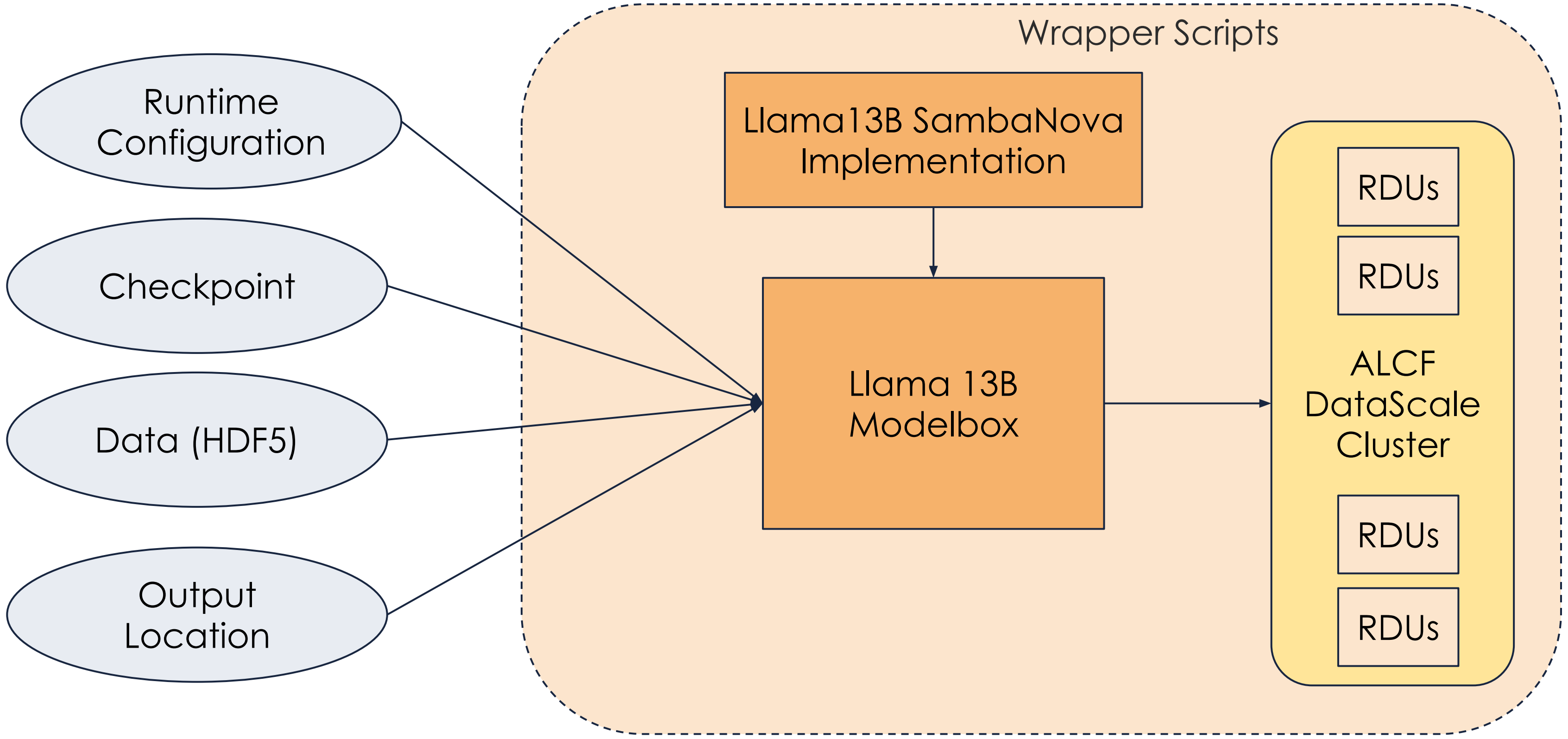
Large Language Models on DataScale SN30

April 2024

How to Use a Modelbox for a Large Language Models

- Modelbox is the simplest way for end users to execute preverified SambaNova model implementations
- To run a Modelbox, end users must:
  1. Understand the inputs that must be provided
  2. Leverage the wrapper scripts for easy Pretraining, Fine Tuning, and Inference

Overview: LLM Deployment Using Llama 13B Modelbox



▭ **User Provided**

🟧 **SambaNova Provided and Installed in ALCF Environment**

🟨 **ALCF Provided**

Details on User Provided Information (Example Requirements for Llama 13B Modelbox)

User Provided Details for Llama 13B Modelbox

Runtime Configuration	• Details on user specific runtime configuration are required as part of Wrapper Scripts for execution • Defaults are provided, but users may change configurations such as, hyperparameters
Checkpoint	• For pretraining, randomized initialization is needed • For fine tuning or inference, leverage Hugging Face or bring your own checkpoint
Data (HDF5)	• Example datasets are pre-installed • Specific use cases will require preparation of custom data sets
Output Location	• Need to reserve specific space for checkpoints and logs

SambaNova SYSTEMS

User Provided Details: *Runtime Configuration*

- The wrapper scripts include default runtime configuration, but some changes are allowed by users

Example Changes	
Training Hyperparameters	Batch size, workers, learning rate/schedule, weight decay, warmup
Inference Hyperparameters	Sampling, seeding
Other	Logging and checkpoint frequency

SambaNova SYSTEMS

User Provided Details: *Checkpoint*

- The necessary tokenizer and weights for common open source models are typically stored by ANL
  - Consult with ANL team on common location or request to download

- Example directory structure (includes tokenizer):

```
added_tokens.json   config.json                pytorch_model-00001-of-00003.bin
pytorch_model-00003-of-00003.bin  special_tokens_map.json  tokenizer.json
generation_config.json  pytorch_model-00002-of-00003.bin
pytorch_model.bin.index.json      tokenizer_config.json    tokenizer.model
```

Size: ~13GB

User Provided Details: *Sample Data (HDF5)*

- Users can leverage sample datasets that are pre-installed for users to become familiar with Modelbox

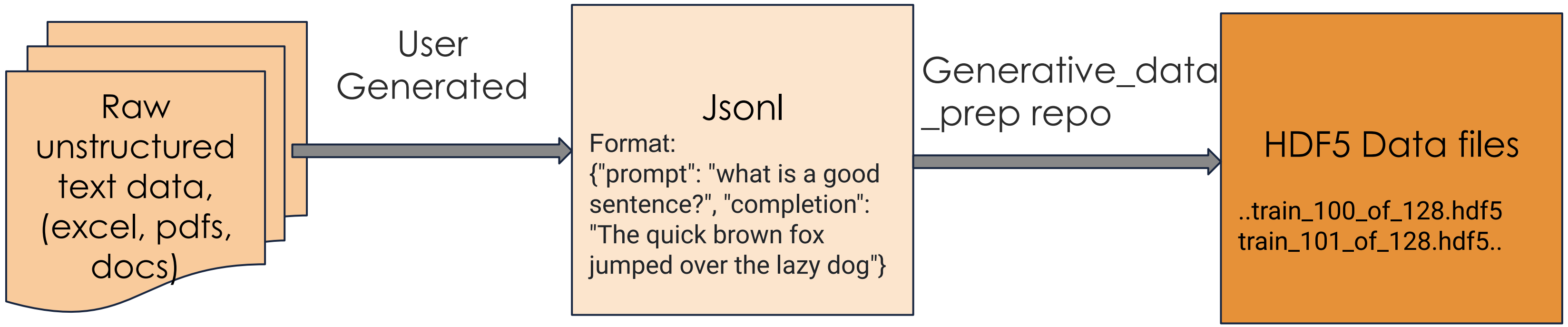
Type	**Name**	**Location**	**Size**
Pretraining	openwebtext	/data/ANL/openwebtext_4k/hdf5/hdf5	31 GB
Fine tuning	Superglue	/data/ANL/superglue_4k_2/hdf5	86 MB
Inference	Generative prompts	/software/sambanova/singularity/images/llm-modelbox/datasets/generative_prompts	

- Datasets are for facilitating end to end examples only; validity and quality of dataset is not guaranteed

User Provided Details: *Preparing a Custom Dataset (HDF5)*

- SambaNova provides a dataset preparation tool that provides a common structure to input data and process it in a way that can be consumed by a Modelbox container
  - Github: https://github.com/sambanova/generative_data_prep
  - Supports pretraining, continual pretraining and fine tuning
  - Capabilities include shuffling, splitting, various Hugging Face tokenizers, and packing

- Steps include:
  1. Setup data preparation repo
  2. Make sure the data is in the expected format
  3. Execute data processing


Raw unstructured text data, (excel, pdfs, docs)
User Generated
Jsonl
Format:
{"prompt": "what is a good sentence?", "completion": "The quick brown fox jumped over the lazy dog"}
Generative_data_prep repo
HDF5 Data files
..train_100_of_128.hdf5
train_101_of_128.hdf5..

Generative Dataset Preparation: *Repo Setup*

- For repo setup, it is recommended to use a python virtual or conda environment

- Steps for repo setup are:

```
> git clone https://github.com/sambanova/generative_data_prep.git

> cd generative_data_prep

> python virtual -m venv gen_data_env (python 3.9)

> source gen_data_env/bin/activate

> pip install .
```

- To verify installation and that generative data prep is working properly:

```
> pip install -r requirements/tests-requirements.txt

> pytest
```

Generative Dataset Preparation: *Expected Data Format*

- Input is expected to be of jsonl file format (https://jsonlines.org/examples/)
- Pretraining:

```
{"prompt": "", "completion": "I have mentioned in many other articles “}
```

Note: .txt is also allowed

- Fine Tuning:

```
{"prompt": "what is a good sentence?", "completion": "The quick brown fox jumped
over the lazy dog"}
```

Generative Dataset Preparation: *Execute Data Processing*

- Once the repo is setup, data processing can begin with just a few items to customize the data preparation pipeline:
  - Path to input file
  - Path to output file
  - A few arguments

```
python3 -m generative_data_prep pipeline
--input_file_path=<PATH TO DATASET FILE> \
--output_path=<PATH TO OUTPUT DIRECTORY> \
--pretrained_tokenizer=Llama-2 \
--max_seq_length=4096 \
--input_packing_config='greedy::drop' \
--shuffle=on_RAM \
…
```

- The full list of flags can be found on github

Generative Dataset Preparation: *Example JSONL -> HDF5*

- jsonl:

{"text": "Judge arrested in Aruba case Fifth suspect in custody after U.S. teen's disappearance Paul Van Der Sloot was arrested after being questione…….sts? \u0095 Interactive: Safety tips for travelers YOUR E-MAIL ALERTS Aruba Alabama Crime, Law and Justice or or Create Your Own ORANJESTAD, ……………….", "meta": {"title": "Judge arrested in Aruba case", "lang": "en", "url": "http://www.cnn.com/2005/LAW/06/23/aruba.holloway/index.html", "word_count": 1260, "elapsed": 2.1922969818115234, "scraper": "newspaper", "domain": "www.cnn.com", ….."subreddits": ["reddit.com"], "reddit_created_utcs": ["2005-06-23T21:07:16"]}}

- Command:

```
#! /bin/bash
DATA_PREP_PYTHONPATH=/<local filepath>/generative_data_prep
PYTHONPATH=$DATA_PREP_PYTHONPATH:$PYTHONPATH python -m generative_data_prep pipeline --input_file_path=/<local
filepath>/openweb_text.jsonl --output_path=/data/scratch/$USER/openwebtext_4k/ --dev_ratio=0.0 --test_ratio=0.0
--shuffle large_file --pretrained_tokenizer=/<local filepath>/Llama-2-13b-hf/ --completion_keyword=text
--num_workers 16 --max_seq_length=4096  --input_packing_config full
```

- Output:

```
>>> import h5py
>>> f = h5py.File('train_100_of_128.hdf5')
>>> f['input_ids'][0]
array([    1, 29871,  7307, ...,  8145, 21629, 30019], dtype=int32)
>>> f.keys()
<KeysViewHDF5 ['input_ids', 'token_type_ids']>
```

Generative Dataset Preparation: *Batch Size Dependency*

- Tips and tricks for batch size:
    - The dataset should be large enough to run one batch of training
    - The number of sequences in the output dataset files satisfy this by checking ***`max_batch_size_train`*** in the ***`<OUTPUT_DIR>/metadata.yaml`*** file.
    - Use this value to set batch_size accordingly when starting a training job

User Provided Details: *Output Location*

- Default location included in wrapper script is:
  - `/data/scratch/${USER}/${MODEL_NAME}`

- Key considerations:

Item	Details
Disk Space	~25 GB per checkpoint for Llama 13B
Location	In distributed data parallel, location must be accessible across all nodes

- Output directory structure

```
epoch.txt  global_train_steps.txt  learning_rate.txt   step_loss.txt  train_loss.txt
train_steps_per_second.txt  train_steps.txt: ML logger outputs for plotting/tracking
ml_app_debug.log  ml_app_info.log: Running trace
step_102 step_204: Generated checkpoints
```

Details on Wrapper Scripts (Example of Llama 13B Modelbox)

Wrapper Scripts for Llama 13B Modelbox

Wrapper Scripts

located under /data/ANL/scripts/

Type	Details	Location	Args
Pretraining	Submits a pre training job of Llama 13B model on openwebtext dataset to slurm	**13B_modelbox_pretraining_setup.sh** 13B_modelbox_pretraining_run.sh	Number of nodes, LogDir
Fine tuning	Submits a fine tuning job of LLama 13B model using Hugging Face checkpoint on superglue dataset to slurm	**13B_modelbox_fine_tuning_setup.sh** 13B_modelbox_fine_tuning_run.sh	Number of nodes, LogDir
Inference	Submits a job to generate 100 tokens for prompt “once upon a time” using Hugging Face checkpoint	**13B_modelbox_inference.sh**	LogDir

Assumes Modelbox .sif file is at: /software/sambanova/singularity/images/llm-modelbox/llama_v2_13B.sif

Use setup.sh, not run.sh scripts

Wrapper Script Difference for Llama 13B Modelbox

Wrapper Scripts

located under /data/ANL/scripts/

Type	**Name**	**Details**
Setup	*setup.sh	• Creates out directory in /data/scratch/$USER/ • Allows specific bindings to be used with container for user • Setups the container to be used with Slurm with sbatch
Run	*run.sh	• Creates output directory in /data/results/$(hostname)/$USER/ for slurm job • Saves user specific parameters • Runs the specific python and related flags for the model. Runs the job with slurm using srun
Inference	*inference.sh	• Uses specific PEF for optimized performance • Uses a specific flag (--inference), additional flags, and specific data

Assumes Modelbox .sif file is at: /software/sambanova/singularity/images/llm-modelbox/llama_v2_13B.sif

Use setup.sh, not run.sh scripts

SambaNova SYSTEMS

Using the Wrapper Scripts for Llama 13B Modelbox

- Example pretrain from scratch run command:

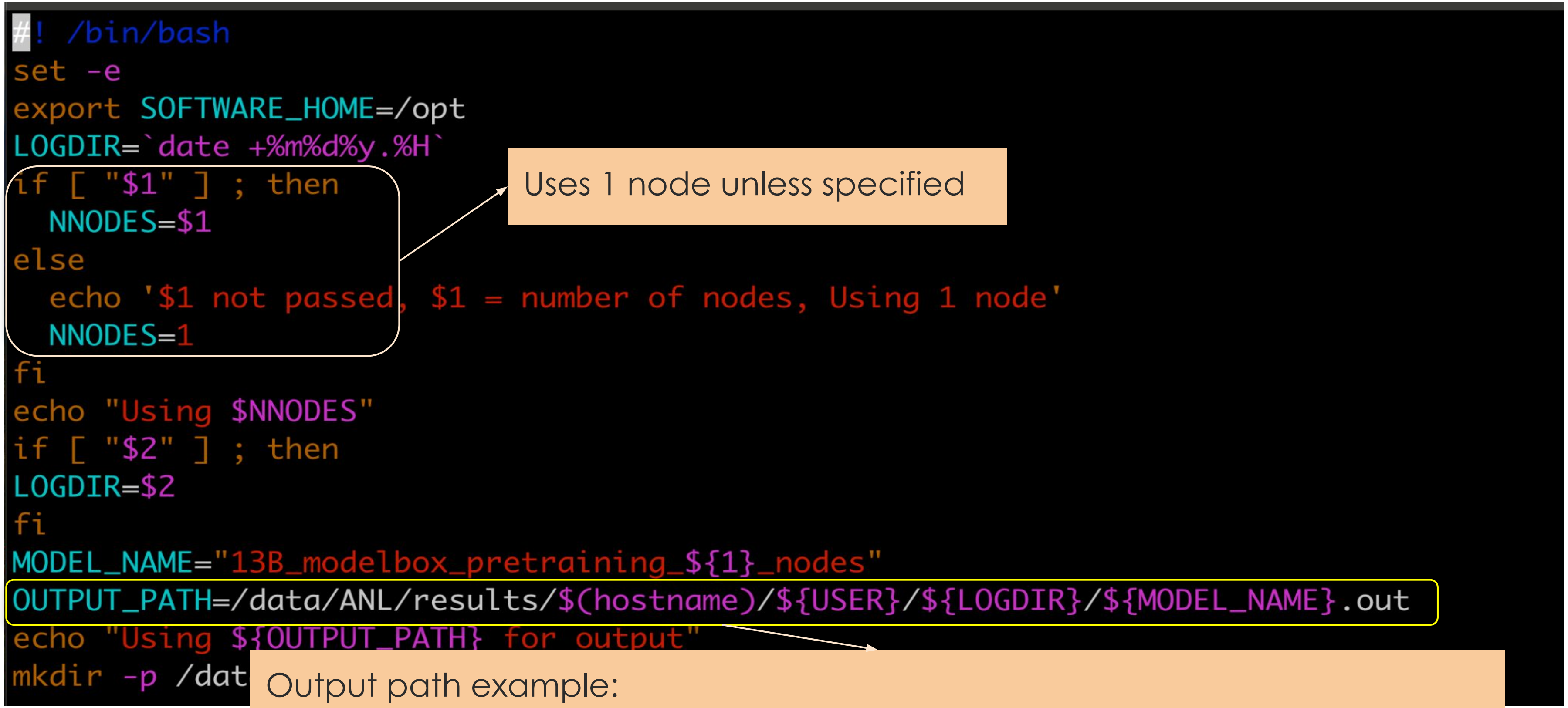
```
/data/ANL/scripts/13B_modelbox_pretraining_setup.sh 2 llama13B_results
```

  Assumes the following:

  + The following script takes two optional arguments: **`$1=Nodes,`** which is the number of nodes to use, and **`$2=Results`**, which is the directory name for results
    + If $1 is not specified, 1 node is used
    + If $2 is not specified, the date at the time the command is run is used for the directory name of the results
  + Automatic slurm integration
  + `OUTDIR=/data/scratch/$USER, OUTPUT_PATH=/data/ANL/results/`

- Recommendations
  + Alter scripts my copying to your scratch directory then altering:
    `cp -r /data/ANL/scripts/13B*.sh /data/scratch/$USER`
  + Do not run scripts from /data/ANL/scripts; run from scratch directory
  + Use setup.sh scripts rather than run.sh scripts

Wrapper Script Details: *13B_modelbox_pretraining_setup.sh (1)*



```
#! /bin/bash
set -e
export SOFTWARE_HOME=/opt
LOGDIR=`date +%m%d%y.%H`
if [ "$1" ] ; then
   NNODES=$1
else
   echo '$1 not passed, $1 = number of nodes, Using 1 node'
   NNODES=1
fi
echo "Using $NNODES"
if [ "$2" ] ; then
LOGDIR=$2
fi
MODEL_NAME="13B_modelbox_pretraining_${1}_nodes"
OUTPUT_PATH=/data/ANL/results/$(hostname)/${USER}/${LOGDIR}/${MODEL_NAME}.out
echo "Using ${OUTPUT_PATH} for output"
mkdir -p /dat
```

Uses 1 node unless specified

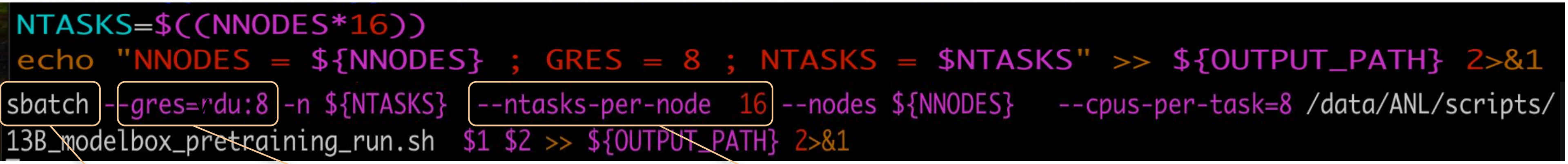
Output path example:

OUTPUT_PATH=/data/ANL/results/sn30-r3-h1/$USER/llama13B_results/13B_modelbox_pretraining_2_nodes.out

OUTPUT_PATH=/data/ANL/results/sn30-r3-h1/$USER/MMDDYY.HH/13B_modelbox_pretraining_1_nodes.out

SambaNova SYSTEMS

Wrapper Script Details: *13B_modelbox_pretraining_setup.sh (2)*



```
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/
13B_modelbox_pretraining_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.

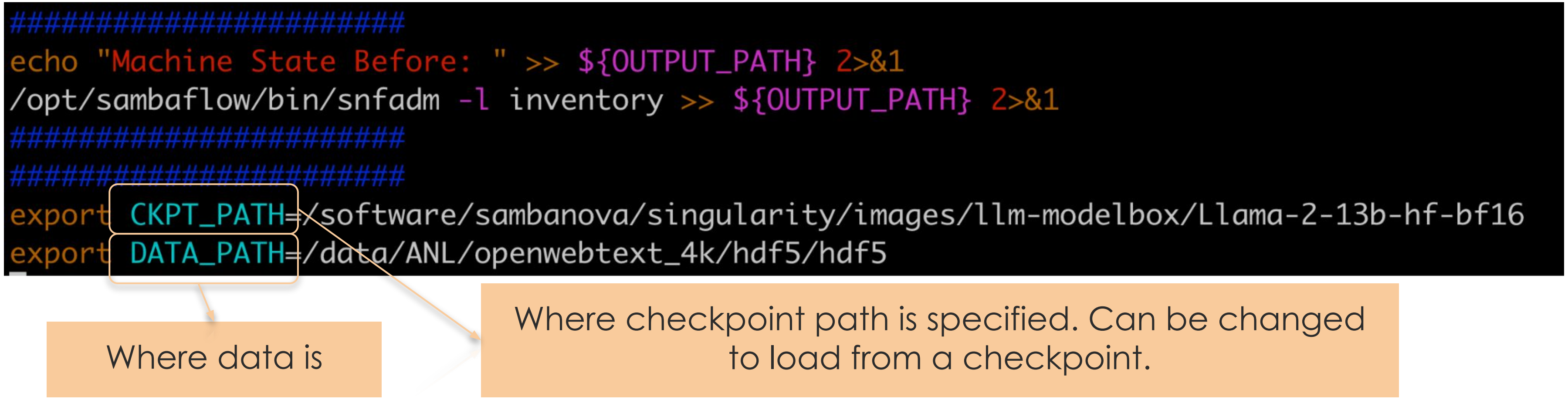
8 x RDUs per node

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

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

Wrapper Script Details:13B_*modelbox_pretraining_run.sh (1)*

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

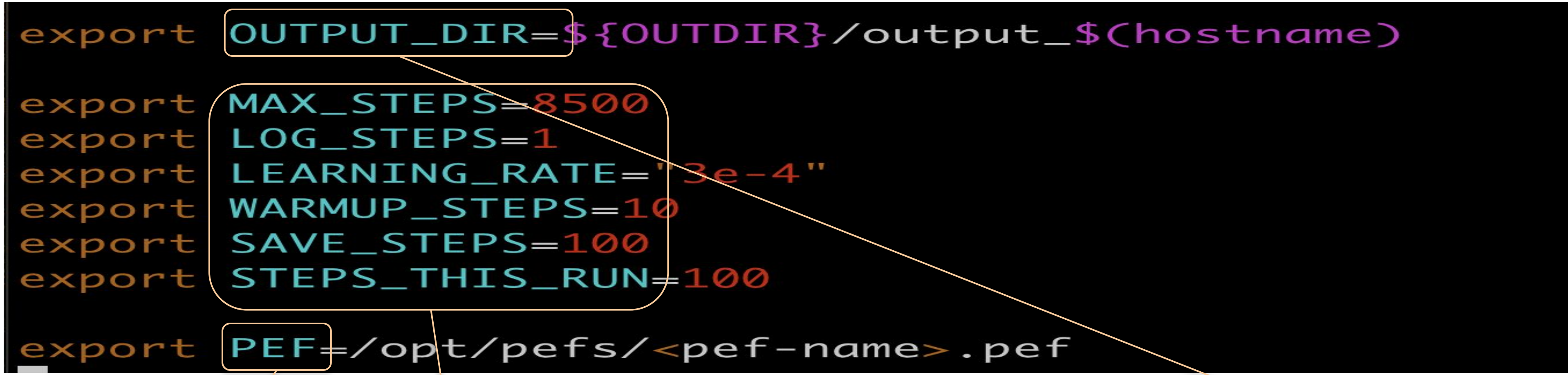


```
########################
echo "Machine State Before: " >> ${OUTPUT_PATH} 2>&1
/opt/sambaflow/bin/snfadm -l inventory >> ${OUTPUT_PATH} 2>&1
########################
########################
export CKPT_PATH=/software/sambanova/singularity/images/llm-modelbox/Llama-2-13b-hf-bf16
export DATA_PATH=/data/ANL/openwebtext_4k/hdf5/hdf5
```

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

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

Wrapper Script Details:*13B_modelbox_pretraining_run.sh* (2)



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

export MAX_STEPS=8500
export LOG_STEPS=1
export LEARNING_RATE="3e-4"
export WARMUP_STEPS=10
export SAVE_STEPS=100
export STEPS_THIS_RUN=100

export PEF=/opt/pefs/<pef-name>.pef
```

User parameters

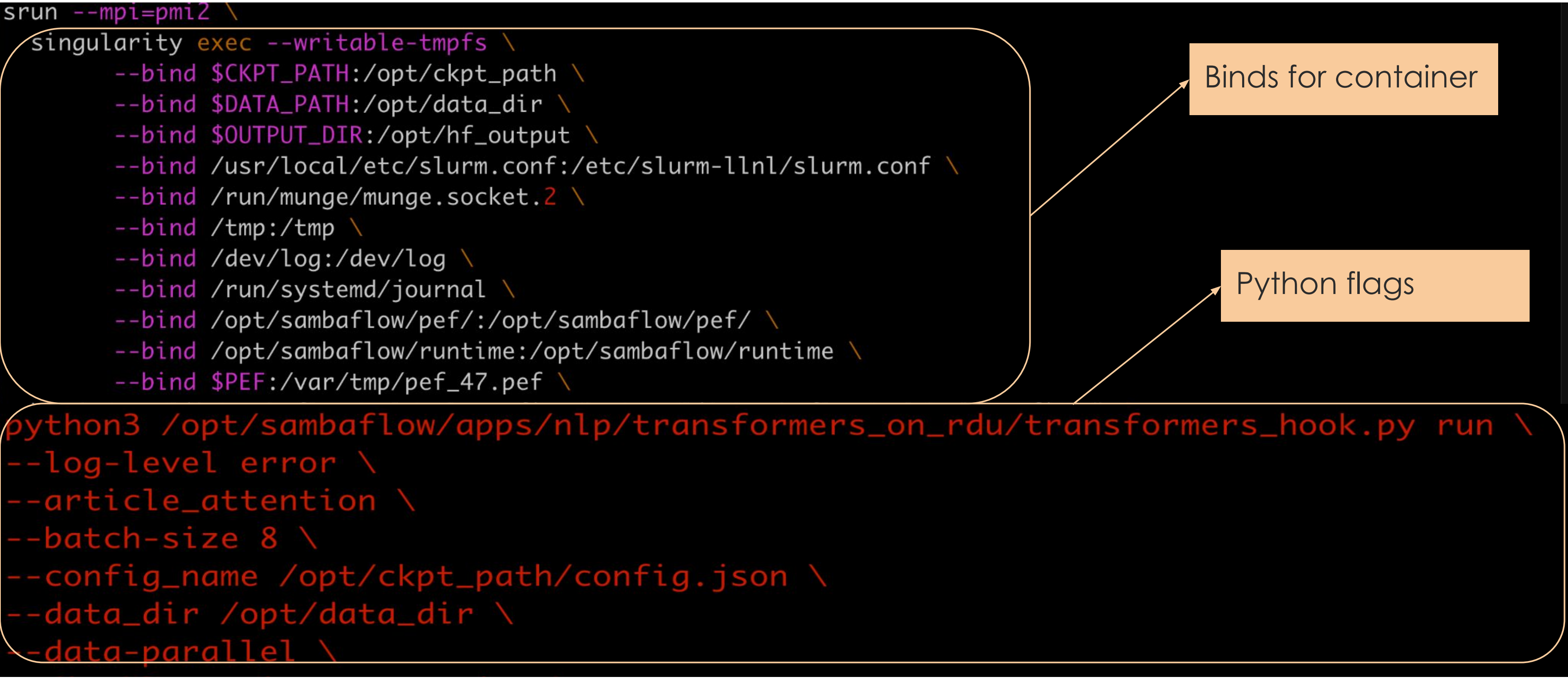
PEF can be found within Modelbox image or on host

Where checkpoints from model run are saved

Ex:

data/scratch/$USER/13B_MODELBOX_pretraining_2_nodes/output_sn30-r3-h1/

Wrapper Script Details:13B_modelbox_pretraining_run.sh (3)

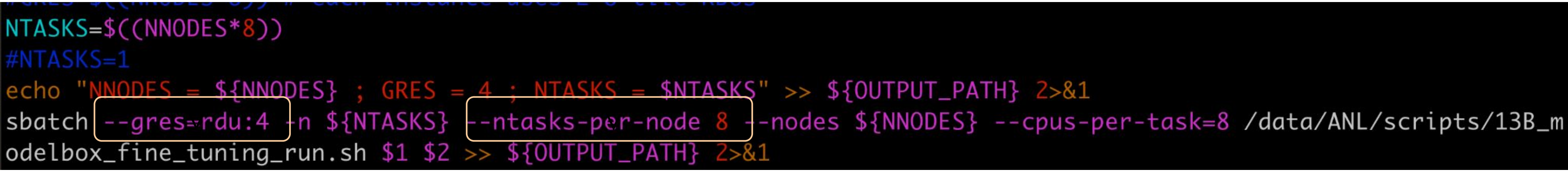


```
srun --mpi=pmi2 \
  singularity exec --writable-tmpfs \
        --bind $CKPT_PATH:/opt/ckpt_path \
        --bind $DATA_PATH:/opt/data_dir \
        --bind $OUTPUT_DIR:/opt/hf_output \
        --bind /usr/local/etc/slurm.conf:/etc/slurm-llnl/slurm.conf \
        --bind /run/munge/munge.socket.2 \
        --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 \
```

SambaNova SYSTEMS

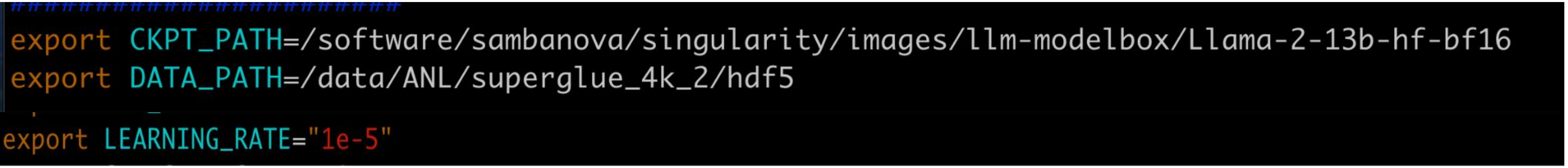
Wrapper Script Details: *13B_modelbox_fine_tuning_*.sh*

- 13B_modelbox_fine_tuning_setup.sh follows is similar to pretraining setup scripts, though it calls for only half of the node usage. This is because fine-tuning is adjusting an existing model rather than learning from scratch.



```
NTASKS=$((NNODES*8))
#NTASKS=1
echo "NNODES = ${NNODES} ; GRES = 4 ; NTASKS = $NTASKS" >> ${OUTPUT_PATH} 2>&1
sbatch --gres=rdu:4 -n ${NTASKS} --ntasks-per-node 8 --nodes ${NNODES} --cpus-per-task=8 /data/ANL/scripts/13B_m
odelbox_fine_tuning_run.sh $1 $2 >> ${OUTPUT_PATH} 2>&1
```

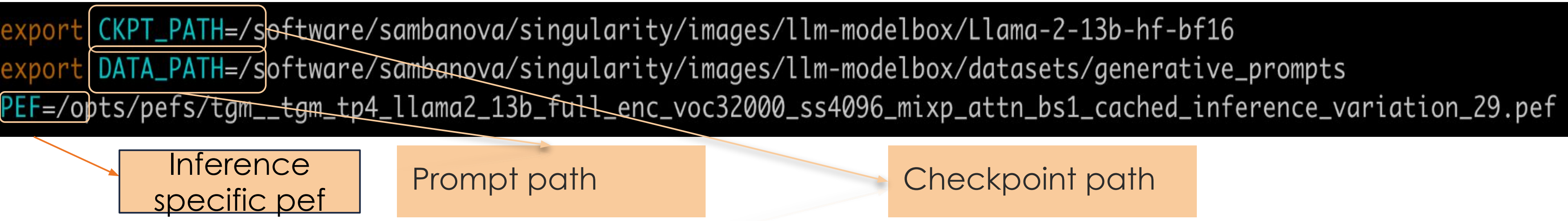
- 13B_modelbox_fine_tuning_run.sh uses a different dataset for fine tuning along with different parameters



```
export CKPT_PATH=/software/sambanova/singularity/images/llm-modelbox/Llama-2-13b-hf-bf16
export DATA_PATH=/data/ANL/superglue_4k_2/hdf5

export LEARNING_RATE="1e-5"
```

Wrapper Script Details: *13B_modelbox_inference.sh*

- Users can specify checkpoints and prompts or inference

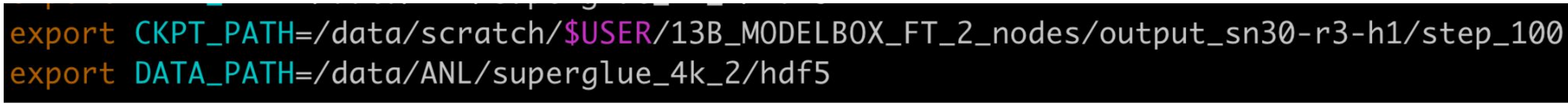


```
export CKPT_PATH=/software/sambanova/singularity/images/llm-modelbox/Llama-2-13b-hf-bf16
export DATA_PATH=/software/sambanova/singularity/images/llm-modelbox/datasets/generative_prompts
PEF=/opts/pefs/tgm__tgm_tp4_llama2_13b_full_enc_voc32000_ss4096_mixp_attn_bs1_cached_inference_variation_29.pef
```

```
python /opt/sambaflow/apps/nlp/transformers_on_rdu/generative_hook.py run \

--inference \

–flags . . . \

-p $PEF " >> ${OUTPUT_PATH} 2>&1
```

Wrapper Script Details: *Running from Existing Checkpoints*

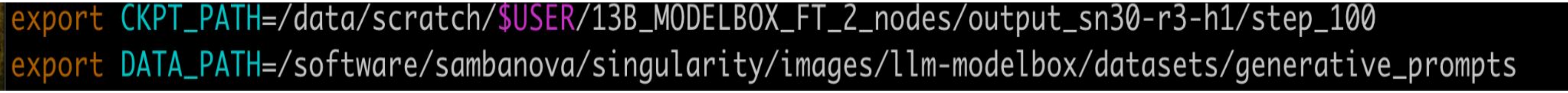
- For fine tuning of LLama 13B, example wrapper script change for checkpoint and data are:



```
export CKPT_PATH=/data/scratch/$USER/13B_MODELBOX_FT_2_nodes/output_sn30-r3-h1/step_100
export DATA_PATH=/data/ANL/superglue_4k_2/hdf5
```

Leverages a saved checkpoint where the model ran for 100 steps and saved

- For inference of Llama 13B, leverage the stored generative prompts or user specific prompts.



```
export CKPT_PATH=/data/scratch/$USER/13B_MODELBOX_FT_2_nodes/output_sn30-r3-h1/step_100
export DATA_PATH=/software/sambanova/singularity/images/llm-modelbox/datasets/generative_prompts
```

SambaNova SYSTEMS

Thank you