPU**: Modern multi-core processors capable of handling PyTorch computations.
- **GPU** (Optional but recommended): A CUDA-enabled NVIDIA GPU (e.g., RTX 3000 series or higher) for faster model inference and training. The BERT models are large and require significant computational power when processing large text datasets.
- **RAM**: At least 8 GB of system memory is recommended, but more (16 GB or more) will improve performance when handling larger datasets.

## 4. Methodology

This solution uses two pre-trained BERT models:

1. **Sentiment Analysis Model**:
   - Model: nlptown/bert-base-multilingual-uncased-sentiment (multilingual BERT model fine-tuned for sentiment analysis).
   - The model is used to classify input text into one of the five sentiment categories: Very Negative, Negative, Neutral, Positive, Very Positive.
2. **Question Answering Model**:
   - Model: bert-large-uncased-whole-word-masking-finetuned-squad (fine-tuned BERT model on the SQuAD dataset for question answering).
   - The model is used to extract an answer from a given context based on a specific question.

## 5. Implementation Details

The core implementation is divided into two parts: **Sentiment Analysis** and **Question Answering**. Below are the details of each:

**Sentiment Analysis**:

- **Input**: A string of text (e.g., "I love programming in Python!").
- **Process**: The input text is tokenized using BERT's tokenizer, and the tokenized input is passed to the sentiment analysis model to get logits.
- **Output**: The sentiment is classified based on the logits (probabilities), which are mapped to one of the sentiment labels (Very Negative, Negative, Neutral, Positive, Very Positive).

**Question Answering**:

- **Input**: A question and a context (e.g., "What is BERT?" and a paragraph about BERT).
- **Process**: The question and context are tokenized, and the model predicts the start and end positions of the answer within the context.
- **Output**: The answer is extracted and converted back to a human-readable string.

## 6. Results

The model successfully classifies text sentiment and answers questions based on the provided context. For instance:

- **Sentiment Analysis Example**:
    - Input Text: “I love programming in Python!”
    - Output: “Very Positive”
- **Question Answering Example**:
    - Input Question: “What is BERT?”
    - Input Context: A description of BERT.
    - Output: "BERT is a transformer-based model designed for various NLP tasks."