Programming New AI Accelerators for Scientific Computing

Petro Junior Milan
Principal AI Engineer, SambaNova Systems

Tutorial at Supercomputing 2024
17 November 2024, Atlanta, GA

Agenda

1. SambaNova: Who We Are
2. Hardware and Software Architecture
3. AI Inference using SambaNova Cloud
4. AI Training on SambaNova DataScale

1. SambaNova: Who We Are

SambaNova SYSTEMS

Who We Are

Snapshot

- Founded in 2017 by industry luminaries and originated at Stanford University
- Fully integrated generative AI platform, from 4th generation hardware to pre-trained models
- $1B+ funding raised

Founded by pioneers in AI

Lip-Bu Tan
Executive Chairman

Rodrigo Liang
Co-founder & CEO

Kunle Olukotun
Co-founder & Chief Technologist & Stanford Professor

Christopher Ré
Co-founder & Stanford Professor

Sophisticated, long-term investors

BlackRock Capital Investment Corporation™ · SoftBank Investment Advisers · TEMASEK · G/ · intel Capital · SAMSUNG CATALYST FUND · GIC · Micron · SK telecom · WALDEN INTERNATIONAL

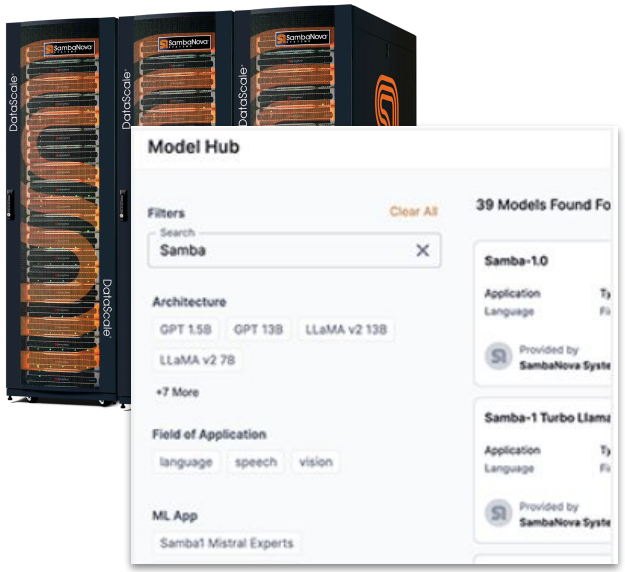
SambaNova SYSTEMS

SambaNova Delivers

SambaNova DataScale

Fully integrated hardware-software AI system (link)

SambaNova Suite



Secure, on-premises AI platform for training and inference (link)

SambaNova Cloud

API service for inference at record-breaking speeds (link)

SambaNova SYSTEMS

2. Hardware and Software Architecture

SN40L: SambaNova's 4th Gen AI Chip

"Cerulean" Architecture-based Reconfigurable Dataflow Unit (RDU)

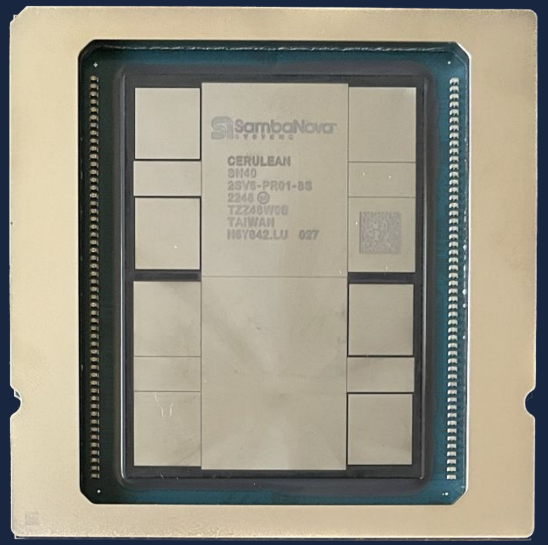
5nm TSMC

102B Transistors

1,040 RDU Cores

638 TFLOPS (bf16)

SambaNova SYSTEMS
Cerulean SN40L RDU



3-tier Dataflow Memory

520 MB
On-Chip Memory

64 GB
High Bandwidth Memory

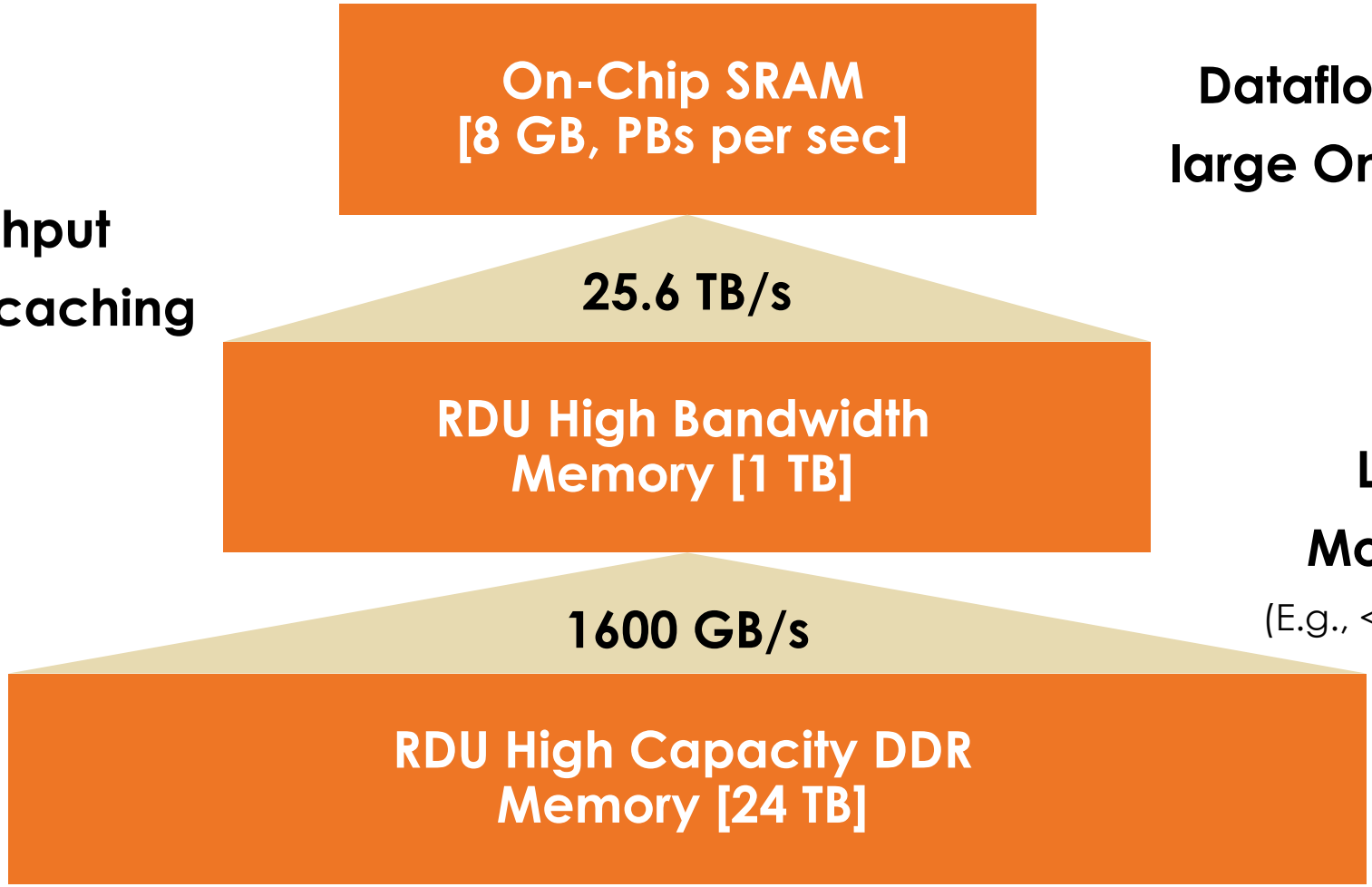
1.5 TB
High Capacity Memory

Generative AI Training and Inference

SambaNova SYSTEMS

SN40L: Three Tier Memory Architecture

3-tier Memory System with SRAM, HBM, and DDR



Dataflow enabled by large On-Chip Memory

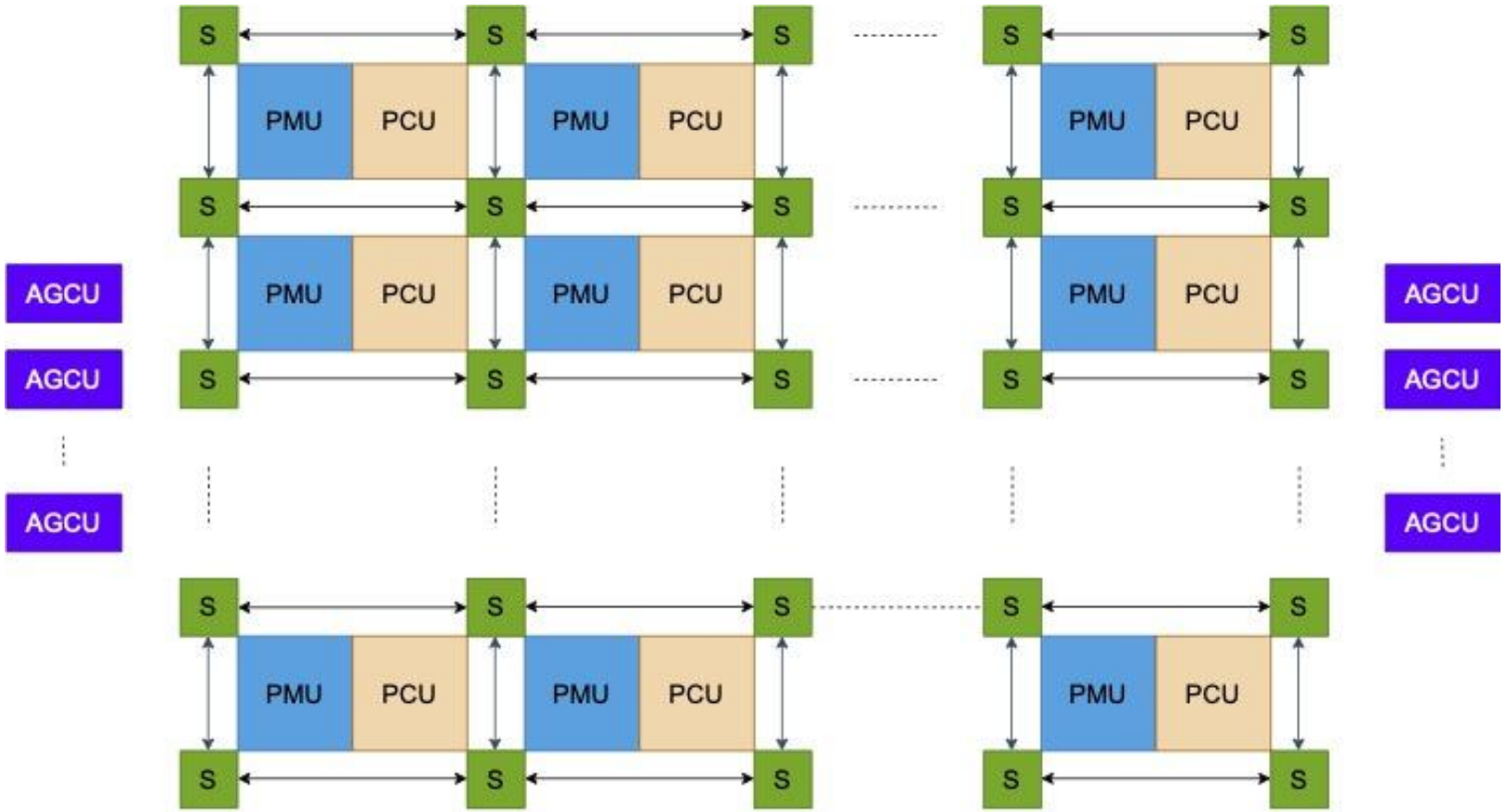
High throughput inference with caching

Low Latency Model Switching

(E.g., < 0.01s for llama3.1 8B)

SambaNova SYSTEMS

SN40L: Tile Architecture



1040 PCUs and PMUs

PCU: Compute unit

PMU: Memory unit

S: Mesh switches

AGCU: Portal to off-chip memory and IO

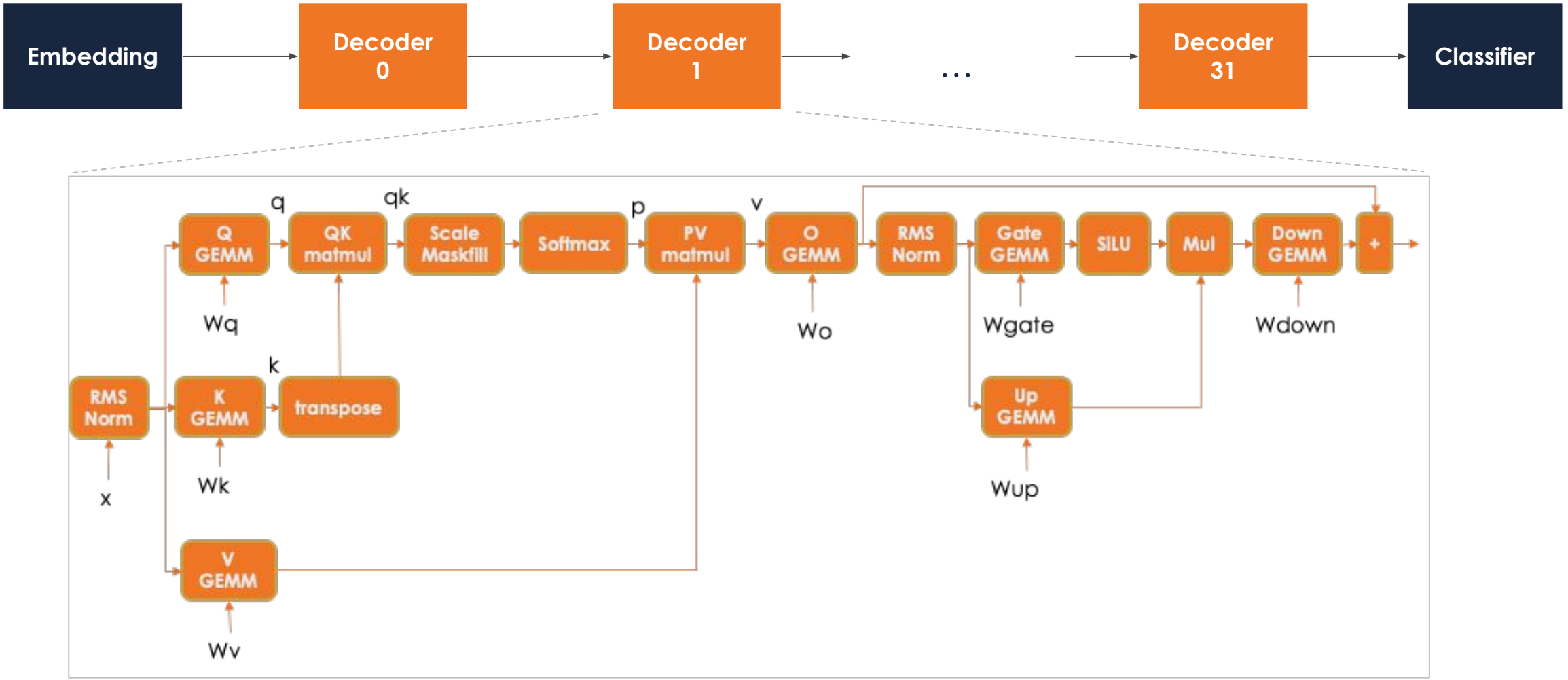
SambaNova SYSTEMS

Structure of a Transformer

Example: Llama3.1 8B

Structure of a Transformer

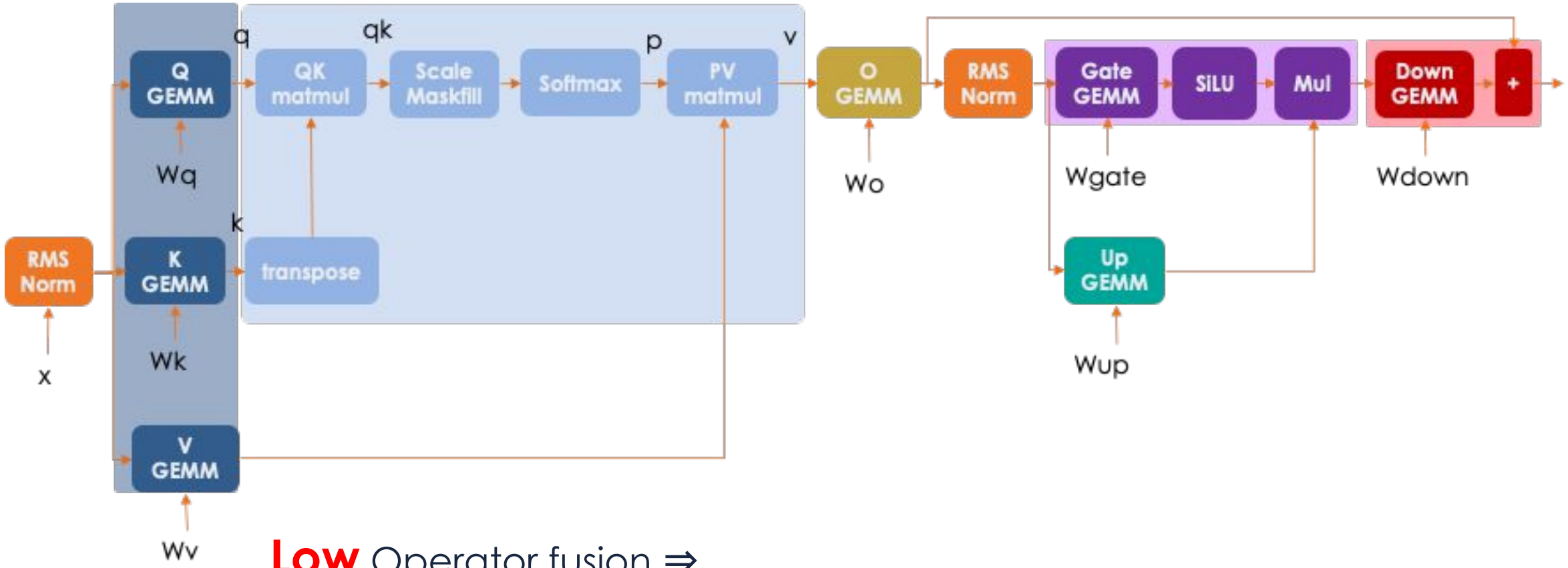
Example: Llama3.1 8B



Decoder: Conventional Fusion (GPU)

Example: Llama3.1 8B

1 colored group == Fused ops == 1 kernel call



Low Operator fusion ⇒

High Kernel Launch Overheads

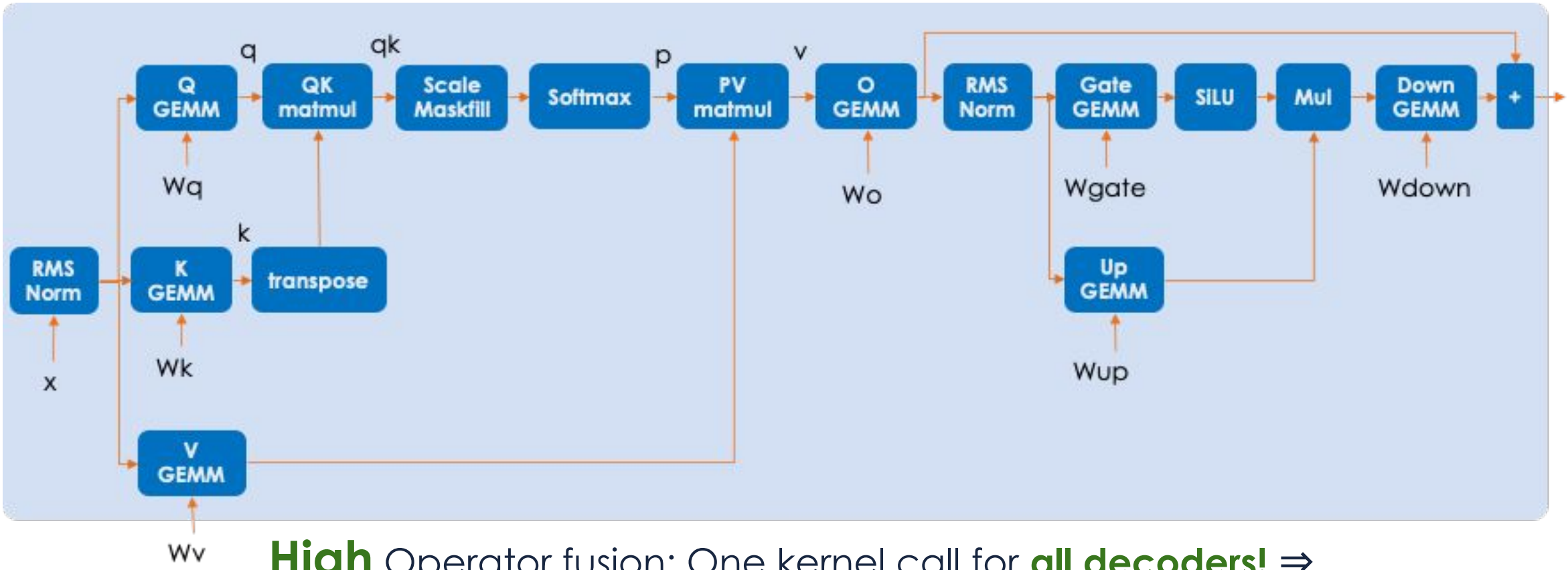
Low data locality

SambaNova SYSTEMS

Decoder: Streaming Dataflow (RDU)

Example: Llama3.1 8B

1 colored group == Fused ops == 1 kernel call



High Operator fusion: One kernel call for **all decoders!** ⇒

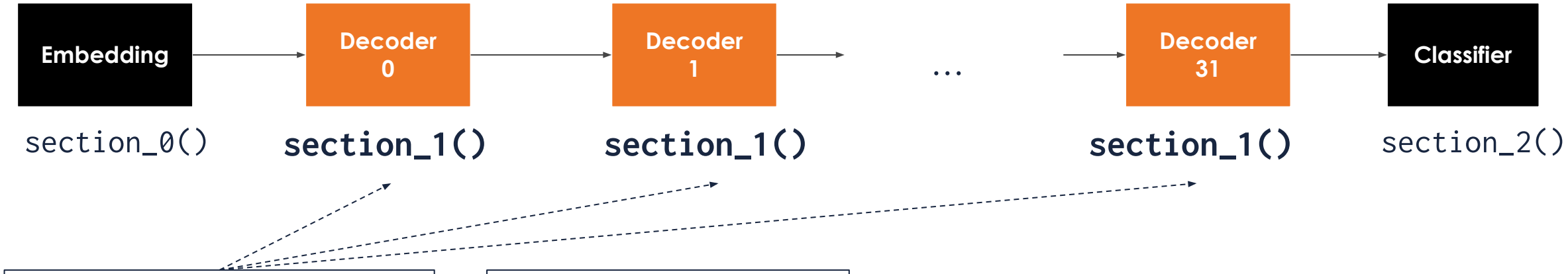
Zero Kernel Launch Overheads

High data locality

SambaNova SYSTEMS

RDU Hardware Graph Orchestration: Temporal Fusion

Example: Llama3.1 8B



Single Section Decoder
+
Repeated Section Calls
=
New fusion opportunities!

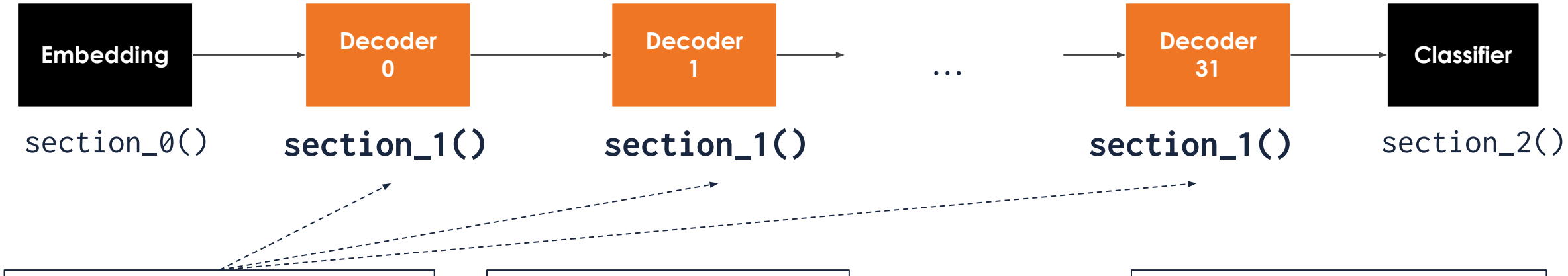
```
section_1(arg):
  Foo;
  Bar;

// Repeat 32 times
section_1(arg0)
section_1(arg1)
…
section_1(arg31)
```

SambaNova SYSTEMS

RDU Hardware Graph Orchestration: Temporal Fusion

Example: Llama3.1 8B



Single Section Decoder
+
Repeated Section Calls
=
New fusion opportunities!

```
section_1(arg):
  Foo;
  Bar;

// Repeat 32 times
section_1(arg0)
section_1(arg1)
…
section_1(arg31)
```

LoopConversion

```
section_1(args[]):
  for i in range(32):
    Foo;
    Bar;

// 1 time
section_1(args)
```

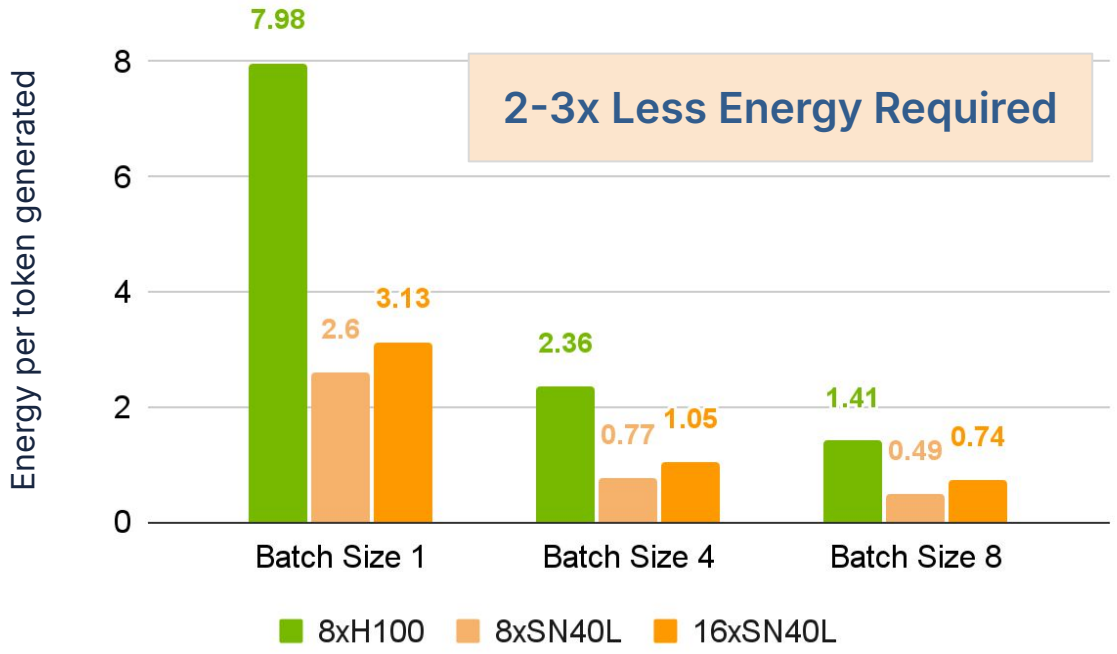
1 kernel launch for all decoders!

SambaNova SYSTEMS

2-5x Better Efficiency for GenAI Inference

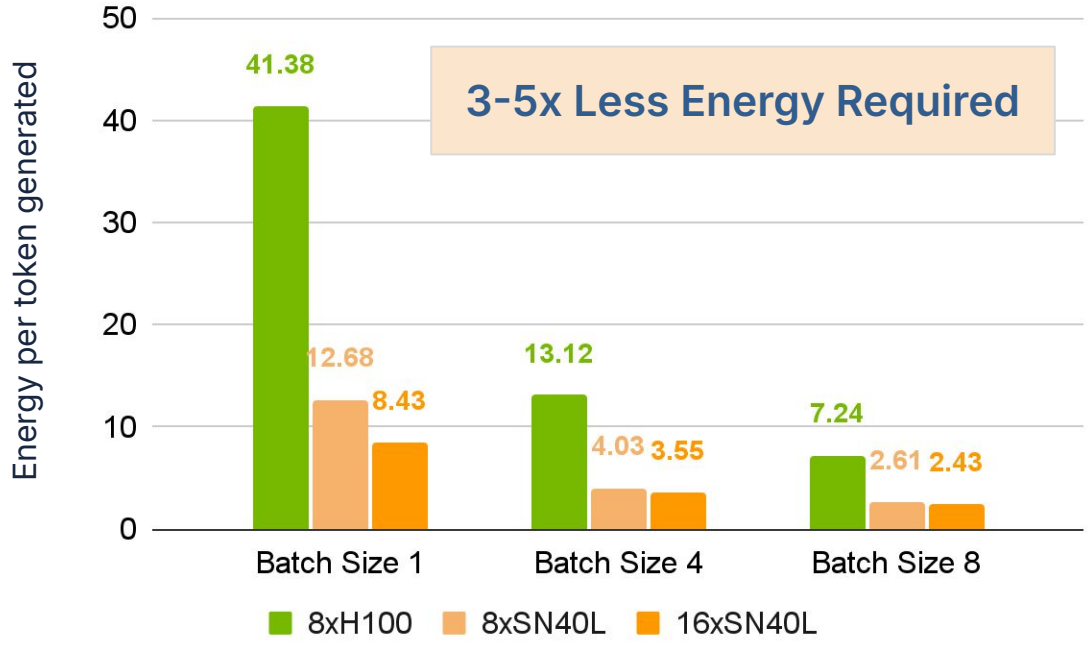
Llama 3.1 8B Inference

(total measured GPU or RDU energy per token generated)



Llama 3.1 70B Inference

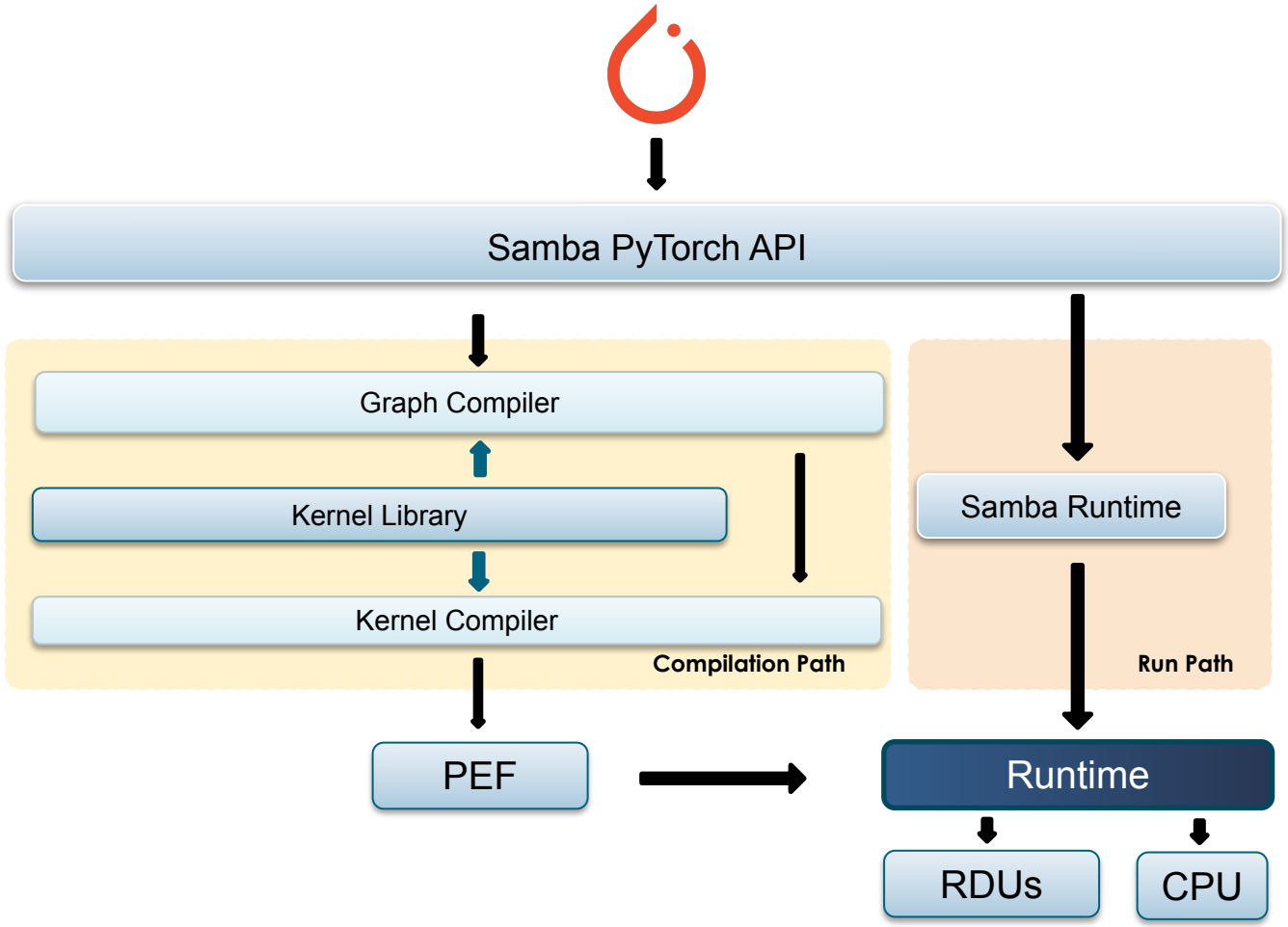
(total measured GPU or RDU energy per token generated)



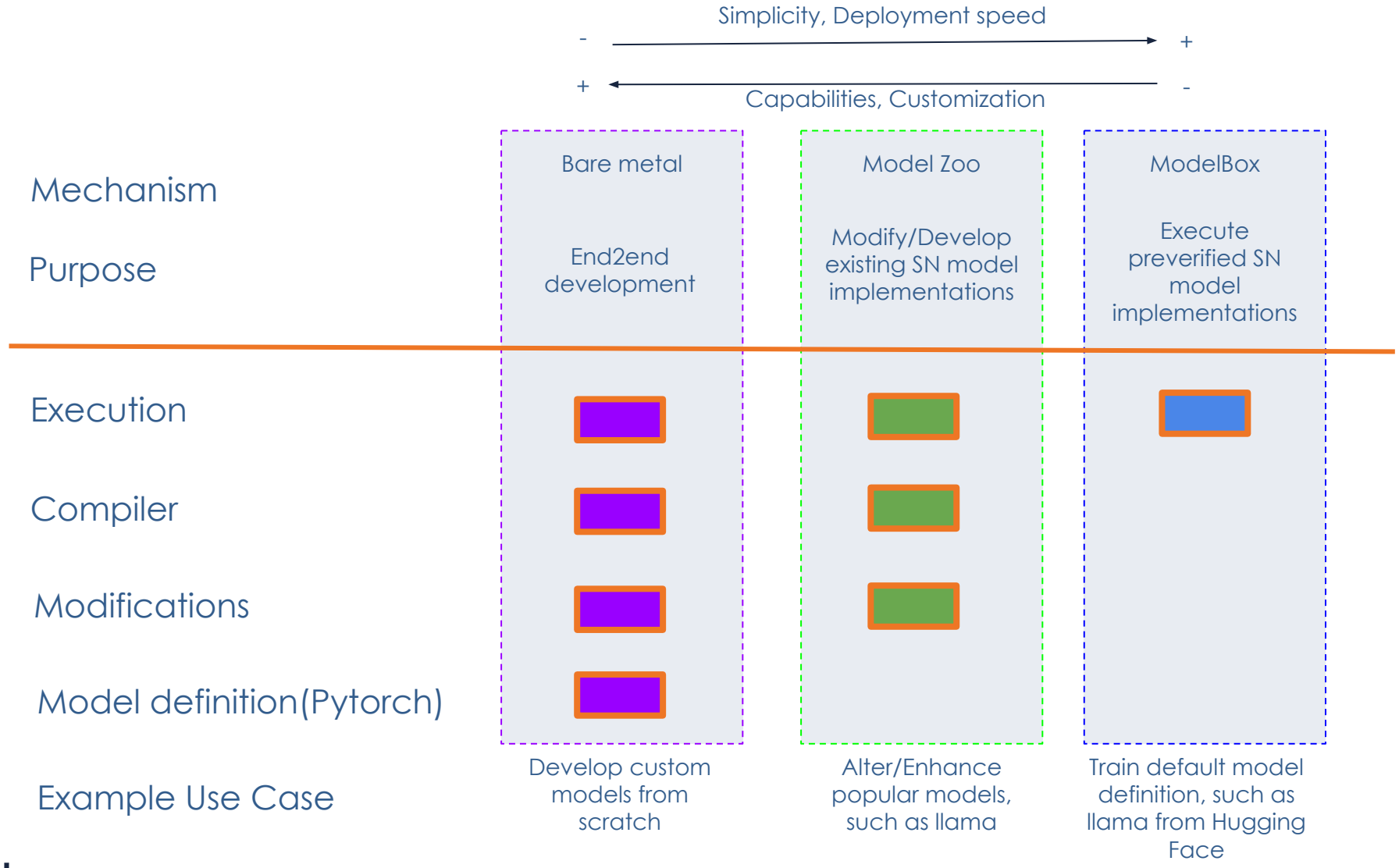
SambaNova SYSTEMS

Samba Compilation Flow

- **Samba**
  - + SambaNova PyTorch compilation & run APIs
- **Graph compiler**
  - + High-level ML graph transformation & optimizations
- **Kernel compiler**
  - + Low-level RDU operator kernel transformation & optimizations
- **Kernel library**
  - + RDU operator implementations



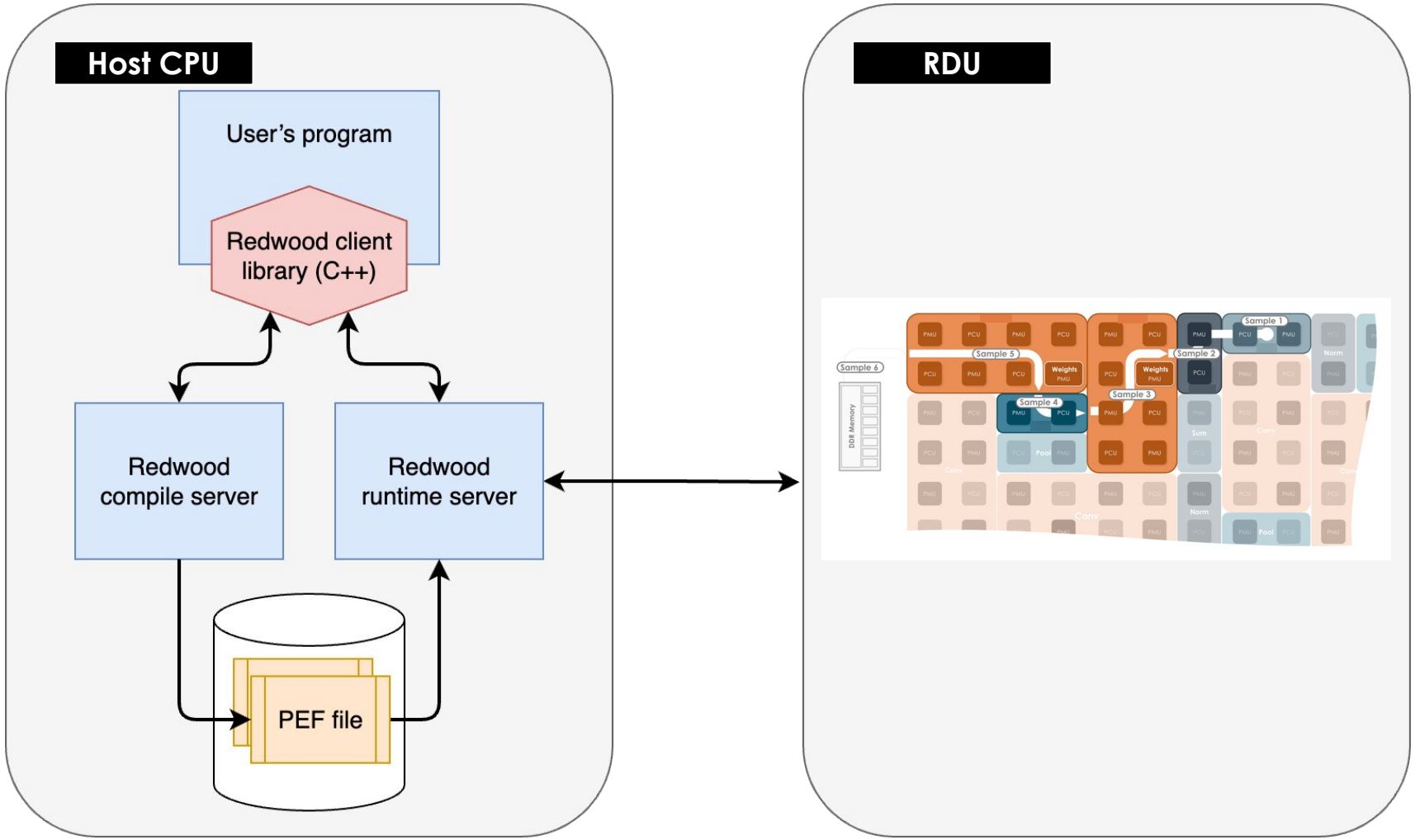
Distribution Options: Bare Metal vs Model Zoo vs Model Box



Redwood: A C++ SDK for the RDU

- Redwood lets users
  - **Specify tensor functions** in C++
  - **Compile them from** C++
  - **Run them from** C++
  - **Tune** them from C++, aided by SambaTune
  - **Debug** them from C++

Redwood: System Components



Redwood: Design Goals and Status

Design Goals

- Enable expert developers to exploit the capabilities of RDUs
- For new innovation vs. porting
- Example use cases
  - + Express critical portion of application as tensor operations and offload to RDUs
  - + Develop high-performance ML operators

Status

- Ramping up internal use
- Early release with select customers
- Feedback collection to inform design choices
- Public release coming soon

AI Computing at Scale in Multiple US National Laboratories

- Fine-tuning and inferencing large language models to apply AI to complex science problems
- Code generation
- Trustworthiness and security
- Drug discovery
- Climate science
- Brain mapping
- Physics simulations
- Cancer research

3. AI Inference using SambaNova Cloud

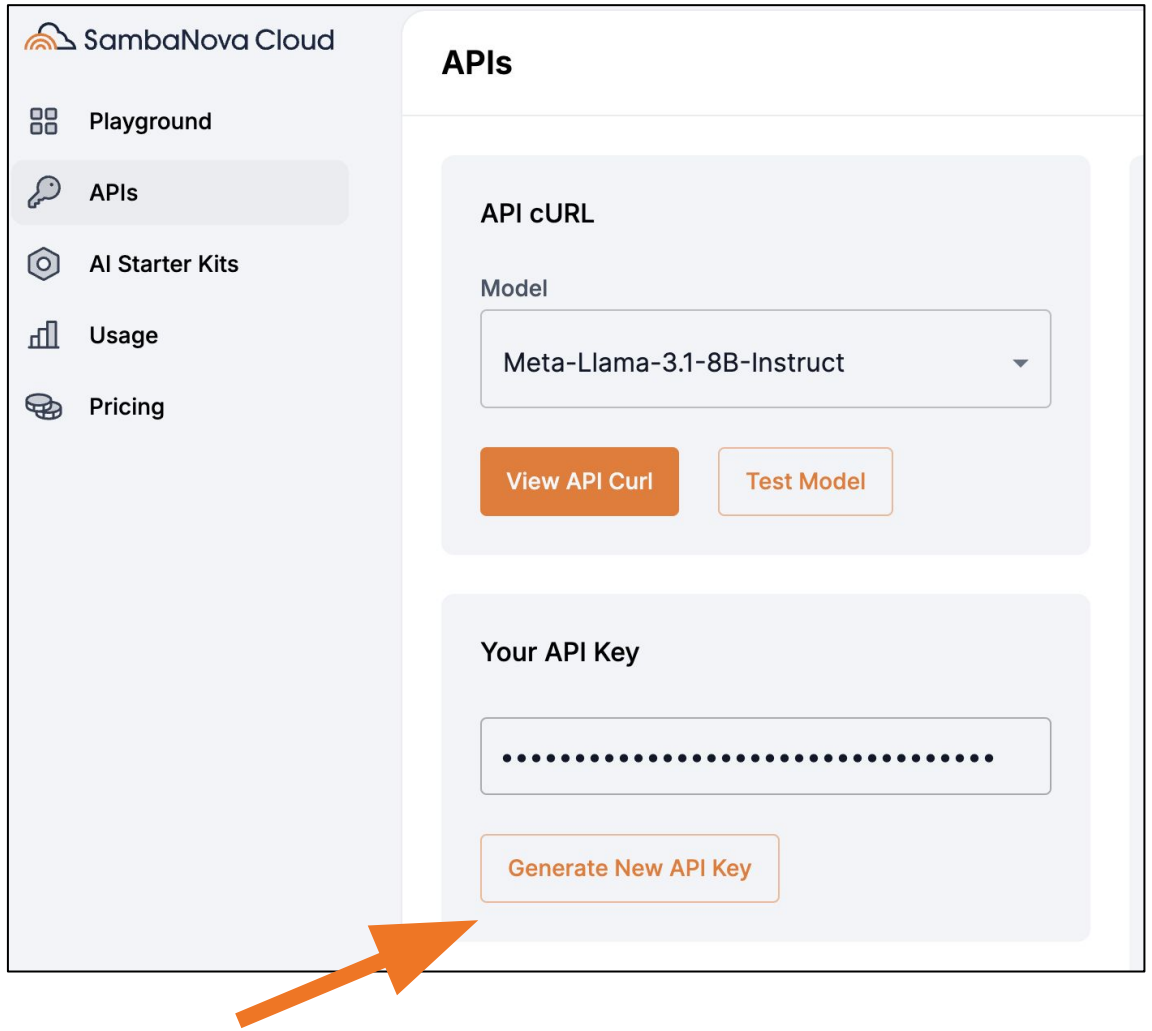
SambaNova SYSTEMS

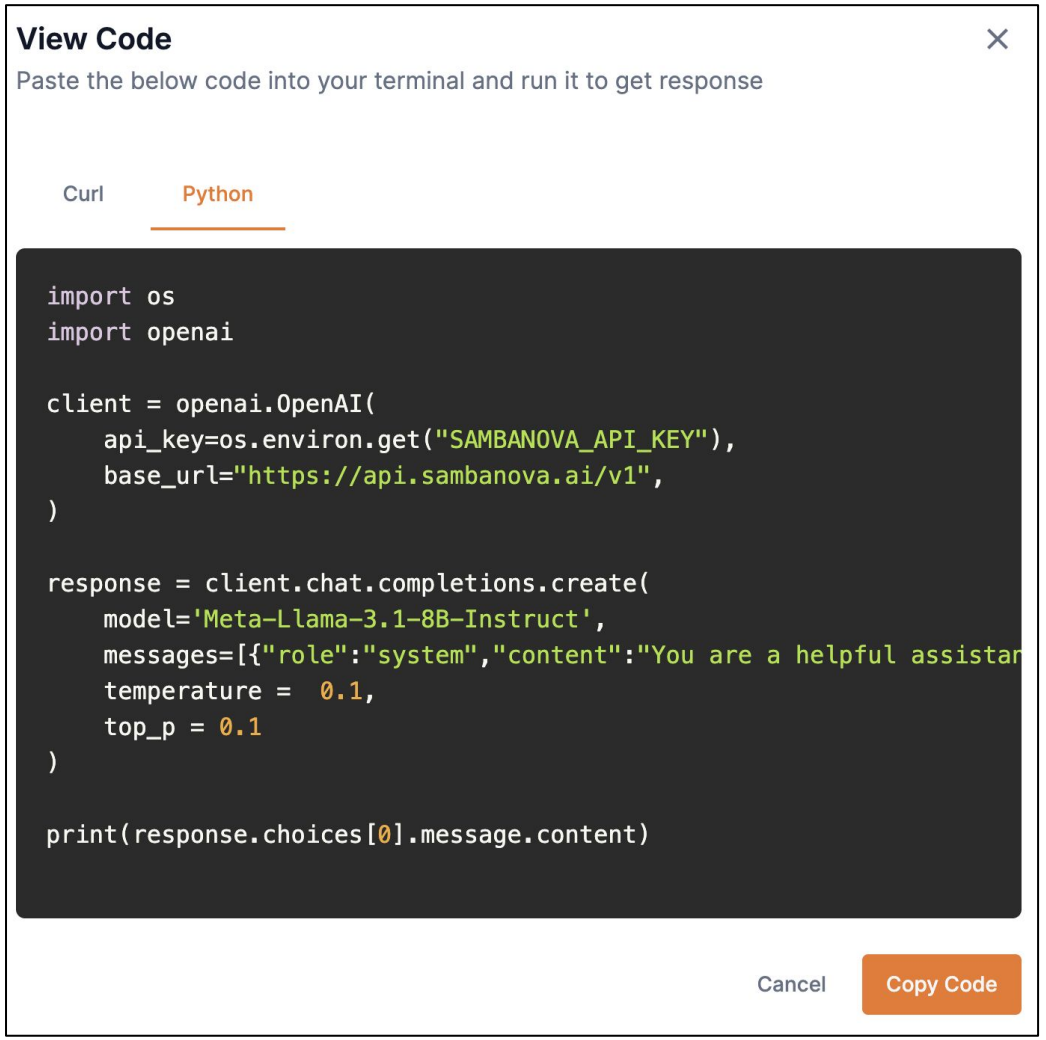
SambaNova Cloud

Over 10X Faster Tokens/Second/User

	SambaNova	GPU
Llama 3.2 1B 16-bit	2477	304
Llama 3.1 8B 16-bit	1066	93
Llama 3.1 70B 16-bit	460	32
Llama 3.1 405B 16-bit	200	14

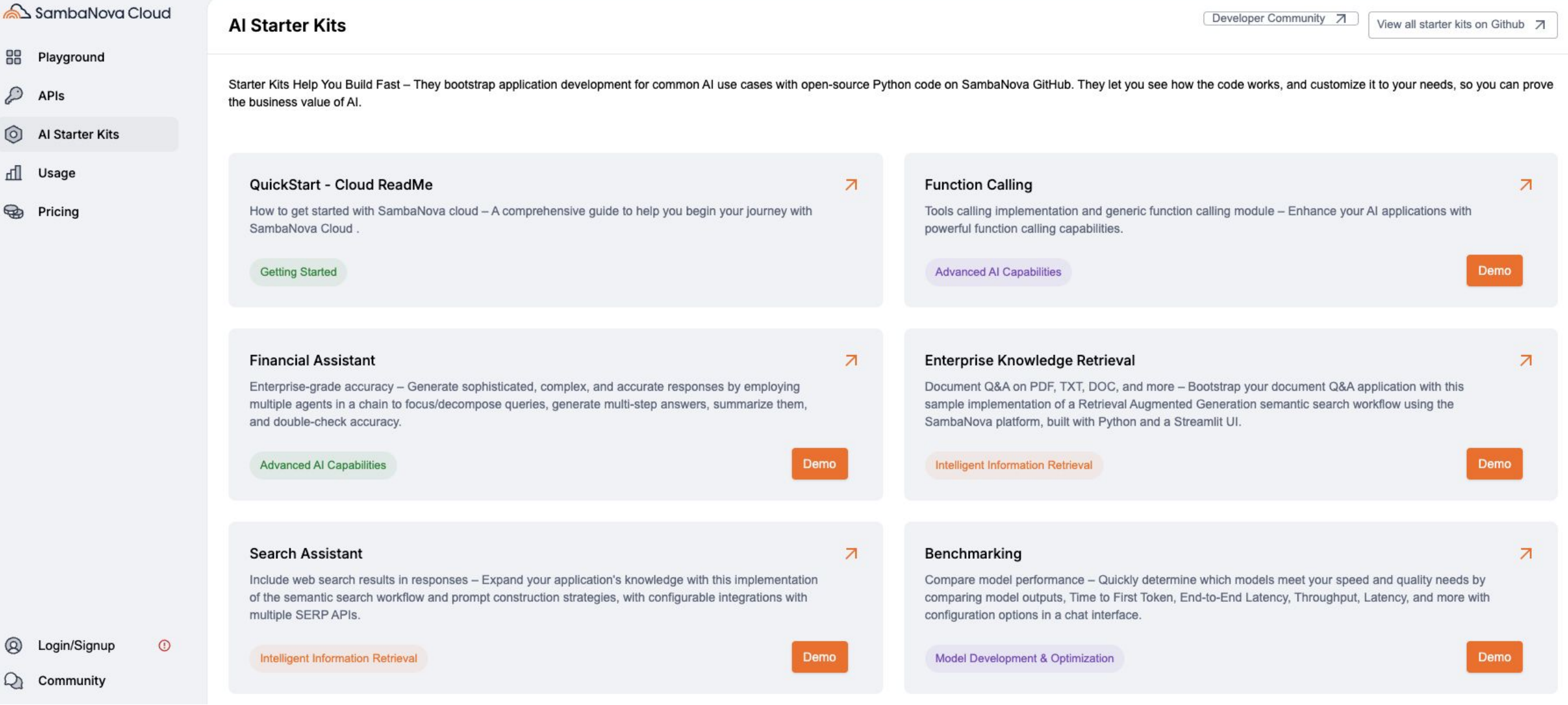
Get Your API Key at cloud.sambanova.ai





SambaNova SYSTEMS

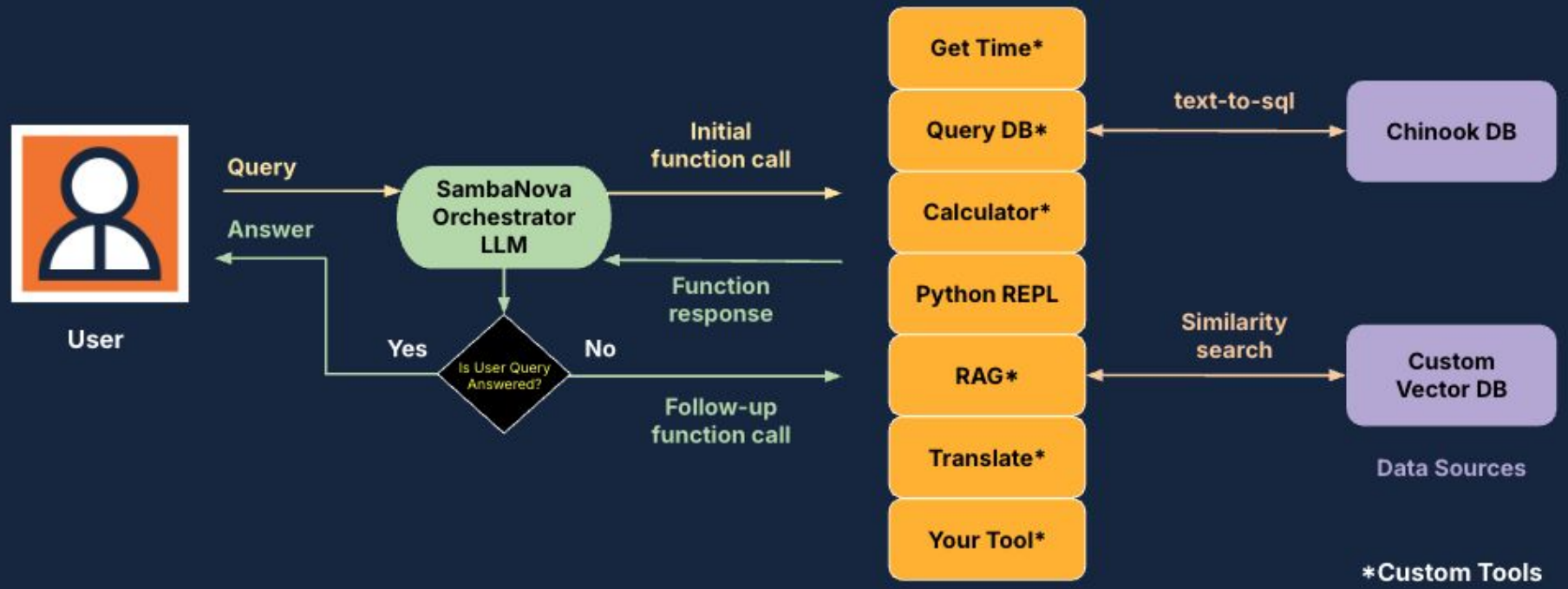
AI Starter Kits



Tool/Function Calling

ChatBot that invokes external tools relevant to user inputs

Easily enable business use cases by adding function calls into NLP applications that do more than understand text by querying databases, performing calculations, talking to other applications, and more.



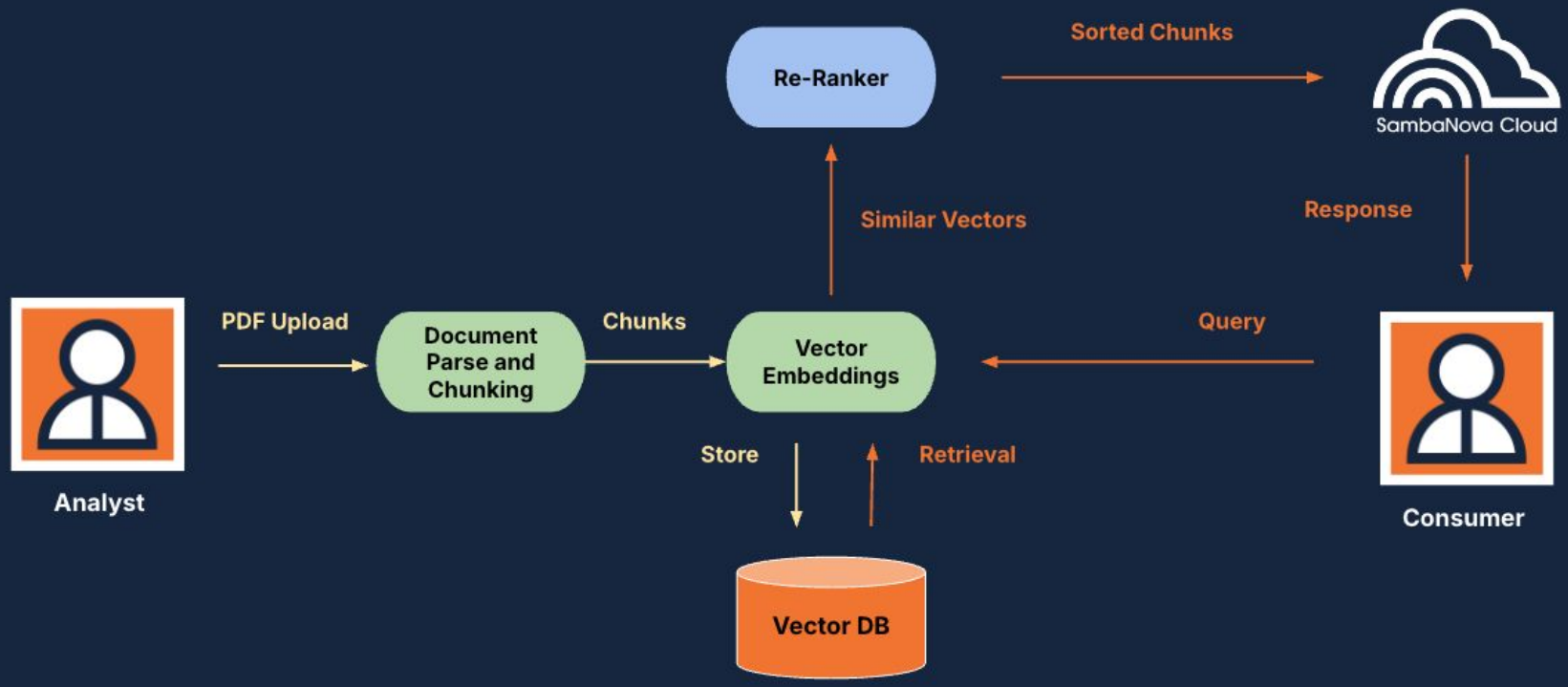
View Step-by-Step Instructions for This Kit on GitHub

View Demo-Video on YouTube

Retrieval Augmented Generation (RAG)

Document Q&A on PDF, TXT, DOC, and more

Bootstrap your document Q&A application with this sample implementation of a RAG semantic search workflow using the SambaNova platform, built with Python and a Streamlit UI.



View Step-by-Step Instructions for This Kit on GitHub

View Demo-Video on YouTube

SambaNova SYSTEMS

Gradio-SambaNova Integration

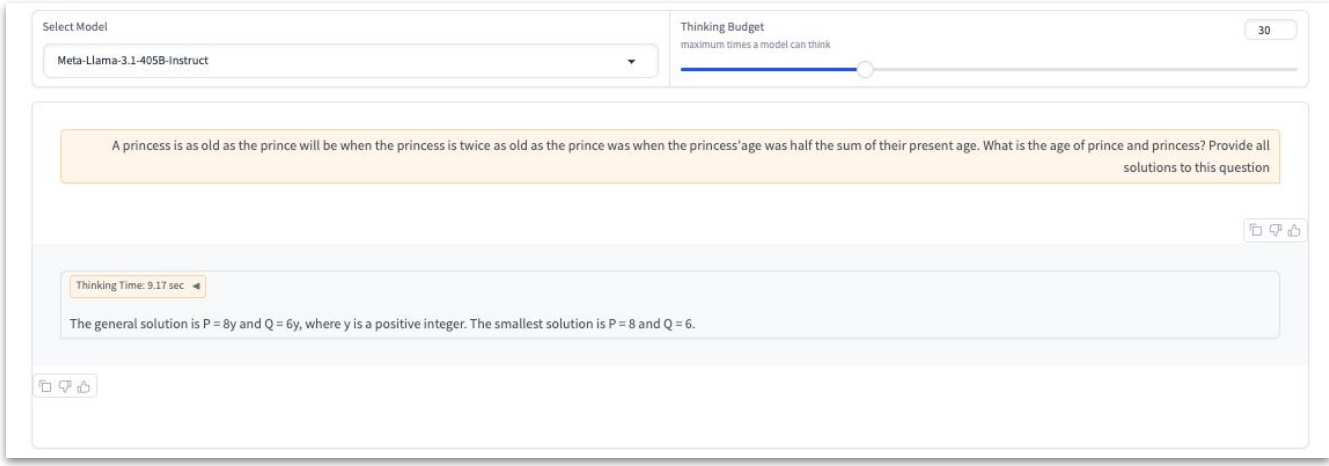
- Enables users to create web apps powered by SambaNova models using Gradio's gr.load() function
- 1 line of code to start a Gradio chat interface connected with SambaNova models (more details on GitHub):

Curl Python Gradio

```
import os
import gradio as gr
import sambanova_gradio

os.environ['SAMBANOVA_API_KEY'] = '<your_api_key>'

gr.load(
    name='Meta-Llama-3.1-8B-Instruct',
    src=sambanova_gradio.registry,
).launch()
```



Llama3.1-Instruct-O1 Demo on Hugging Face Spaces
(Try It for Yourself!)

4. AI Training on SambaNova DataScale

SambaNova SYSTEMS

PyTorch to SambaFlow: How Does It Work?

- Import your model from PyTorch
- Run **`samba.from_torch_model_(...)`** to convert model parameters to SambaTensors
  - Convert input Torch Tensors with **`samba.from_torch_tensor(...)`**
- Run **`samba.session.compile(...)`** to compile the model
  - Sometimes requires adaptations for compatibility (more details coming up)
- Start running via **`samba.session.run(...)`** and **`samba.utils.trace_graph(...)`**

Sample Code (Model)

- Model code all in PyTorch
- A few restrictions
  - Modules with parameters are defined before forward
  - The return type of forward should be simple (tensor, tuple/list of tensors)

```
class ResFFNLogReg(nn.Module):
    """Feed Forward Network with two different activation functions and a residual connection"""
    def __init__(self, num_features: int, hidden_size: int, num_classes: int) -> None:
        super().__init__
        self.gemm1 = nn.Linear(num_features, hidden_size, bias=True)
        self.gemm2 = nn.Linear(hidden_size, hidden_size, bias=True)
        self.gemm3 = nn.Linear(hidden_size, num_classes, bias=True)

        self.norm1 = nn.LayerNorm(hidden_size)
        self.norm2 = nn.LayerNorm(hidden_size)

        self.tanh1 = nn.Tanh()
        self.sigmoid1 = nn.Sigmoid()

        self.criterion = nn.CrossEntropyLoss()

        self.apply(basic_weight_init)

    def forward(self, inputs: torch.Tensor, targets: torch.Tensor) -> Tuple[torch.Tensor]:
        out = self.gemm1(inputs)
        out = self.norm1(out)
        out = self.tanh1(out)
        residual = out
        out = self.gemm2(out)
        out = self.norm2(out)
        out = out + residual
        out = self.sigmoid1(out)
        out = self.gemm3(out)
        loss = self.criterion(out, targets)
        return loss, out
```

SambaNova SYSTEMS

Sample Code (App Code)

- Running is 2 Steps, **compile** then **run**
- Need to pass **model**, **inputs**, and **optimizer** to compile/trace
- Write your own training loop!

```
import samba

#Input definition: “Dummy” SambaTensors
image = samba.randn(args.batch_size, args.num_features, name='image', batch_dim=0)
label = samba.randint(args.num_classes, (args.batch_size, ), name='label', batch_dim=0)
inputs = (image, label)

# Model definition
model = ResFFNLogReg(args.num_features, args.hidden_size, args.num_classes)
samba.from_torch_model_(model)

# Optimizer definition
optim = sambaflow.samba.optim.SGD(model.parameters(),
                                  lr=args.lr,
                                  momentum=args.momentum,
                                  weight_decay=args.weight_decay)

# Compilation & Running, or training, a model must be explicitly carried out
if args.command == "run":
    # Trace the graph
    utils.trace_graph(model, inputs, optim, pef=args.pef, mapping=args.mapping)
    # Within the user defined train function, call: samba.session.run
    train(args, model)
else:
    samba.session.compile(
            model=model,
            inputs=inputs,
            optim=optim,
            name=model.__class__.__name__,
            init_output_grads=not args.inference,
            )
```

SambaNova SYSTEMS

Various Args and Run Modes

- Args used internally expressed as command line args
- Some important args for compile/run:
  - `command`
  - `--inference`
  - `--batch-size/-b, --microbatch-size/-mb`
  - `--pef/-p`

An example compile command:

```
python <app>.py compile -b=64 -mb=4
--inference -p <app.pef>
```

```
args = parse_app_args(argv)
args.command == “compile”
args.batch_size == 64
args.microbatch_size == 4
args.inference == True
```

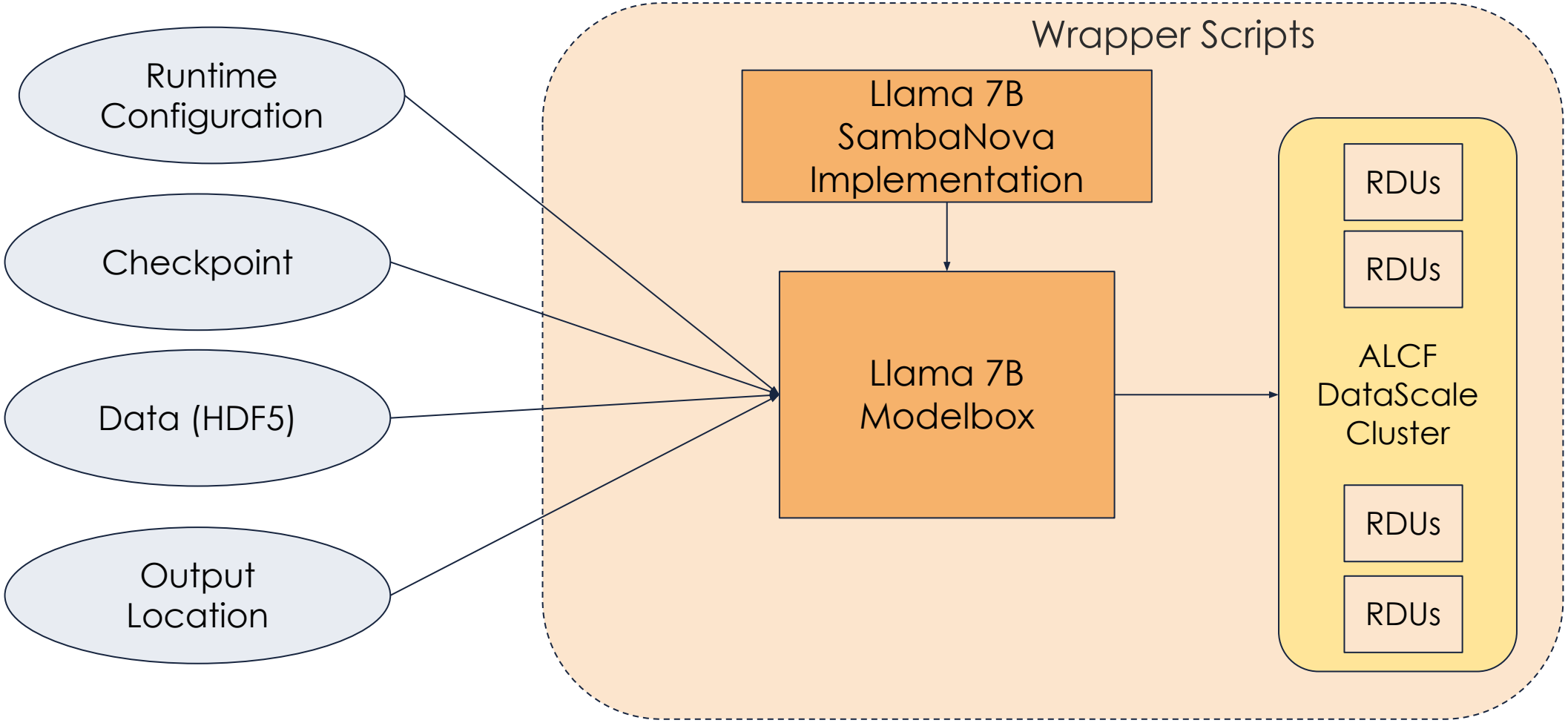
An example run command:

```
python <app>.py run -b=64 -mb=4
--inference -p <app.pef>
```

```
args = parse_app_args(argv)
args.command == “run”
args.batch_size == 64
args.microbatch_size == 4
args.inference == True
```

SambaNova SYSTEMS

LLM Deployment using Llama 7B ModelBox



User Provided

SambaNova Provided and Installed in ALCF Environment

ALCF Provided

SambaNova SYSTEMS

Wrapper Script Details: *7B_modelbox_finetuning_setup.sh*

```
NTASKS=$((NNODES*16))
echo "NNODES = ${NNODES} ; GRES = 8 ; NTASKS = $NTASKS" >> ${OUTPUT_PATH} 2>&1
sbatch --gres=rdu:8 -n ${NTASKS} --ntasks-per-node 16 --nodes ${NNODES} --cpus-per-task=8 /data/ANL/scripts
/modelbox/7B_llama2_modelbox_fine_tuning_run.sh $1 $2 >> ${OUTPUT_PATH} 2>&1
```

Prepares the job for workload orchestration tool

Number of RDUs that will be used for the job.

8 x RDUs per node

Maxim number of ntasks = 16
This is specific to the model architecture.

OUTDIR=/data/scratch/${USER}/${MODEL_NAME}

SambaNova SYSTEMS

Wrapper Script Details: 7B_modelbox_finetuning_run.sh (1)

- Good practice to check state of systems before running
  - /opt/sambaflow/bin/snfadm

```
#########################
echo "Machine State Before: " >> ${OUTPUT_PATH} 2>&1
(/opt/sambaflow/bin/snfadm -l inventory | grep -v Online) >> ${OUTPUT_PATH
} 2>&1
#########################
#########################
export CKPT_PATH=/nvmedata2/data/ANL/ckpt/Llama-2-7b-hf-bf16/
export DATA_PATH=/data/ANL/superglue_4k_2/hdf5
```

Where data is

Where checkpoint path is specified. Can be changed to load from a checkpoint.

- Script automatically exits if these are not specified correctly
- OUTDIR=/data/scratch/${USER}/${MODEL_NAME}
  - Shows results of the slurm job, so this should be monitored

SambaNova SYSTEMS

Wrapper Script Details: 7B_*modelbox_finetuning_run.sh* (2)

```
export OUTPUT_DIR=${OUTDIR}/output_$(hostname)

export WARMUP_STEPS=0
export LOG_STEPS=1
export LEARNING_RATE="1e-5"
export STEPS_THIS_RUN=100
export PEF=/opt/pefs/llama2_7b_ss4096_bs8_training_3d_atten_mixedp_mha_fp32mw_32_enc_vocab_size_32k_rchigh.
pef
```

User parameters

Where checkpoints from model run are saved

PEF can be found within Modelbox image or on host

SambaNova SYSTEMS

Wrapper Script Details: 7B_modelbox_finetuning_run.sh (3)

```
srun --mpi=pmi2 \
  singularity exec --writable-tmpfs \
        --bind $CKPT_PATH:/opt/ckpt_path \
        --bind $DATA_PATH:/opt/data_dir \
        --bind $OUTPUT_DIR:/opt/hf_output \
        --bind /usr/local/etc/slurm.conf:/etc/slurm-llnl/slurm.conf \
        --bind /run/munge/munge.socket.2 \
        --bind /tmp:/tmp \
        --bind /dev/log:/dev/log \
        --bind /run/systemd/journal \
        --bind /opt/sambaflow/pef/:/opt/sambaflow/pef/ \
        --bind /opt/sambaflow/runtime:/opt/sambaflow/runtime \
        --bind $PEF:/var/tmp/pef_47.pef \
python3 /opt/sambaflow/apps/nlp/transformers_on_rdu/transformers_hook.py run \
--log-level error \
--article_attention \
--batch-size 8 \
--config_name /opt/ckpt_path/config.json \
--data_dir /opt/data_dir \
--data-parallel \
```

Binds for container

Python flags

Example Models to Run on DataScale SN30 at ALCF

DataScale SN30					
NLP models	GPT1.5B, GPT13B, BERT, Llama V2 7B, 13B, 70B, Llama V3, Genslm, Mistral, Deepseek Coder	**Vision models**	AutodiCNN, CosmicTagger, Unet2D, Unet 3D, DeepVIT, Rescalenet, dcrnn, Vit, AutophaseNN	**Science models**	Uno, BraggNN, AI4Polymers

- Get more details on Sambanova Public Docs
  - SambaFlow developer documentation
  - SambaNova documentation at ALCF

SambaNova SYSTEMS

THANK YOU!

petro-junior.milan@sambanova.ai

SambaNova® SYSTEMS

Join us
Booth #2309
sambanova.ai/sc24

Meet the Experts and Happy Hours at Booth #2309

Tuesday, November 19

- 1:00 p.m. - 3:30 p.m.
  SambaNova Customer Experts

- 4:00 p.m. - 5:00 p.m.
  SambaNova Experts Happy Hour

- 5:00 p.m. - 6:00 p.m.
  SambaNova Partner Experts Happy Hour

Wednesday, November 20

- 1:00 p.m. - 2:30 p.m.
  Meet SambaNova Customer Experts

Birds of a Feather

Wednesday, November 20

- 5:15 p.m. - 6:45 p.m.
  Democratizing AI Accelerators for HPC Applications: Challenges, Success, and Support

- 5:15 p.m. - 6:45 p.m.
  The National Artificial Intelligence Research Resource (NAIRR) Pilot User Experience BoF

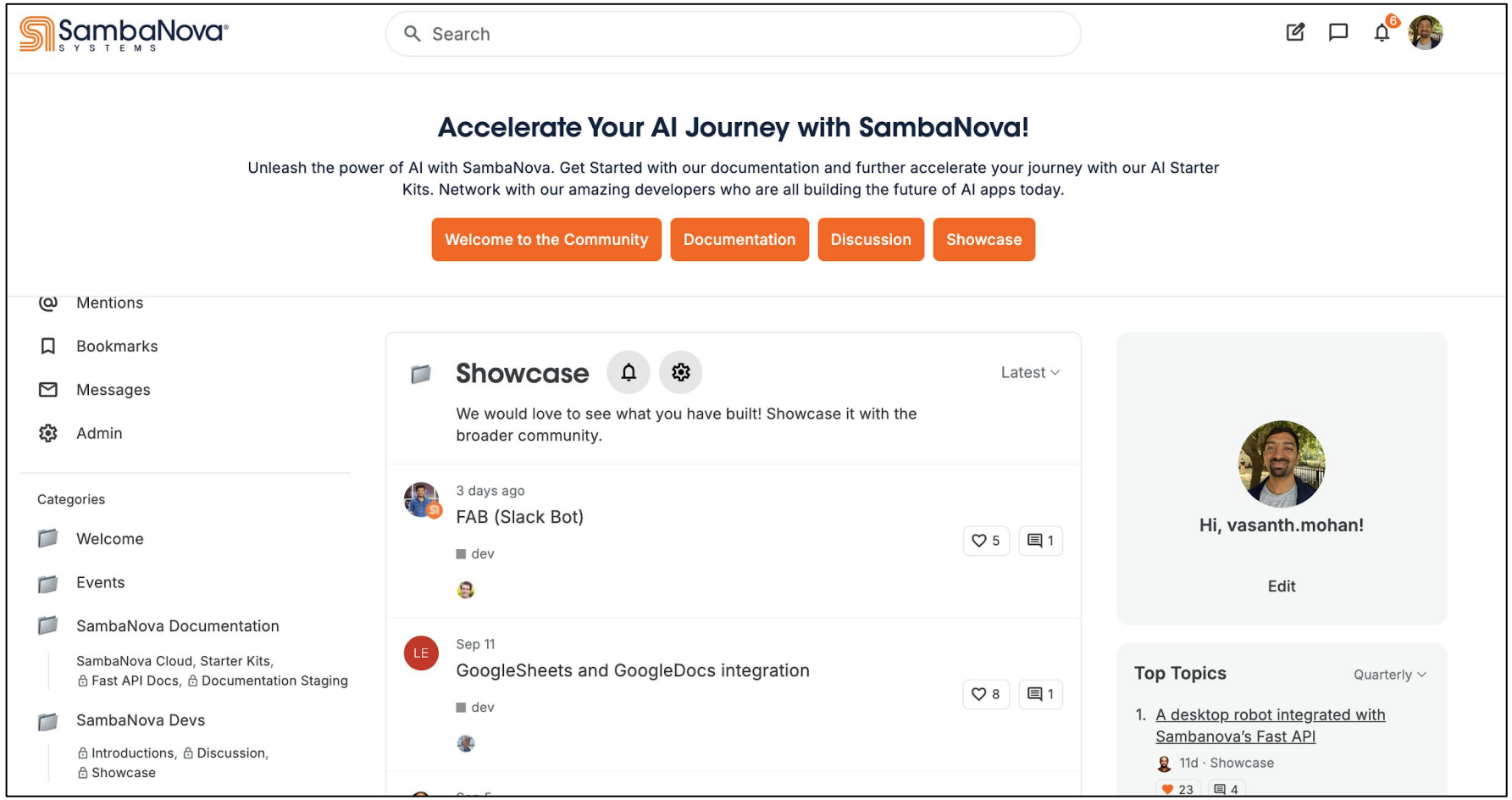
SambaNova® SYSTEMS

Try It Today

cloud.sambanova.ai

SambaNova® SYSTEMS

Questions? Join the Community



Community.SambaNova.ai