Vegi(ocr) - Voice Enhanced Gaming Interface

Abstract:
The objective of this project is to deploy optimized and quantized Optical Character Recognition (OCR) models on AMD Ryzen AI NPU and CPU.
The project will involve a comprehensive comparison of model inference across NPU, CPU and GPU with a particular focus on identifying and addressing the performance shortcomings of the Ryzen AI NPU in handling Transformer-based models.

Introduction:

Optical Character Recognition (OCR) technology has evolved significantly, enabling the efficient digitization of printed text into editable and searchable formats. EasyOCR and the Donut transformer represent the forefront of OCR advancements, known for their robustness and powerful transformer-based architecture, respectively. Deploying these models on AMD Ryzen AI NPU, designed to accelerate AI workloads, offers potential performance gains but also presents unique challenges.

Motivation:

The motivation behind this project is multifaceted:

- Reduced Latency: Achieving instantaneous processing and response times is essential for real-time applications like gaming and video editing.
- Enhanced Privacy: Local data processing minimizes the risk of data breaches, crucial for handling sensitive information in healthcare and finance.
- Cost Efficiency: Lowering operational costs by reducing cloud service usage and dependence on constant internet connectivity.
- Improved Reliability: Ensuring functionality independent of internet connections is vital for remote areas or situations with unstable connectivity.
- Energy Efficiency: Optimizing power consumption through local processing extends battery life in mobile devices and enhances overall energy efficiency.

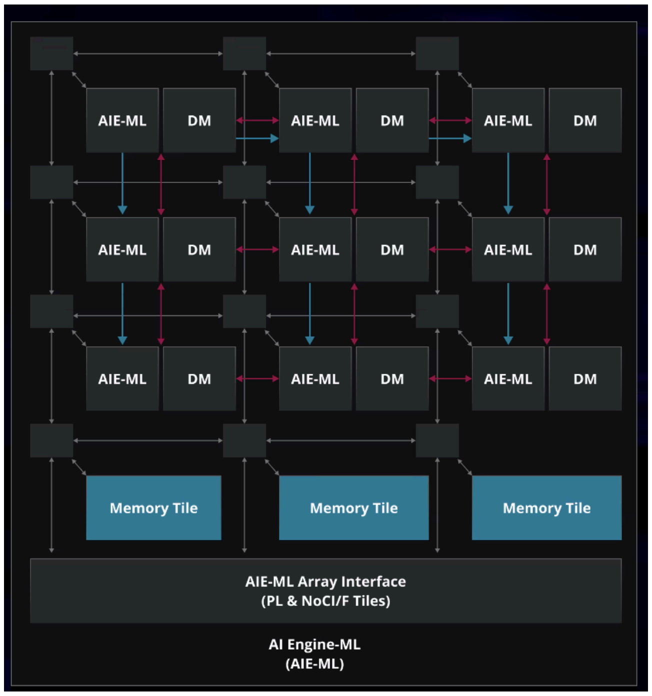
Ryzen AI Architecture:

The Ryzen AI architecture features AI Engine tiles with vector processors for parallel processing, scalar processors for sequential tasks, and local memories to reduce external data access. Memory tiles provide additional local storage. Key interfaces include the Memory Interface for connecting AI Engine tiles to memory tiles, the Stream Interface for managing data flow between tiles, and the Cascade Interface for internal data movement. The AIE-ML Array Interface links AI Engine tiles with components like

Programmable Logic (PL) and Network on Chip (NoC) tiles.

This tiled array organization allows for independent, parallel AI task processing, while on-chip memories enhance efficiency by minimizing external fetches. Custom dataflows support efficient, low-power computing tailored for AI and signal processing tasks. The architecture offers energy efficiency, improved performance through parallel processing, and scalability with its modular design, making the AMD XDNA AI Engine highly efficient and optimized for complex AI applications.

While running AI models on a CPU or GPU alone can drain the battery fast, with a Ryzen AI powered-laptop, AI models operate on the embedded NPU, freeing-up CPU and GPU resources for other compute tasks.



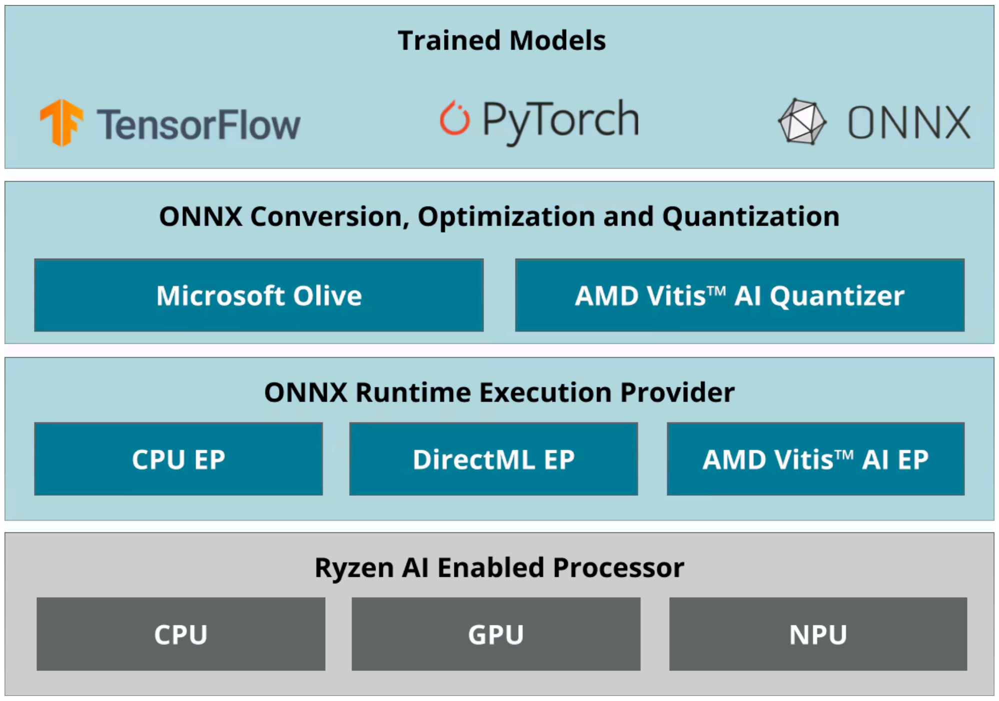
RyzenAI Software:
AMD Ryzen™ AI Software includes the tools and runtime libraries for optimizing and deploying AI inference on AMD Ryzen™ AI powered PCs1. Ryzen AI software enables applications to run on the neural processing unit (NPU).

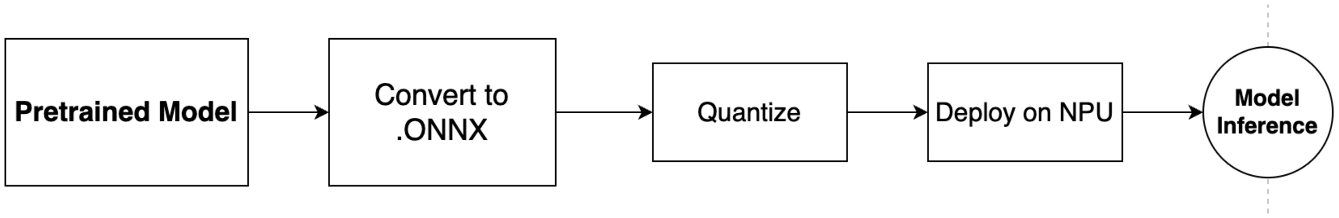
Ryzen AI Development Flow:

a. **ONNX Development Flow:**

Developing AI applications for Ryzen AI involves three steps: Train, Quantize, and Deploy. First, developers select or create and train a model in PyTorch, TensorFlow, or ONNX models in the cloud. Next, the model is quantized into INT8 using the AMD Vitis™ AI quantizer or Microsoft Olive with Vitis AI as a plug-in, saving it in ONNX format. Finally, the ONNX Runtime Vitis AI Execution Provider (EP) partitions, compiles, and executes the quantized models on Ryzen

AI hardware, ensuring optimal performance with low power consumption.

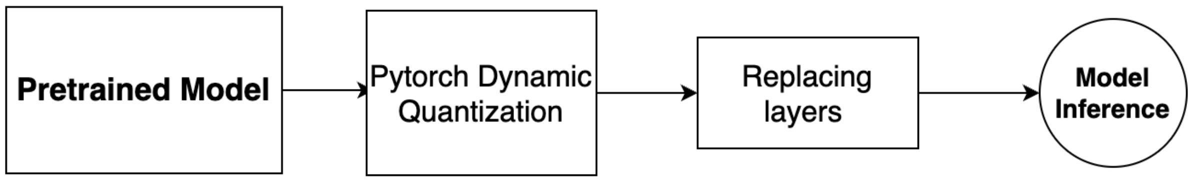




b. Pytorch Development Flow:

The PyTorch development flow for Ryzen AI involves dynamic quantization and model transformation. Starting with a pre-trained model, PyTorch Dynamic Quantization is applied, which converts the model's parameters to lower-precision representations like INT8. This step enhances power efficiency and performance.

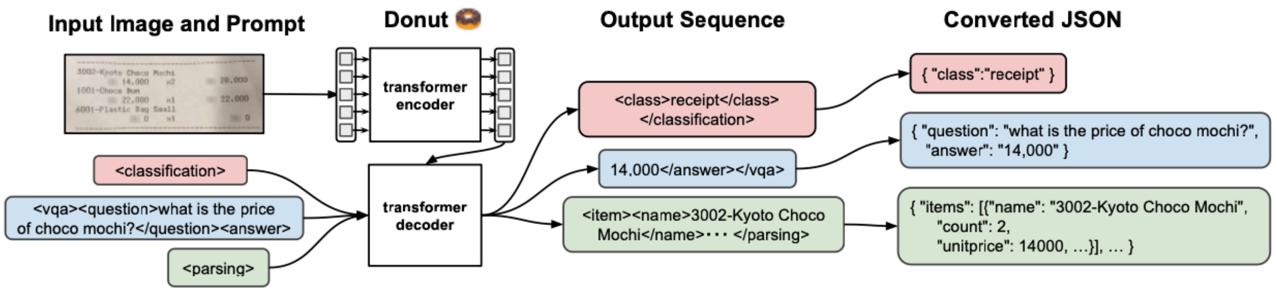
After quantization, the model undergoes a transformation where quantized linear layers are replaced with Ryzen AI-compatible `QLinear` layers. This replacement optimizes the model for inference on Ryzen AI hardware, ensuring efficient execution and low power consumption. The final transformed model is ready for deployment, leveraging the specialized capabilities of Ryzen AI for improved performance.



OCR(optical Character Recognition):

Optical Character Recognition (OCR) technology converts different types of documents, such as scanned paper documents, PDFs, or images captured by a digital camera, into editable and searchable data. This project aims to optimize and deploy OCR models on AMD Ryzen AI hardware to enhance performance and efficiency. Some of the popular models are PyTesseract, EasyOCR, TrOCR, DONUT, Paddle OCR.

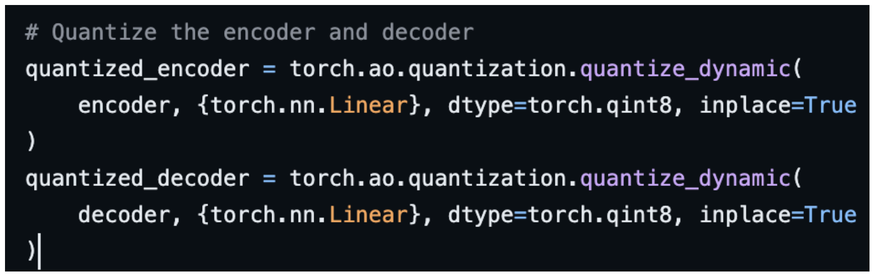
DONUT (Document Object Understanding Transformer):


Input Image and Prompt
Donut
Output Sequence
Converted JSON
transformer encoder
transformer decoder
<classification>
<vqa><question>what is the price of choco mochi?</question><answer>
<parsing>
<class>receipt</class></classification>
14,000</answer></vqa>
<item><name>3002-Kyoto Choco Mochi</name>··· </parsing>
{ "class":"receipt" }
{ "question": "what is the price of choco mochi?", "answer": "14,000" }
{ "items": [{"name": "3002-Kyoto Choco Mochi", "count": 2, "unitprice": 14000, …}], … }

The DONUT (Document Understanding Transformer) is a state-of-the-art OCR model designed for comprehensive document understanding. It leverages transformer architecture to accurately extract and interpret textual information from complex documents, making it highly effective for tasks requiring detailed text recognition and analysis.

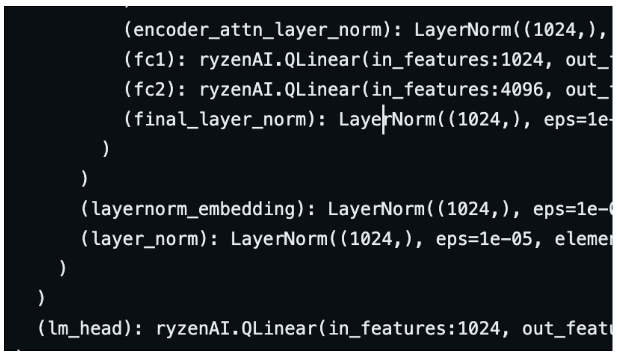
a. **Pytorch Flow:**

   i. *Dynamic Quantization using pytorch*



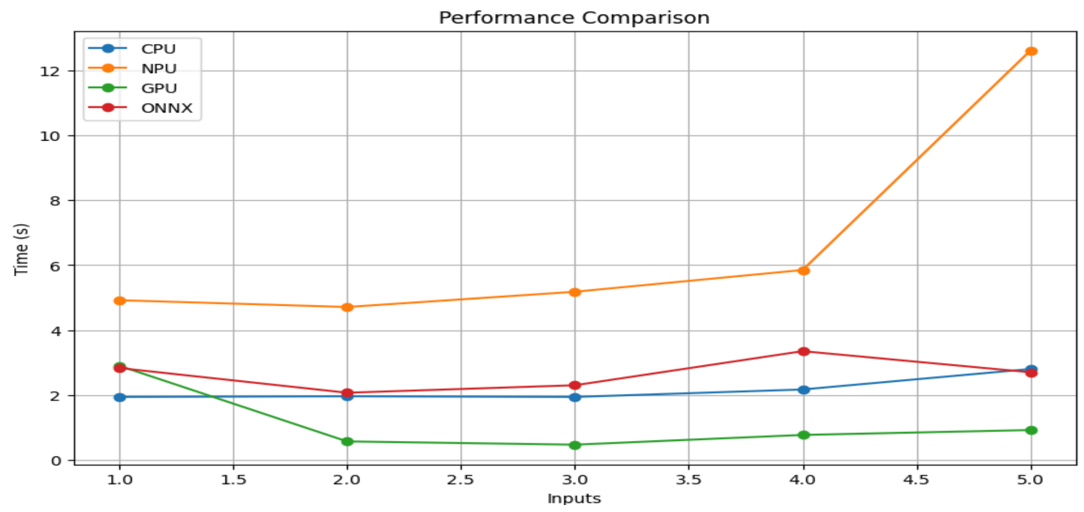
```
# Quantize the encoder and decoder
quantized_encoder = torch.ao.quantization.quantize_dynamic(
    encoder, {torch.nn.Linear}, dtype=torch.qint8, inplace=True
)
quantized_decoder = torch.ao.quantization.quantize_dynamic(
    decoder, {torch.nn.Linear}, dtype=torch.qint8, inplace=True
)
```

   ii. *Node Replacing using QLinear*



```
          (encoder_attn_layer_norm): LayerNorm((1024,),
          (fc1): ryzenAI.QLinear(in_features:1024, out_
          (fc2): ryzenAI.QLinear(in_features:4096, out_
          (final_layer_norm): LayerNorm((1024,), eps=1e-
        )
      )
      (layernorm_embedding): LayerNorm((1024,), eps=1e-
      (layer_norm): LayerNorm((1024,), eps=1e-05, elemen
    )
  )
  (lm_head): ryzenAI.QLinear(in_features:1024, out_feat
```

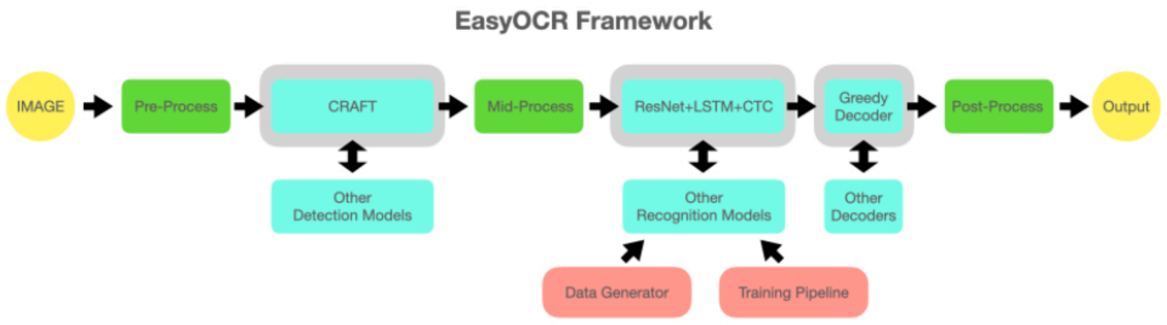
Results comparison:



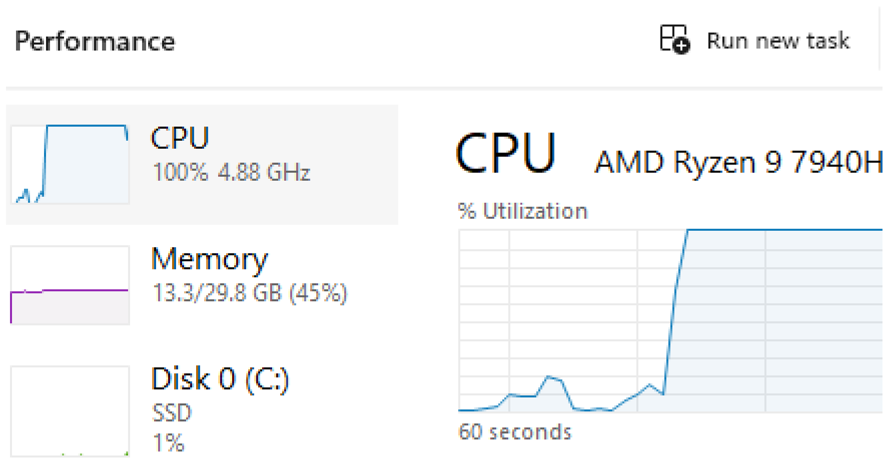
Note: More information on code could be found on the github.

EasyOCR:

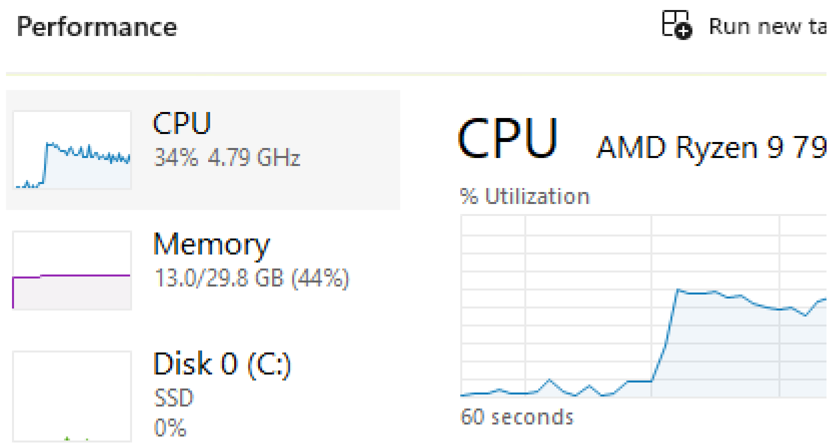
EasyOCR is an advanced Optical Character Recognition (OCR) tool built on deep learning frameworks, supporting over 80 languages. It utilizes a combination of convolutional neural networks (CNN) and recurrent neural networks (RNN) for text detection and recognition, offering high accuracy and robustness in extracting text from images. EasyOCR is widely used for its efficiency and ease of integration into various applications.



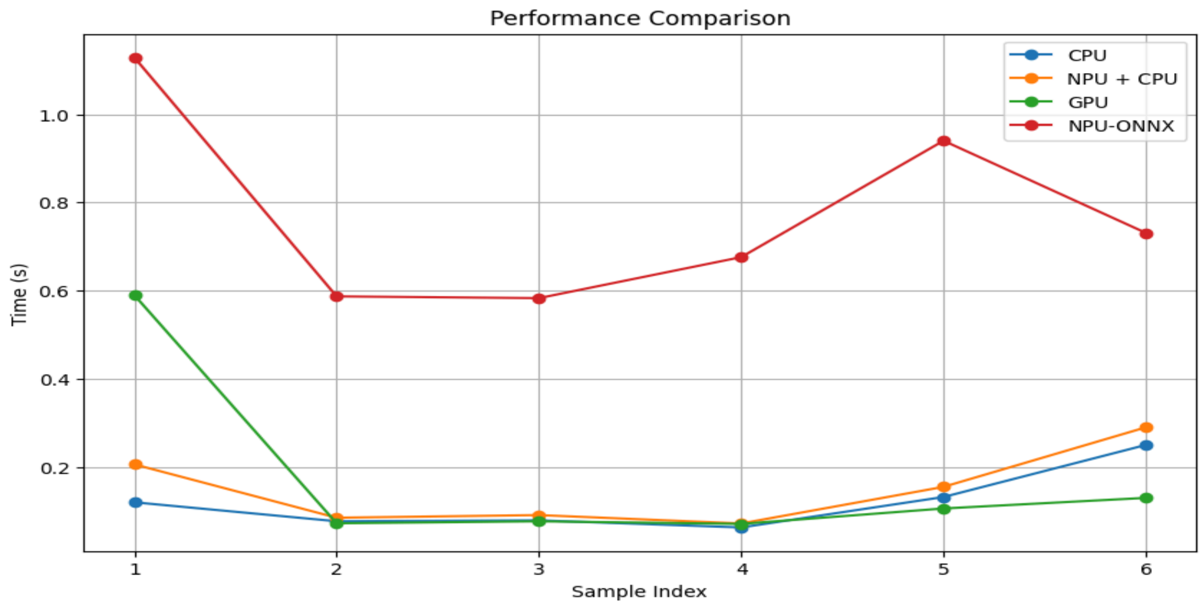
CPU Utilization without NPU:



CPU Utilization with NPU:



Results Comparison:



Conclusion:

In this project, we aimed to deploy optimized and quantized Optical Character Recognition (OCR) models on AMD Ryzen AI NPU and CPU, conducting a comprehensive comparison of model inference across NPU, CPU, and GPU. Our focus was on identifying and addressing the performance shortcomings of the Ryzen AI NPU in handling Transformer-based models.

Future work will include obtaining potential driver updates from AMD to improve NPU utilization, further optimizing models, enhancing accuracy through fine-tuning, and exploring different quantization techniques to boost model performance. This ongoing effort will continue to refine and advance the deployment of AI models on AMD Ryzen AI platforms.