# **Running GenAI on Intel AI Laptops and Simple LLM Inference on CPU and fine-tuning of LLM Models using Intel® OpenVINO™**

# **PROJECT REPORT**

# **Submitted by**

# **SUSHMA M-1SB21CS111**

# **MOUNICCA R-1SB21CS067**

# **CHANDANA A-1SB21CS020**

# **MAMATHA J-1SB21EE017**

# **KEERTHANA R-1SB21ME008**

# 

**Sri Sairam College of Engineering**

**Sai Leo Nagar, Guddanahalli Post, Anekal, Bengaluru-562106**

**Visvesvaraya Technological University (VTU), Belagavi**

# **INTRODUCTION**

# Dive into the world of Generative Artificial Intelligence (GenAI) with our beginner-friendly hands-on exercises. Learn the fundamentals, perform basic Large Language Model (LLM) inference on a CPU, and fine-tune an LLM model to craft your custom Chatbot. Discover the potential of GenAI and its transformative applications.

**OBJECTIVE**

The objective of this problem statement is to provide beginners with a comprehensive introduction to Generative Artificial Intelligence (GenAI) through practical, hands-on exercises. Participants will gain foundational knowledge of GenAI concepts and applications. They will learn how to perform basic Large Language Model (LLM) inference using a CPU, enabling them to understand the mechanics of LLMs. Furthermore, participants will delve into the process of fine-tuning an LLM model, equipping them with the skills to customize and create their own Chatbot. By the end of this experience, participants will not only understand the theoretical underpinnings of GenAI but also acquire practical skills in LLM inference and fine-tuning, empowering them to develop innovative AI-driven solutions. This journey aims to demystify GenAI, making it accessible and engaging for newcomers to the field.

# **PROBLEM STATEMENT**

# This problem statement is designed to introduce beginners to the exciting field of Generative Artificial Intelligence (GenAI) through a series of hands-on exercises. Participants will learn the basics of GenAI, perform simple Large Language Model (LLM) inference on a CPU, and explore the process of fine-tuning an LLM model to create a custom Chatbot.

**SOLUTION**

# In this problem statement, participants will create an educational chatbot designed to address basic queries from students. This chatbot leverages Generative Artificial Intelligence (GenAI) and Large Language Models (LLMs) to provide quick and accurate responses to common student inquiries. Throughout the hands-on exercises, participants will learn to perform simple LLM inference on a CPU and explore the intricacies of fine-tuning an LLM model. The resulting chatbot will serve as a personalized educational assistant, capable of answering questions about coursework, providing explanations on various topics, and assisting with homework queries. By the end of the project, participants will have a solid understanding of GenAI principles and practical experience in developing a functional AI-driven chatbot tailored to meet students' educational needs. This experience will not only enhance their technical skills but also demonstrate the practical applications of AI in the field of education.

# **MAJOR ADVANTAGES**

1. Foundational Understanding of GenAI

2. Hands-on Experience with LLMs

3. Skill Development in Model Fine-Tuning

4. Enhanced Problem-Solving and Critical Thinking

5. Career Advancement Opportunities

**MAJOR CHALLENGES**

1. Understanding the Basics of GenAI

2. Resource Limitations for LLM Inference

3. Data Collection and Preparation for Fine-Tuning

4. Technical Complexity of Fine-Tuning

5. Creating a Custom Chatbot with Desired Behaviour

# **PROGRAMMING CODE**

**model.py**

# Load model directly

from transformers import AutoTokenizer, AutoModelForCausalLM

import torch

# Hugging Face model and tokenizer initialization

model\_name = "bigscience/bloom"

tokenizer = AutoTokenizer.from\_pretrained("bigscience/bloom")

model = AutoModelForCausalLM.from\_pretrained("bigscience/bloom")

# Example input

text = "Sample input text"

inputs = tokenizer(text, return\_tensors="pt")

# Perform inference to get model outputs

outputs = model(\*\*inputs)

# Export the model to ONNX format with opset version 14

onnx\_model\_path = "model.onnx"

torch.onnx.export(

    model,

    (inputs['input\_ids'],),

    onnx\_model\_path,

    input\_names=["input\_ids"],

    output\_names=["output"],

    dynamic\_axes={"input\_ids": {0: "batch\_size"}, "output": {0: "batch\_size"}},

    opset\_version=14

)

**app.py**

from flask import Flask, request, jsonify

from openvino.runtime import Core

from transformers import AutoTokenizer

import numpy as np

import logging

app = Flask(\_\_name\_\_)

# Set up logging

logging.basicConfig(level=logging.DEBUG)

# Initialize OpenVINO runtime and load the model

core = Core()

model\_path = "./model\_ir/model.xml"  # Ensure this path is correct

compiled\_model = core.compile\_model(model\_path, device\_name="CPU")

# Initialize the tokenizer for BLOOM

tokenizer = AutoTokenizer.from\_pretrained("bigscience/bloom")

@app.route('/')

def index():

    return "Welcome to the BLOOM model API. Use the /predict endpoint to get predictions."

@app.route('/predict', methods=['POST'])

def predict():

    try:

        # Get the input text from the request

        data = request.json

        text = data['text']

        logging.debug(f"Received text for prediction: {text}")

        # Tokenize the input text

        inputs = tokenizer(text, return\_tensors="np")

        input\_ids = inputs["input\_ids"]

        attention\_mask = inputs["attention\_mask"]

        logging.debug(f"Tokenized input: {input\_ids}, {attention\_mask}")

        # Convert input tensors to the required format for OpenVINO

        input\_ids = np.array(input\_ids, dtype=np.int32)

        attention\_mask = np.array(attention\_mask, dtype=np.int32)

        # Run inference

        results = compiled\_model({compiled\_model.inputs[0].any\_name: input\_ids, compiled\_model.inputs[1].any\_name: attention\_mask})

        logging.debug(f"Inference results: {results}")

        # Extract the output

        output\_data = results[compiled\_model.outputs[0].any\_name]

        logging.debug(f"Output data: {output\_data}")

        # Return the results as JSON

        return jsonify(output\_data.tolist())

    except Exception as e:

        # Log the error

        logging.error("Error during prediction: %s", e, exc\_info=True)

        return jsonify({"error": str(e)}), 500

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(host='0.0.0.0', port=5000)

**OUTPUT**

(openvino\_env) C:\Program Files (x86)\Intel\openvino\_2024.1.0\openvino\_env>cd C:\Users\User\OneDrive\Desktop\INTEL\_OPENVINO

(openvino\_env) C:\Users\User\OneDrive\Desktop\A INT>python app.py

DEBUG:urllib3.connectionpool:Starting new HTTPS connection (1): huggingface.co:443

DEBUG:urllib3.connectionpool:https://huggingface.co:443 "HEAD /bigscience/bloom/resolve/main/tokenizer\_config.json HTTP/11" 200 0

\* Serving Flask app 'app'

\* Debug mode: off

.json HTTP/11" 200 0

\* Serving Flask app 'app'

\* Debug mode: off

INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.

\* Running on all addresses (0.0.0.0)

\* Running on <http://127.0.0.1:5000>

\* Running on <http://192.168.46.4:5000>

INFO:werkzeug:Press CTRL+C to quit

# **CONCLUSION**

Generative Artificial Intelligence (GenAI) opens up a world of possibilities for creating intelligent systems capable of generating human-like text. This problem statement serves as an introduction to this exciting field, offering beginners hands-on experience with Large Language Models (LLMs). Through this series of exercises, participants will build a solid foundation in understanding the core concepts and applications of GenAI. They will gain practical skills by performing simple LLM inference on a CPU, allowing them to see firsthand how these powerful models generate text based on given inputs.

Moreover, participants will delve into the intricacies of fine-tuning LLMs, a crucial process for tailoring models to specific applications such as custom chatbots. This experience will highlight the flexibility and adaptability of LLMs, demonstrating their potential in various domains. By the end of these exercises, participants will not only have enhanced their technical proficiency in working with GenAI but also gained insight into the broader implications and possibilities of this technology. This foundational knowledge and hands-on experience will empower participants to explore and innovate within the field of GenAI, paving the way for future advancements and applications in artificial intelligence.