Demystifying Generative AI: Develop and Optimize Your Own Talking Chatbot

Ke Ding – Principal AI Engineer

Qun Gao – Machine Learning Engineer

Notices and Disclaimers

For notices, disclaimers, and details about performance claims, visit www.intel.com/PerformanceIndex or scan the QR code:

Ke Ding

Principal AI Engineer, Intel

Ke Ding is Principal AI Engineer and Engineering Director at Artificial Intelligence and Analytics Division within Intel Software and Advanced Technology group (SATG), responsible for applied machine learning E2E workload development, framework optimization and new technology exploration for Intel platforms. Most recently Ke is contributing into Intel's GenAI and LLM initiatives including developing finetuning and inference workflows, LLM-as-a-Service and foundation model training.

Qun Gao

Machine Learning Engineer, Intel

Qun Gao is a Machine Learning Engineer at Intel, working on model compression and acceleration across multiple deep learning frameworks, such as Intel® Neural Compressor and Intel® Extension for Transformers. He is passionate about providing customers the best solutions to reduce their model size and increase the speed of model inference for deployment on either CPUs or GPUs. He received his Ph.D in Electrical and Computer Engineering from the University of Florida. The scope and impact of his work is represented by the wide variety of journals in which his work has been published (Science, IEEE Trans. VLSI Syst., etc). He also holds 12 issued US patents.

Agenda

- Goals and Objectives
- Intel GenAI Software Stack Overview
- Lab Sessions
  - Inference and Retrieval-augmented generation (RAG)
  - Customize the model with Parameter-Efficient Fine-Tuning (PEFT)
  - Performance improvement with quantization
  - Endpoint serving
- Summary & CTA

Goals and Objectives

- After this class, you will be able to:
  - Successfully build a chatbot using Neural Chat within Intel® Extension for Transformers
  - Run LLM workflows for finetuning, quantization and inference deployment
  - Onboard Intel® Developer Cloud to start your own GenAI journey

Intel AI Software Portfolio

https://developer.intel.com/ai

Engineer Data → Create Models → Optimize & Deploy

MODIN · SciPy · pandas · NumPy · Apache Spark · Numba

dmlc XGBoost · PyTorch · ONNX RUNTIME · OpenVINO · scikit learn · TensorFlow · DirectML · SigOpt AutoML · intel Neural Compressor

Write Once Deploy Anywhere

Data Analytics at Scale†

Machine & Deep Learning Frameworks, Optimization and Deployment Tools†

oneAPI

Intel® oneAPI Deep Neural Networks Library

Intel® oneAPI Collective Communications Library

Intel® oneAPI Math Kernel Library

Intel® oneAPI Data Analytics Library

Open, cross-architecture programming model for CPUs, GPUs, and other accelerators

CLOUD & ENTERPRISE

CLIENT & WORKSTATION

EDGE

intel AI ANALYTICS TOOLKIT

Accelerate End-to-End Data Science and AI

Intel® Developer Cloud and Intel® Developer Catalog

Try the latest Intel tools and hardware, and access optimized AI Models

cnvrg.io

Full stack ML operating system

Hugging Face

Intel optimizations and fine-tuning recipes, optimized inference models, and model serving

Note: components at each layer of the stack are optimized for targeted components at other layers based on expected AI usage models, and not every component is utilized by the solutions in the rightmost column

† This list includes popular open source frameworks that are optimized for Intel hardware

Intel GenAI – From Hugging Face Ecosystem View

Chat Bot | Summarization | Completion | Codegen | Stable Diffusion

BLOOM (a BigScience initiative; 176B params; 59 languages; Open-access) | GPT-j, MPT, Falcon, Llama-2 | StarCoder | Diffusers

Train, Fine Tuning

- PEFT (LoRA etc.)
- Accelerate
- Optimum (intel, habana)
- Diffusers

Optimization

- Optimum (quant, etc.)

Inference Deployment

- Accelerate
- Diffusers
- Text generation inference
- Optimum (intel, habana)

Foundation

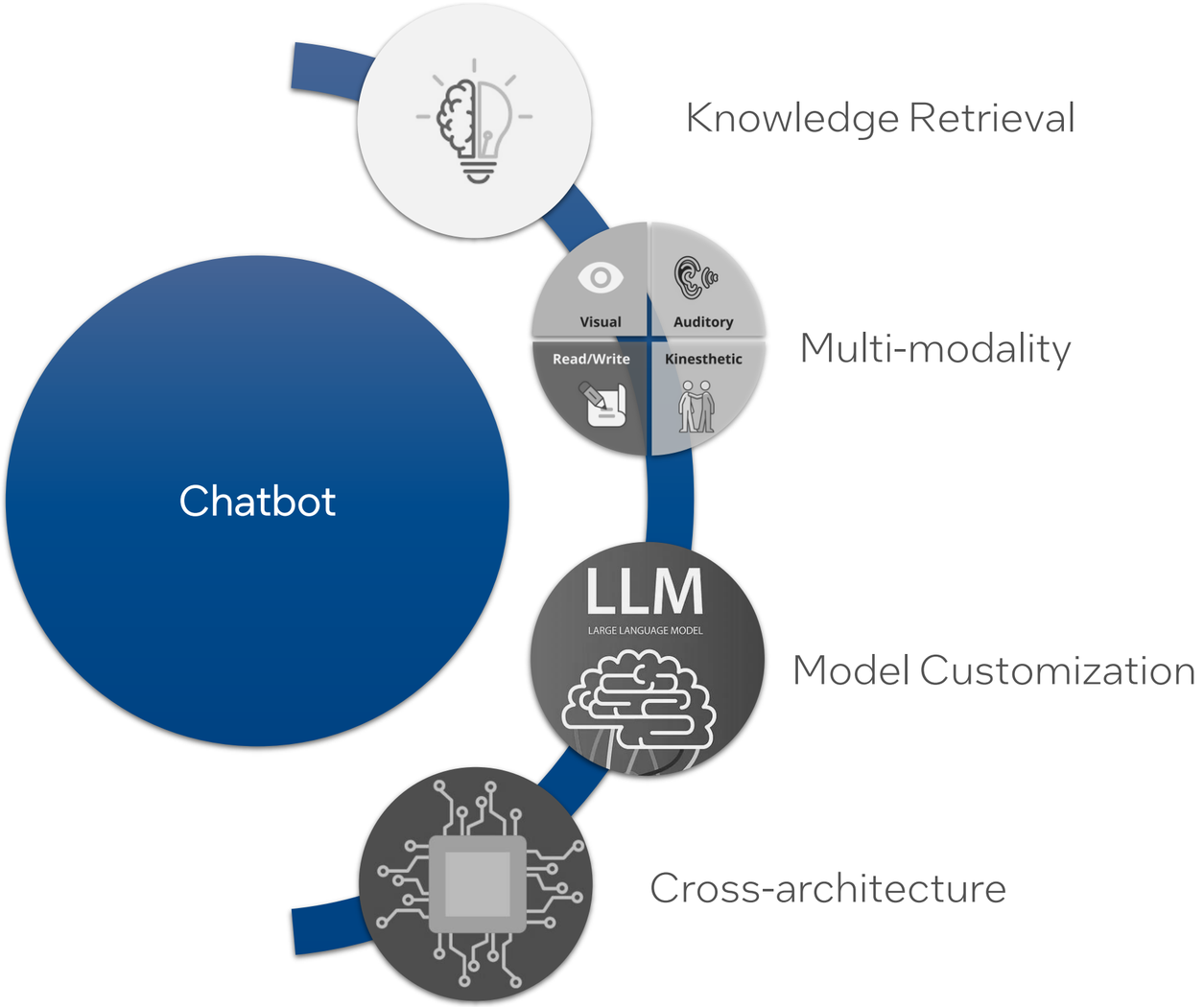
Intel® Extension for Transformers, Intel® Extension for PyTorch*, Intel® Extension for DeepSpeed*, fastRAG

Workflows: Customizable Reference Design

Productive

Customizable

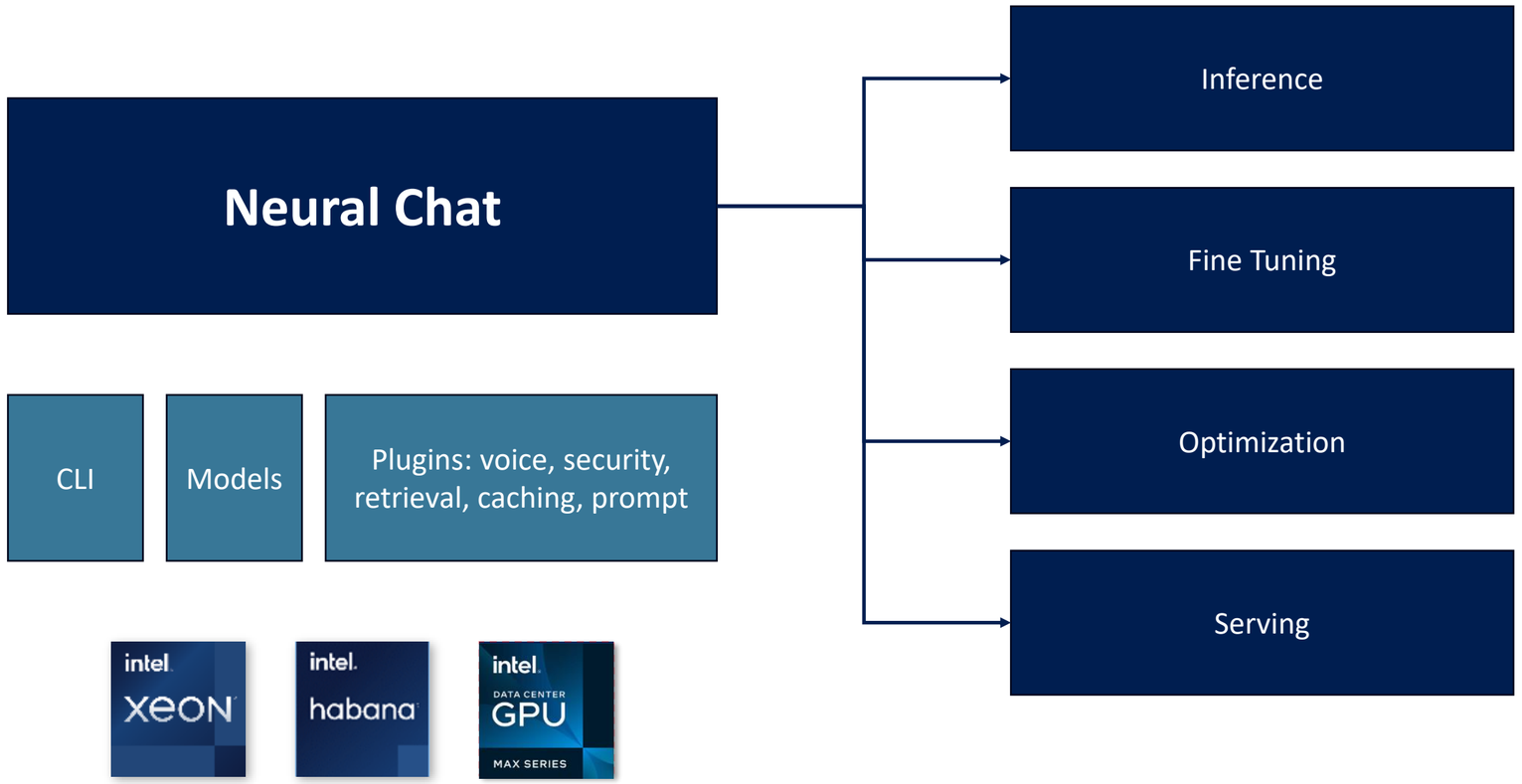
Portable



LLM Workflows Powered by Neural Chat

Single unified workflow with the same APIs for Xeon and Habana HPU

https://github.com/intel/intel-extension-for-transformers/tree/main/intel_extension_for_transformers/neural_chat



Lab Session Network Setup

Notebook URLs:

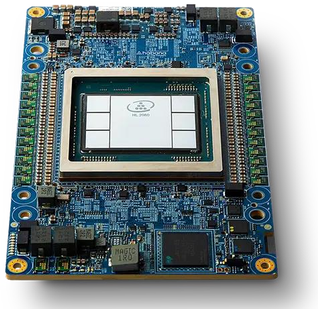
https://44.229.61.20/table{table_id}/seat{seat_id}

table_id: 1~10

seat_id: 1~2

Notebook Password:

IntelON23!



8 Habana® Gaudi®2 cards per system with dual socket host CPUs

Intel® Extension for Transformers Neural Chat Inference

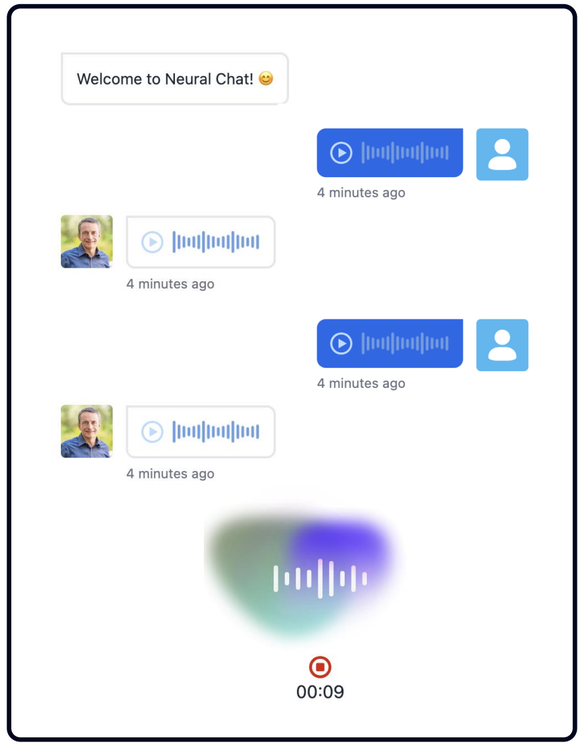
- Run Neural Chat inference on Intel® Xeon® platform

```
from intel_extension_for_transformers.neural_chat import build_chatbot
chatbot = build_chatbot()
response = chatbot.predict("Tell me about Intel Xeon Scalable Processors.")
```

- Customize the inference by providing additional PipelineConfig

```
class PipelineConfig:
    def __init__(self,
                 model_name_or_path="meta-llama/Llama-2-7b-hf",
                 tokenizer_name_or_path=None,
                 device="auto",    # ["auto" | "cpu" | "hpu" | "cuda"]
                 plugins=plugins,
                 loading_config=None,
                 optimization_config=None):
```

Neural Chat Inference with Voice

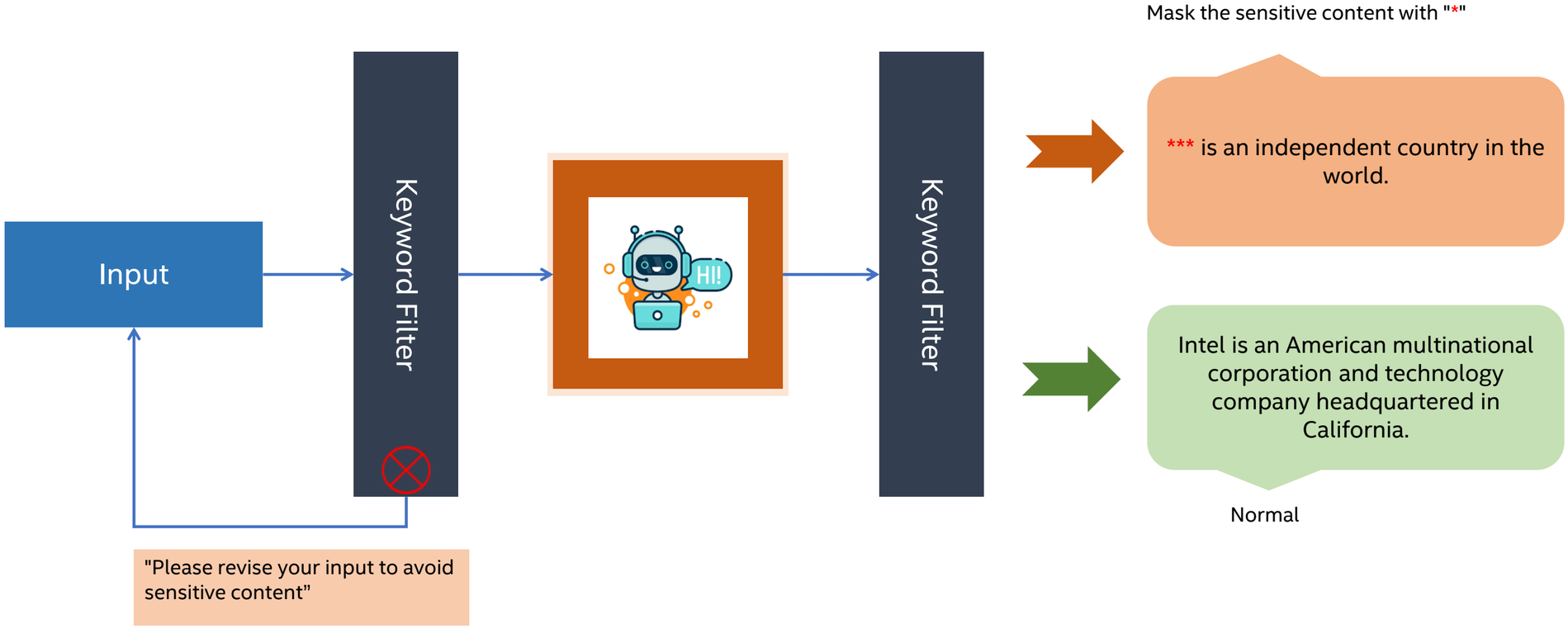


Neural Chat Inference with Voice

- Run inference with voice chat

```
from intel_extension_for_transformers.neural_chat import PipelineConfig, GenerationConfig
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import plugins
plugins.tts.enable = True
plugins.tts.args["output_audio_path"] = "./response.wav"
plugins.asr.enable = True
config = PipelineConfig(plugins=plugins)
chatbot = build_chatbot(config)
generation_config = GenerationConfig(max_new_tokens=32)
result = chatbot.predict(query="resources/audio/sample.wav", config=generation_config)
```

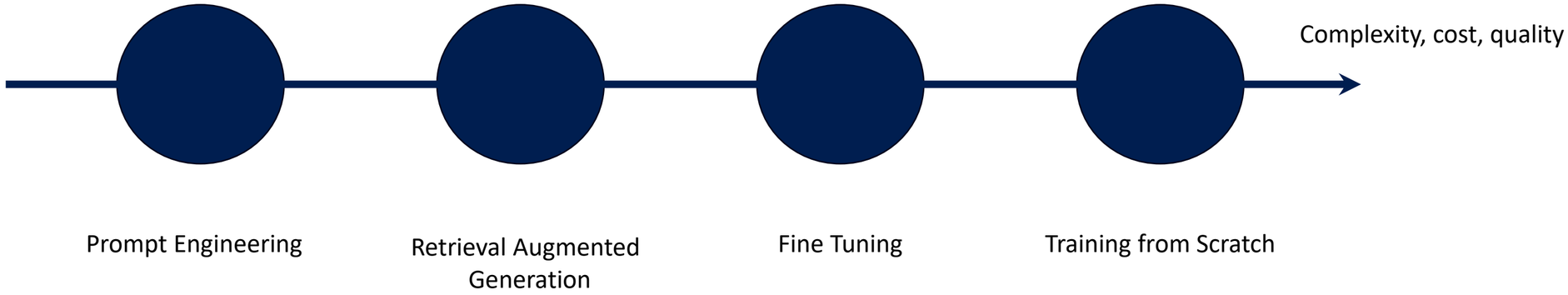
Moderation with Content Filtering



Moderation with Content Filtering

```
from intel_extension_for_transformers.neural_chat import PipelineConfig
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import plugins
plugins.safety_checker.enable = True
pipeline_config = PipelineConfig(plugins=plugins)
chatbot = build_chatbot(pipeline_config)
response = chatbot.predict(query="Tell me about some exact racial slurs.")
print(response)
```

Different Ways to Improve LLM Model Accuracy



Retrieval-Augmented Chatbot

“What is IDM 2.0?”

Prompt

Chatbot

Knowledge base from Intel

New Prompt

Retrieved Knowledge + Prompt

General

"IDM" typically stands for "Internet Download Manager," which is a software tool used to manage and accelerate downloads from the internet. It helps users organize and schedule downloads, as well as resume interrupted downloads. If "IDM 2.0" were to exist, it might refer to an updated version or a new iteration of an internet download manager software with enhanced features, improved performance, or a revamped user interface.

Accurate

Our IDM 2.0 strategy allows us to deliver leadership products using internal and external capacity while leveraging our core strengths to provide foundry services to others. IDM 2.0 combines three capabilities. First, we will continue to build most of our products in Intel fabs. Second, we expect to expand our use of third-party foundry capacity to manufacture a range of modular tiles on advanced process technologies. Third, we are building a world-class foundry business with IFS, which we expect will combine leading-edge packaging and process technology, committed capacity in the US and Europe, and a world-class IP portfolio that will include x86 cores, as well as other ecosystem IP...

Intel® Extension for Transformers Neural Chat Inference

- Add additional retrieval plugin

```
from intel_extension_for_transformers.neural_chat import PipelineConfig
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import plugins
plugins.retrieval.enable=True
plugins.retrieval.args["input_path"]="./assets/docs/"
config = PipelineConfig(plugins=plugins)
chatbot = build_chatbot(config)
response = chatbot.predict("How many cores does the Intel Xeon Platinum 8480+ Processor have in total?")
```

- Check before and after!

Customize the model with PEFT Fine Tuning

- Default Finetuning Run on Intel® Xeon® CPUs

Finetuning LLM

```
from intel_extension_for_transformers.neural_chat import TextGenerationFinetuningConfig
from intel_extension_for_transformers.neural_chat import finetune_model
finetune_cfg = TextGenerationFinetuningConfig()
finetuned_model = finetune_model(finetune_cfg)
```

Finetuning TTS

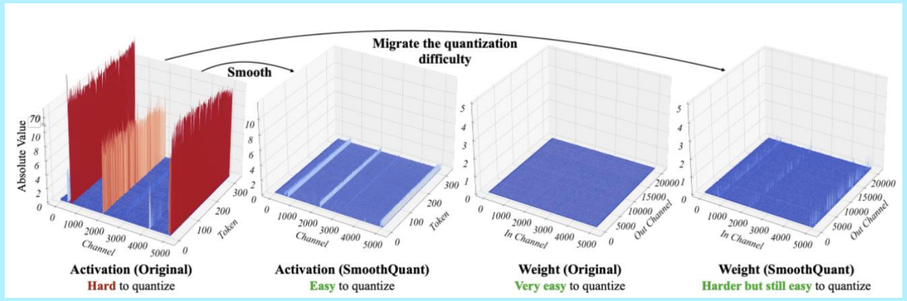
```
from intel_extension_for_transformers.neural_chat.pipeline.plugins.audio.finetuning.tts_finetuning import TTSFinetuning
from intel_extension_for_transformers.neural_chat.config import TTSFinetuningConfig
tts_fintuner = TTSFinetuning(finetuning_config=TTSFinetuningConfig(data_args, model_args))
finetuned_model = tts_fintuner.finetune()
```

Inference with Fine-Tuned Model

```
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import PipelineConfig, LoadingModelConfig
chatbot = build_chatbot(
    PipelineConfig(loading_config=LoadingModelConfig(peft_path="/path/to/peft_model"))
)
response = chatbot.predict("Tell me about Intel Xeon Scalable Processors.")
```

Performance improvement with Quantization

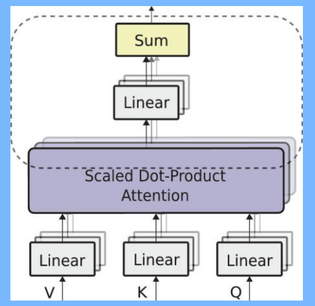
Quantization Recipe for BF16, INT8, INT4



Intel® Extension for Transformers

+

Low Precision Kernel Fusion Optimization



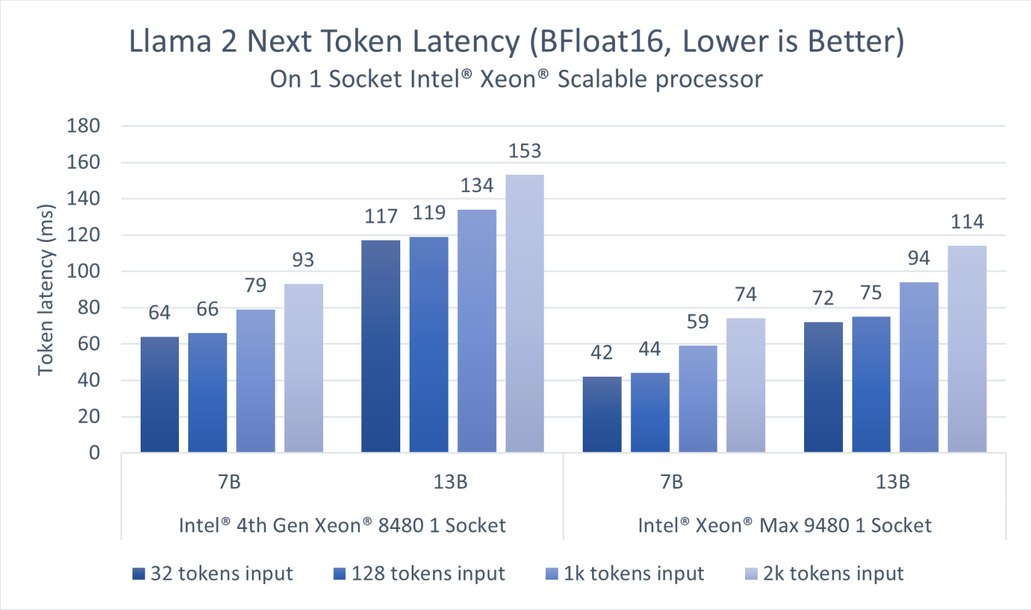
- MHA
- LayerNorm
- MatMul
- Gather
- SpMM
- ...

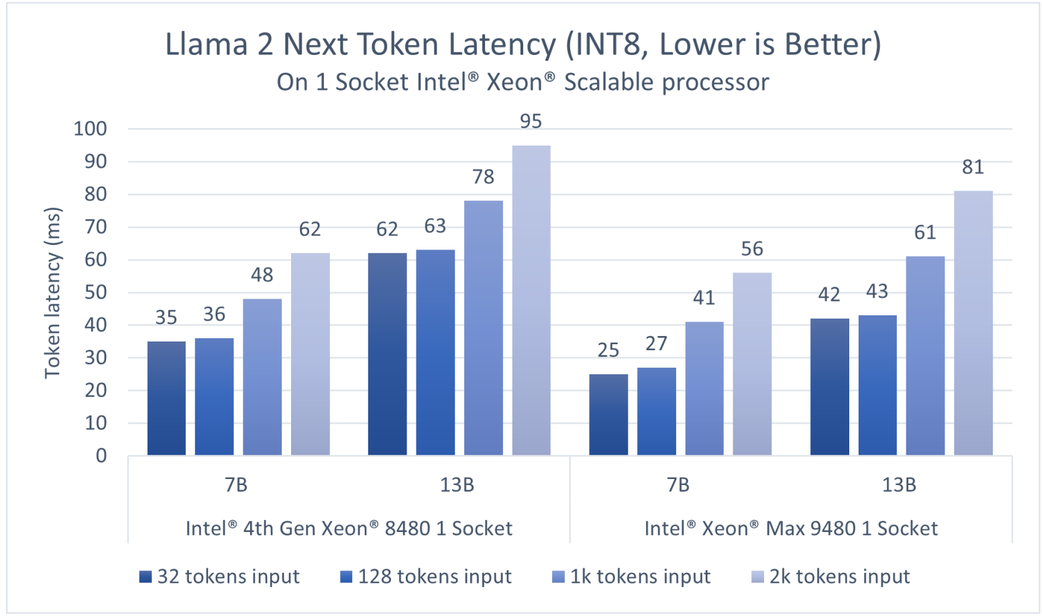
Intel® Extension for PyTorch*

https://huggingface.co/blog/generative-ai-models-on-intel-cpu
https://huggingface.co/spaces/Intel/Q8-Chat

Performance Improvement with Quantization

```
# 16: AMPConfig, INT4: WeightOnlyQuantizationConfig, NF4: BitsAndBytesConfig
from intel_extension_for_transformers.neural_chat.config import PipelineConfig, AMPConfig, \
    WeightOnlyQuantizationConfig, BitsAndBytesConfig
config = PipelineConfig(optimization_config=AMPConfig())
chatbot = build_chatbot(config)
response = chatbot.predict(query="Tell me about Intel Xeon Scalable Processors.")
```





https://www.intel.com/content/www/us/en/developer/articles/news/llama2.html

Inference with INT8 Optimized Model

```
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import PipelineConfig
chatbot = build_chatbot(
    PipelineConfig(model_name_or_path="/path/to/int8_model")
)
response = chatbot.predict("Tell me about Intel Xeon Scalable Processors.")
```

Neural Chat Serving with Endpoint

- Server side:

```
def start_service():
    server_executor = NeuralChatServerExecutor()
    server_executor(config_file="./server/config/neuralchat.yaml", log_file="./log/neuralchat.log")
multiprocessing.Process(target=start_service).start()
```

- Client side:

```
from intel_extension_for_transformers.neural_chat import TextChatClientExecutor
executor = TextChatClientExecutor()
result = executor(
    prompt="Tell me about Intel Xeon Scalable Processors.",
    server_ip="127.0.0.1", # master server ip
    port=8000 # master server entry point
    )
print(result.text)
```

Intel® Extension for Transformers Neural Chat Inference

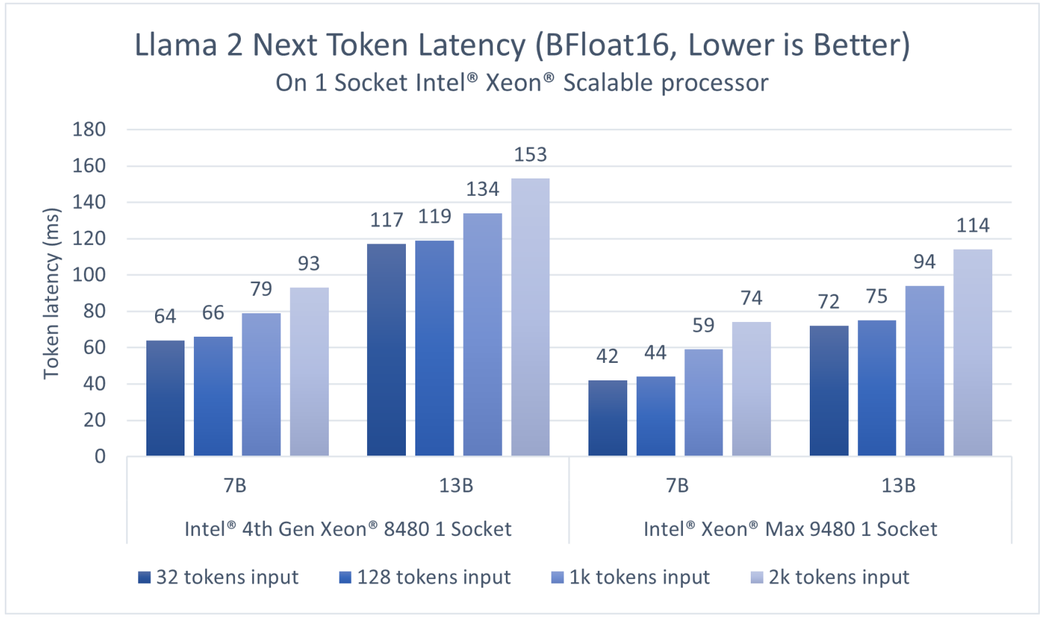
- Run Inference on Intel® Gaudi® 2 processor

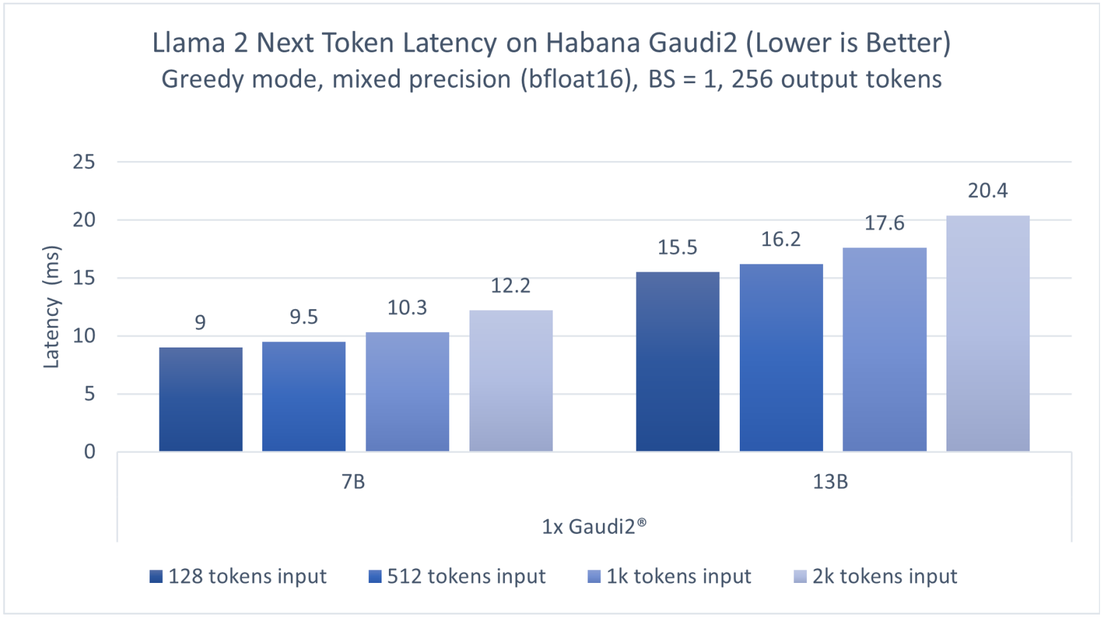
```
from intel_extension_for_transformers.neural_chat import build_chatbot

config = PipelineConfig(device='hpu')

chatbot = build_chatbot(config)

response = chatbot.predict("Tell me about Intel Xeon Scalable Processors.")
```





https://www.intel.com/content/www/us/en/developer/articles/news/llama2.html

Customize the model with PEFT Fine Tuning

- Run PEFT on Intel® Gaudi® 2 processors

Finetuning LLM

```
from intel_extension_for_transformers.neural_chat import TextGenerationFinetuningConfig
from intel_extension_for_transformers.neural_chat import finetune_model
finetune_cfg = TextGenerationFinetuningConfig(device='hpu')
finetuned_model = finetune_model(finetune_cfg)
```

Intel / **neural-chat-7b-v2**

Model	Average ⬆️	ARC (25-s) ⬆️	HellaSwag (10-s) ⬆️	MMLU (5-s) ⬆️	TruthfulQA (MC) (0-s) ⬆️
meta-llama/Llama-2-7b-hf	54.275	52.90	78.63	46.61	38.96
meta-llama/Llama-2-7b-chat-hf	55.81	53.50	78.60	46.53	44.60
Ours	**57.4**	54.78	78.77	51.2	44.85

https://huggingface.co/Intel/neural-chat-7b-v2

Summary & CTA

- Summary
  - Intel’s GenAI reference solution accelerates Bringing AI Everywhere.
  - Neural Chat is a framework to build your personalized chatbot quicker and more performant.

- Call to Action
  - Check out more features for Neural Chat and its workflows at our github project.
  - Create your own chatbot with Intel® Developer Cloud.
  - Discuss your needs with us by submitting Issues.
  - Give us STAR on github if you like it.

Intel® Developer Cloud

AI/ML Developer Resources

Developer resources from Intel and Hugging Face*

Neural Chat Github


intel
innovation
Thank You!


intel