Intel® Unnati

Intel® Unnati Industrial Training Program

intel.

Openvino Chatbot

This presentation introduces beginners to Generative AI with hands-on exercises. Learn GenAI basics, perform LLM inference, and fine-tune models for custom Chatbots.

https://www.linkedin.com/in/siddhant-saini/

https://github.com/Siddhant-Saini

Problem Statement

PS-16 : Running GenAI on Intel AI Laptops and Simple LLM Inference on CPU and fine-tuning of LLM Models using Intel® OpenVINO™

https://www.linkedin.com/in/siddhant-saini/

Unique Idea Brief (Solution)

1. Optimized Model Handling Across Different Precision Formats: My chatbot leverages the OpenVINO toolkit to handle models in various precision formats (FP16, INT8, INT4). This allows for optimized performance across different hardware configurations, catering to environments with varying computational power. This capability ensures your chatbot can be deployed efficiently in diverse scenarios, from high-power server deployments to more constrained edge devices.

```
# OpenVINO Settings
MODEL_PRECISION="INT4"          # "FP16", "INT8", "INT4"
INFERENCE_DEVICE="CPU"          # "CPU", "GPU", "GPU.0" (iGPU)
```

FP16

INT8

INT4

1. FP16 (16-bit Floating Point)

- Use Case: Ideal for deployment on GPUs or CPUs with FP16 support. It is suitable if you need a balance between performance and accuracy without significant degradation.
- Advantages: Faster computations than FP32 with a minimal loss in accuracy. It can significantly reduce memory usage, which is beneficial for running larger models.
- Consider If: Your deployment environment supports FP16 operations and you require near-original model performance.

2. INT8 (8-bit Integer)

- Use Case: Best for CPUs and edge devices where memory and processing power are limited. It's widely supported by many hardware accelerators designed for efficient AI inference.
- Advantages: Provides substantial performance improvements and reductions in memory usage compared to FP16 and FP32. Especially effective for reducing latency in real-time applications.
- Consider If: You need high throughput and latency is critical, and you can handle some loss in accuracy through proper calibration and testing.

3. INT4 (4-bit Integer)

- Use Case: Suitable for scenarios where model size and inference speed are extremely critical, even at the cost of significant accuracy loss.
- Advantages: Maximizes the reduction in model size and computational demand, potentially enabling the deployment on very constrained environments.
- Consider If: Your application can tolerate a higher accuracy trade-off and the deployment environment supports INT4 operations. This might require advanced techniques to retain acceptable performance.

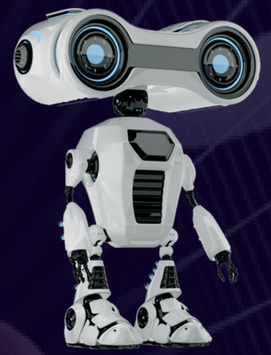
Unique Idea Brief (Solution)

2. Integration with Hugging Face and OpenVINO:
The chatbot integrates models from Hugging Face, optimized through OpenVINO's toolkit, which is not commonly seen in standard chatbot implementations. This integration allows the use of cutting-edge NLP models with enhanced inference speed and reduced resource consumption, positioning your chatbot to handle complex queries with greater responsiveness.

Features Offered

1. Model Authentication and Downloading:
Scripts for authentication using Hugging Face and downloading different language models based on their vendors (e.g., Intel, meta-llama). Conversion of these models into different OpenVINO-compatible precision formats for optimized inference.

2. Model Conversion and Optimization:
Conversion of PyTorch models to OpenVINO format (FP16, INT8, INT4) using Optimum and NNCF (Neural Network Compression Framework) for different levels of precision and efficiency in model inference. Application of quantization techniques to compress models further and optimize performance.



3. Document Extraction and Vectorstore Generation:
Automated parsing of HTML documents to extract main content and convert these into document objects. Indexing of the extracted documents using vector embeddings for efficient retrieval based on semantic similarity (using langchain and Chroma).

4. Server and Client Setup for Interaction:
Configuration of an API server using FastAPI that can handle user queries, process them through a retrieval and generation pipeline, and return answers efficiently. Client-side script using Streamlit for a user-friendly interface to interact with the QA server, displaying conversations and handling user inputs.

Process Flow

A) Document Preparation/Extractor:

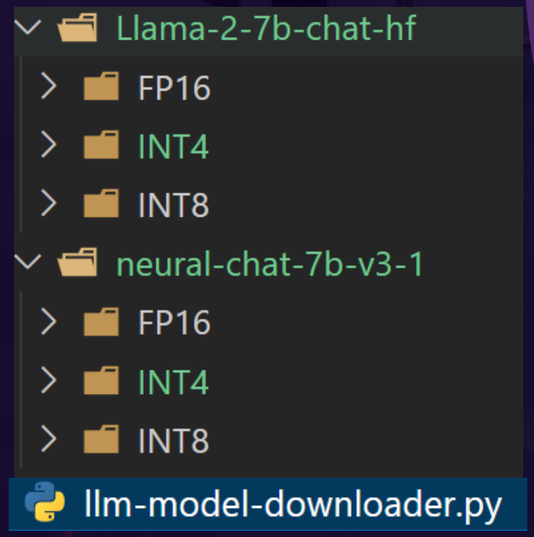
1. **Environment Setup:** Load necessary configurations from environment variables and prepare the embedding model.
2. **Document Extraction:** Use glob to find HTML files, then extract text content from the <main> section using BeautifulSoup.
3. **Data Handling:** Check if data already exists in a pickle file; if not, proceed with fresh data extraction and processing.
4. **Text Processing:** Split documents into manageable chunks using SpacyTextSplitter.
5. **Embedding Generation:** Convert text chunks into embeddings with the HuggingFaceEmbeddings model.
6. **Vector Store Creation:** Store the embeddings in a Chroma vector database, facilitating efficient data retrieval.

openvino-doc-specific-extractor.py

B) Model Download:

neural-chat-7b-v3-1

Llama-2-7b-chat-hf

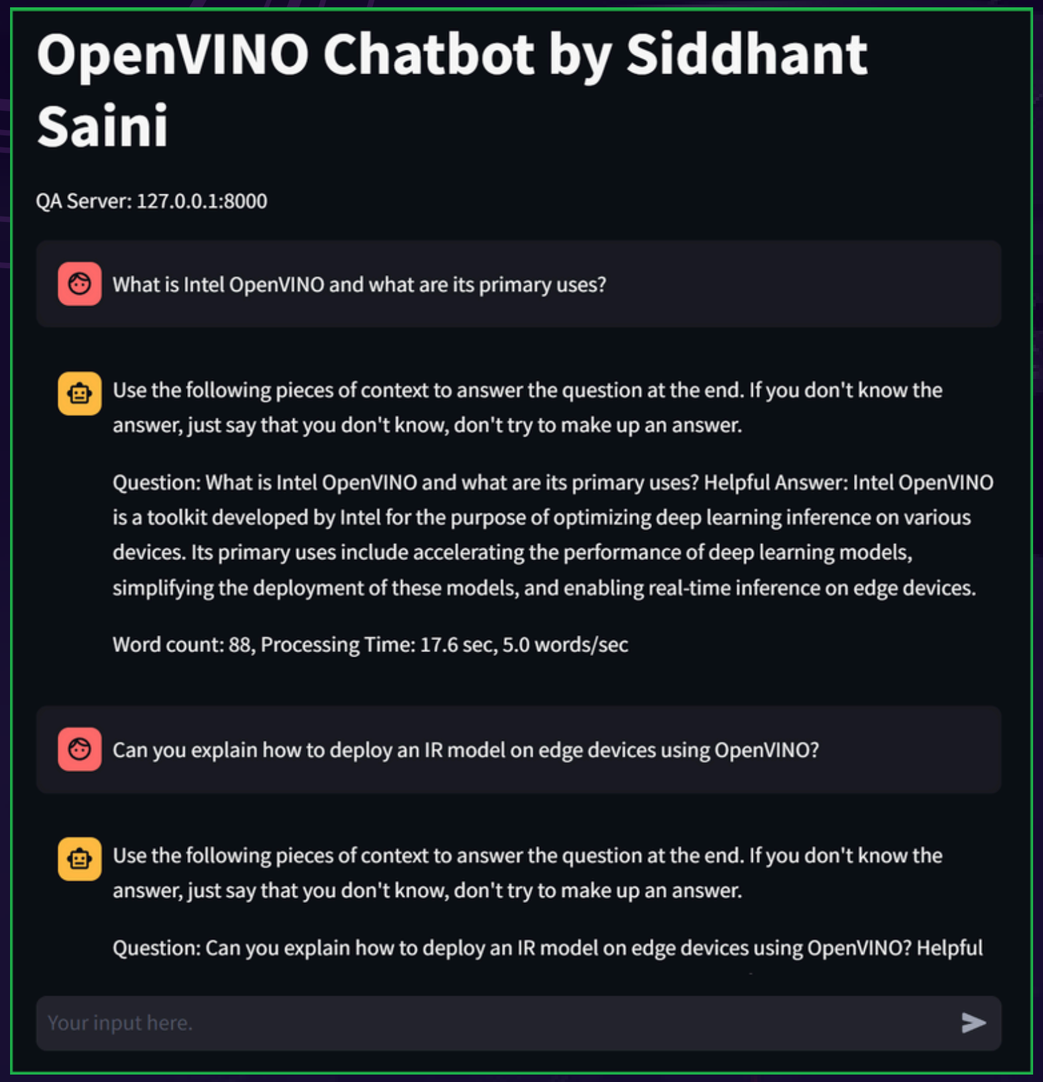


C) Client-Server Integration:

Our OpenVINO chatbot leverages a robust client-server architecture designed to enhance user interaction and streamline backend processing. The client side, developed with Streamlit, offers an intuitive and responsive user interface that allows real-time input and displays of chat responses. This setup ensures that users can interact seamlessly with the AI, making the experience as engaging and efficient as possible.

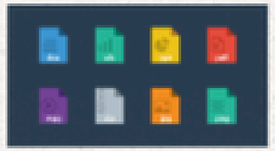
On the server side, FastAPI handles API requests and orchestrates the data processing through OpenVINO's optimized models. This server setup not only ensures secure handling of user data through protocols like HTTPS but also enhances the system's ability to scale and manage varying loads efficiently. The integration between Streamlit and FastAPI allows for rapid data exchanges, maximizing the chatbot's performance and reliability for all user interactions.

D) Deployment/Results :



In the screenshot of the OpenVINO chatbot interface, the processing time for answering a question about Intel OpenVINO's primary uses is recorded as 17.6 seconds, with a word count of 88 words. This results in a processing speed of approximately 5.0 words per second. This metric indicates the efficiency and performance of the chatbot in handling queries, showcasing its capability to process and deliver answers in real-time on a user-friendly platform.

OpenVino-Chatbot



Source Files
(zip archives)

ARCHITECTURE

Extracts text from HTML files using BeautifulSoup

Beautiful Soup

spaCy

.vectorstore_300_0

Text Extraction
(Pickle file created for future processing)

Convert to Chunks
(Size: 300/500/1000)

Chroma Vectorstore
(Chunk Size & Overlap)

Embeddings Similarity Search
(Server-side)

Prompt Template + RetrievalQA

Vectorstore-Chunksize-Overlap

- .vectorstore_300_0
- .vectorstore_500_0
- .vectorstore_500_100
- .vectorstore_1000_100

intel. LLaMA 2

Intel-neural-chat-7b
Meta-llama2-7b

LangChain + Chroma

Response from Model

User Query/ Questions
(Client-Side)

Streamlit Interface

https://github.com/Siddhant-Saini

https://linkedin.com/Siddhant-Saini

Architecture Diagram

Technologies Used:

- Hugging Face Hub
- Optimum by Hugging Face
- OpenVINO Toolkit
- NNCF (Neural Network Compression Framework)
- Streamlit
- FastAPI
- Spacy
- Langchain
- BeautifulSoup
- Requests
- TQDM
- python-dotenv
- Pickle
- Uvicorn
- Chroma (from Langchain)
- Transformers
- AutoModel (from Transformers)

Team members and contribution:

Name : Siddhant Saini

Email-id: siddhant.saini@learner.manipal.edu

Phone: +91-9559900558

(Individual Project)

https://www.linkedin.com/in/siddhant-saini/

https://github.com/Siddhant-Saini

Conclusion

The OpenVINO chatbot project represents my sophisticated integration of modern technologies to create a powerful and efficient chatbot capable of real-time user interactions. Leveraging the OpenVINO Toolkit for optimized model inference and employing a suite of tools from Hugging Face for advanced natural language processing capabilities, my project successfully demonstrates how cutting-edge AI can be seamlessly combined with high-performance backend frameworks like FastAPI and user-friendly front-end interfaces like Streamlit.

My project not only achieves enhanced performance through model optimization techniques like FP16, INT8, and INT4 quantizations but also ensures scalability and responsiveness suitable for enterprise-level applications. By incorporating technologies such as Spacy for text processing and Langchain for efficient document handling and vector storage, it sets a robust foundation for handling complex queries and delivering accurate responses.

Overall, this project stands out as a testament to the potential of AI in revolutionizing user interactions in various domains, providing a template for future advancements in chatbot technology and AI-driven applications. It showcases a balanced approach to performance, user experience, and security, making it a valuable asset for any organization looking to enhance their customer interaction frameworks with AI.

Thank You!

Get in touch

https::/Personal-Portfolio/

https://www.linkedin.com/in/siddhant-saini/

https://github.com/Siddhant-Saini

+91-9559900558

siddhant.saini@learner.manipal.edu