Intel Unnati Industrial Training Program 2024-2025

Problem Statement

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

Unique Idea Brief (Solution)

- Created an advanced chatbot for Gitam University to instantly address user queries on university topics.
- Powered by Meta Llama 3-8b Instruct, fine-tuned with Intel AI Tools for top performance.
- Integrated with the Gradio Interface for a user-friendly experience.
- Provides detailed information on programs, admissions, facilities, fees, and scholarships.
- Allows users to customize response settings to explore different response styles.

Features Offered:-

1. Configuration File Handling
2. Hugging Face Authentication
3. Model Selection Interface
4. OpenVINO Runtime Initialization
5. Gradio Interface

Process flow

1. **Setup:**
   - Disable an environment variable.
   - Install/update Python packages.
   - Uninstall conflicting packages and install updated versions.
2. **Configuration:**
   - Fetch `llm_config.py` from a local path or GitHub if missing.
3. Hugging Face Authorization:Ensure login for model access.
4. **Model Selection:**
   - Use `ipywidgets` dropdowns for model language and selection.
5. **Model Precision:**
   - Checkboxes for selecting model precision (FP16, INT8, INT4) and optional AWQ.

6.Model Conversion:

-Convert models with optimum-cli export.

-Configure INT4 compression.

7.Model Size:

-Display model sizes and compression rates.

8.OpenVINO Initialization:

-Initialize OpenVINO and set up devices.

9.Model Inference:

-Load the model and run a test inference ("10 + 20 =").

10.Gradio Interface:-Setup Gradio for user interaction with predefined examples related to GITAM University.

Technologies Used

The technologies used in code leverages a combination of state-of-the-art machine learning libraries, optimization tools, and interactive interfaces to facilitate the deployment, optimization, and interaction with large language models.

1.Python Libraries and Tools:

- **os**: For environment variables and file operations.
- **Pathlib**: For file and directory path manipulations.
- **requests**: For making HTTP requests.
- **shutil**: For file operations such as copying files.
- **Hugging Face Transformers**: For handling transformer models.
- **Optimum-Intel**: Hugging Face's extension for optimized inference and training on Intel hardware.
- **OpenVINO**: Intel's toolkit for optimized deep learning model inference.

- **gradio**: For creating interactive user interfaces.
- **IPython Widgets (ipywidgets)**: For creating interactive widgets in Jupyter
- **datasets**: For handling and processing datasets.
- **accelerate**: For faster training of transformer models.
- **torch (PyTorch)**: A deep learning framework used for training and deploying machine learning models.
- **onnx**: Open Neural Network Exchange, a format for deep learning models.
- **einops**: For tensor operations.
- **transformers_stream_generator**: A library for streaming generation with transformers.
- **tiktoken**: For tokenization tasks in transformers.
- **bitsandbytes**: For 8-bit optimizations in transformer models.

2. External Dependencies and Repositories:

- **Hugging Face GitHub Repositories**: Cloning and installing from GitHub repositories for Hugging Face's optimum-intel and Intel's nncf.
- **pip**: Python package installer used for managing libraries and dependencies.

3.Model Compression and Optimization:

- **INT4, INT8, FP16 Models**: Different precision formats for model weights, aiming at optimizing model performance and size.
- **AWQ (Asymmetric Weight Quantization)**: For further compressing model weights.
- **optimum-cli**: Command-line interface for exporting and converting models to OpenVINO format.

4.Interactive Widgets:

- **ipywidgets**: For creating dropdowns, checkboxes, and displaying information in an interactive manner within Jupyter Notebooks.

- **Model and Tokenizer Handling**:

❑ **AutoConfig**: For loading model configurations.

❑ **Auto Tokenizer**: For handling tokenization tasks.

❑ **OVModelForCausalLM**: From the optimum.intel.openvino library, for loading and running models optimized for OpenVINO.

5.Gradio UI:

❑ **gradio**: For creating web-based user interfaces to interact with the models.

Team Members and Contributions:

1. Ramireddygari Yaswanth Reddy(yaswanthramireddy18@gmail.com)

- Managed Installation and Environment Setup.
- Designed Interface and Interaction.
- Deployed the Model.

2. Poluru Sai Chaitanya(spolur@gitam.in)

- Researched GITAM University information from College Website and Brochure.
- Compiled the End Report.
- Studied about Hugging Face and Transformers.

3. Lalam Jithendhra(jlalam@gitam.in)

- Prepared and Converted the Model.
- Ensured Code Quality Assurance.
- Collected industrial insights and critical data from Industrial Sessions and Q-A Sessions.
- Created and managed the GitHub repository.

Special Thanks to our Mentors:-

Dr. Arvind Kumar

Assistant Professor

Department of EECE

Gitam University,Bengaluru

Abhishek Nandy

Intel Unnati Labs External Mentor

Kolkata

Conclusion

The provided code showcases a robust framework for optimizing and deploying large language models using Intel's OpenVINO and related tools. It leverages various precision formats and quantization techniques to enhance inference efficiency while maintaining accuracy. The integration of Hugging Face's libraries and Gradio enables seamless model interaction, while ipywidgets facilitate a user-friendly configuration experience. Overall, this code is a valuable resource for efficient AI model deployment on Intel hardware, offering both performance and usability.